Proiect SIPOCA 593 – „Sistem de monitorizare a fluxurilor de deșeuri menajere și similare în scopul îmbunătățirii mecanismelor de gestionare a instrumentului economic "Plătește Pentru Cât Arunci"" Beneficiar: Ministerul Mediului, Apelor și Pădurilor

Procedura operațională de gestionare a platformei software pentru colectarea și analiza datelor privind deșeurile colectate prin programul PPCA

Autori:

Rădulescu Maria Cristina

Gorgan Vasile

Harhătă Victor-Florin

Ene Marian

Olariu Ana Alexandra

RESPONSABIL DE PROIECT ASE:

VALENTIN LAZĂR

Cuprins

Introducere

Raportul de față are drept scop principal oferirea de clarificări și recomandări cu privire la instalarea și punerea în funcțiune a sistemului informatic privind colectarea și analiza datelor prin programul PPCA.

Pentru a-și îndeplini obiectivele în mod efectiv, sistemul informatic suport pentru programul PPCA privind colectarea deșeurilor trebuie să se potrivească perfect în arhitectura IT folosită de Ministerul Mediului, atât în ceea ce privește infrastructura hardware, cât și din punct de vedere al alternativelor de ordin funcțional vizând arhitectura software. În acest sens, raportul acordă atenție deosebită operaționalizării sistemului informatic de analiză a datelor colectate prin programul PPCA prin:

- prezentarea sistemului operațional de gestionare a colectării deșeurilor
- oferirea de opțiuni de găzduire
- prezentarea etapelor necesare instalării și punerii în funcțiune a sistemului care colectează datele din sistemele operaționale
- furnizarea unui set de bune practici și proceduri necesare monitorizării și mentenanței sistemului instalat
- enunțarea unor direcții viitoare de integrare a sistemului dezvoltat cu alte soluții informatice.

1. Prezentarea aplicației operaționale

1.1 Prezentarea generala a aplicației

Cu puțin efort din partea cetățenilor, putem ajuta la reducerea poluării si contribui la pastrarea unui ritm de viata sanatos, atat pentru noi, cat si pentru mediul inconjurator. Mai exact, dorim incurajarea populatiei de a isi selecta propriul gunoi (in functie de deseul propriu-zis: gunoi biodegradabil, hartie, plastic, rezidual si sticla), in containere special amenajate din imprejurul locuintelor, platind pretul real al colectarii si transportarii deseurilor fiecaruia.

Cu ajutorul tag-urilor cu cip individual, in vederea monitorizarii volumului colectat al recipientilor per tip de deseu, soferii utilajelor de salubritate isi onoreaza intr-un mod eficace si eficient responsabilitatile cu privire la colectarea si transportarea deseurilor catre zonele special amenajate, putand astfel sa se tina evidenta gunoaielor, dupa principiul " Platesti pentru cat arunci".

1.2 Descrierea modului de utilizare al aplicației

Fiecare utilizator va primi un link distinct de conectare astfel încât sa poată accesa aplicația.

Autentificarea

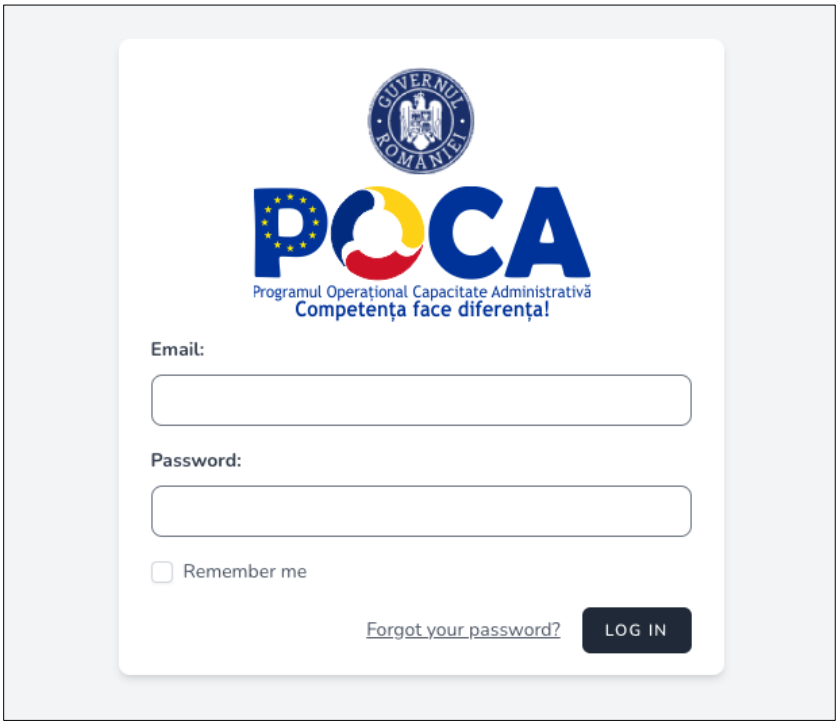


Figura 1 Autentificarea

Administratorul sistemului va transmite fiecărui colector de deșeuri datele necesare pentru conectare (**e-mail si parola**).

Ulterior, colectorul va putea schimba parola cu ajutorul unui click pe "*Forgot your password?*", urmand sa primeasca pe e-mail un link, pe care, accesandu-l, va fi redirectionat catre aceeasi pagina de inceput, adaugand de 2 ori noua parola, iar mai apoi va recompleta datele de autentificare.

Dashboard (panou de control)

Statistici - diagramele ne ofera informatii despre numarul, respectiv procentajul recipientelor de deseuri per fiecare tip existent (biodegradabil, hartie, rezidual, plastic,metal,plastic/metal) in functie de volumul acestora (0.12 mc., 0.24 mc., 1.10 mc.)

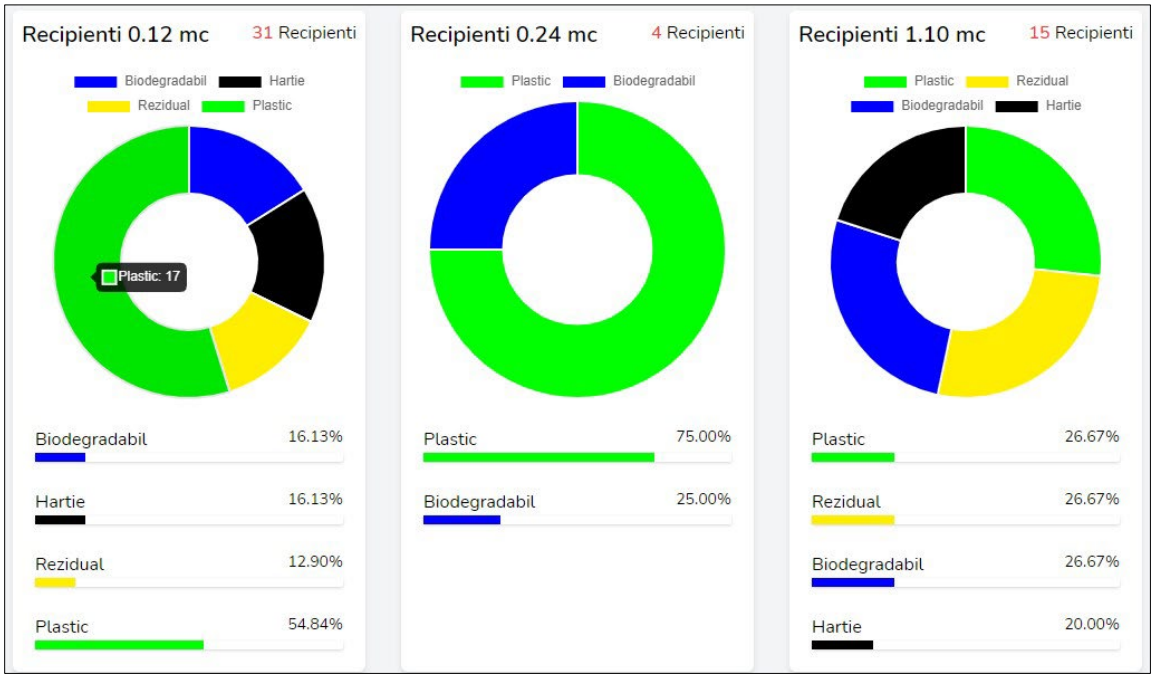


Figura 2 Statistici privind numărul recipientelor

Gestiunea sucursalelor

Se pot adauga sucursale noi ori edita pe cele deja create apasand pe iconita 

Figura 3 Adaugarea de noi sucursale

Gestiunea utilizatorilor

Se vor putea adauga utilizatori noi ori, apasand pe iconita se vor putea edita datele pentru utilizatorii existenti, precum: nume, e-mail, numar de telefon, sucursala asociata, zona asociata, rolul pe care persoana il detine si se pot trece observatii in dreptul fiecaruia daca sunt necesare.

Figura 4 Gestionarea utilizatorilor

In cazul in care se doreste eliminarea unui utilizator (din diverse motive), este necesar sa se ia legatura cu Administratorul pentru a gestiona aceasta modificare.

Gestiunea colectării deșeurilor

Se vor putea urmari datele de colectare ale deseurilor in functie de cod-ul tag-ului situat in interiorul pubelelor, client, adresa, numar auto, tip deseu, volum, cat si data si ora colectarii.

Apasand pe iconita putem vedea in detaliu colectarile per fiecare locatie.

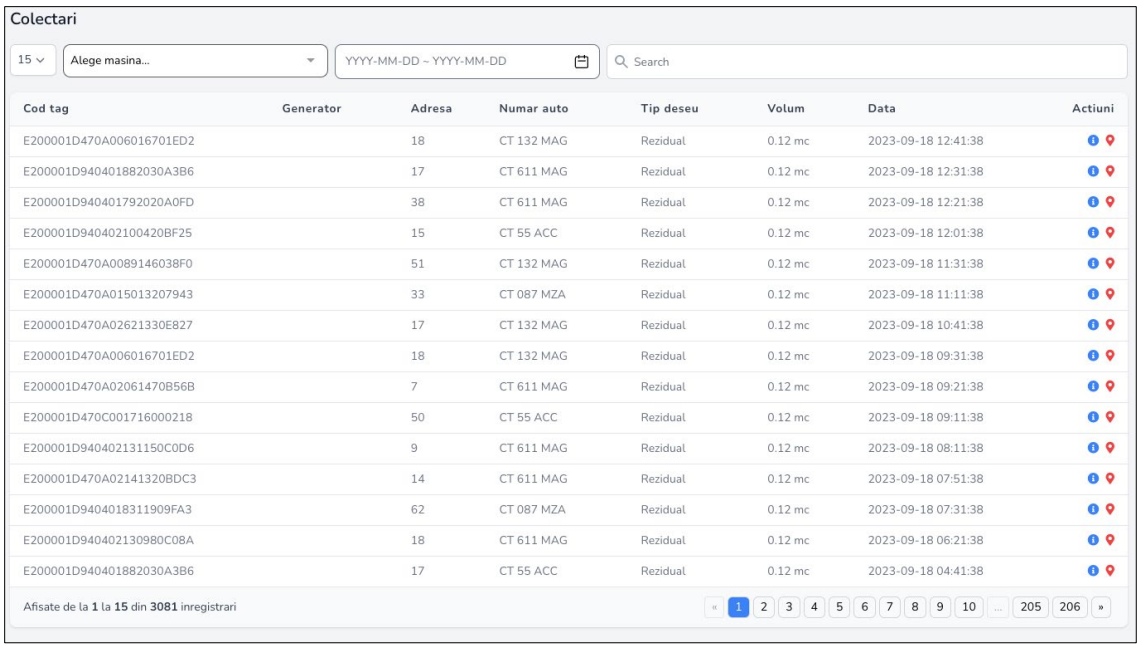


Figura 5 Colectarea deșeurilor

Apasand pe iconita  putem vedea in detaliu colectarile per fiecare locatie in functie de data, ora, numarul masinii ce a ridicat deseul, tipul deseului, numarul persoanelor si adresa.

Detalii colectari luna curenta TAG - E200001B19170134255065DB

Tip deseu	Volum	Numar auto	Data colectarii	Numar persoane	Adresa
Biodegradabil	1.10 mc		2021-11-09 17:45:32	2	
Biodegradabil	1.10 mc		2021-11-09 17:45:12	2	
Biodegradabil	1.10 mc		2021-11-09 17:43:42	2	
Biodegradabil	1.10 mc		2021-11-09 17:43:02	2	
Biodegradabil	1.10 mc		2021-11-09 17:41:39	2	
Biodegradabil	1.10 mc		2021-11-09 17:41:27	2	
Biodegradabil	1.10 mc		2021-11-09 11:18:08	2	
Biodegradabil	1.10 mc		2021-11-09 11:18:00	2	
Biodegradabil	1.10 mc		2021-11-09 10:48:06	2	
Biodegradabil	1.10 mc		2021-11-09 10:47:49	2	
Biodegradabil	1.10 mc		2021-11-09 10:47:42	2	
Biodegradabil	1.10 mc		2021-11-08 16:11:16	2	
Biodegradabil	1.10 mc		2021-11-08 16:09:26	2	
Biodegradabil	1.10 mc		2021-11-08 16:09:01	2	
Biodegradabil	1.10 mc		2021-11-08 15:36:03	2	
Total	16.50 mc				

Figura 6 Colectarea deșeurilor

Apasand pe iconita rosie  vom vizualiza localizarea pe harta in functie de codul tag-ului selectat.

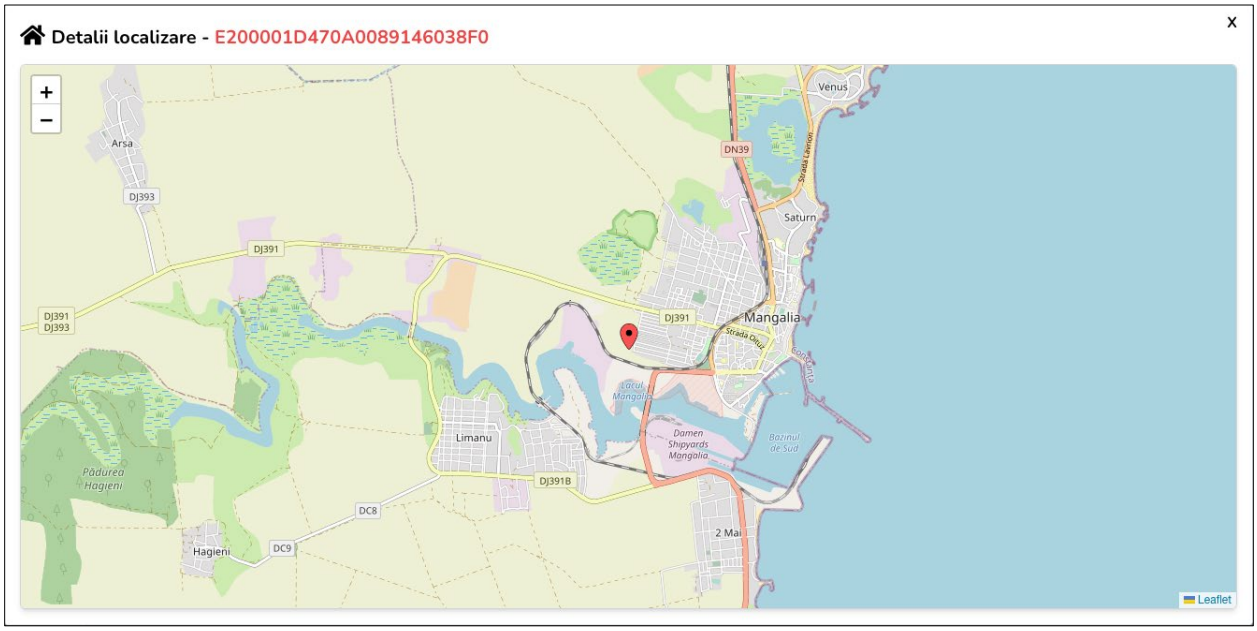
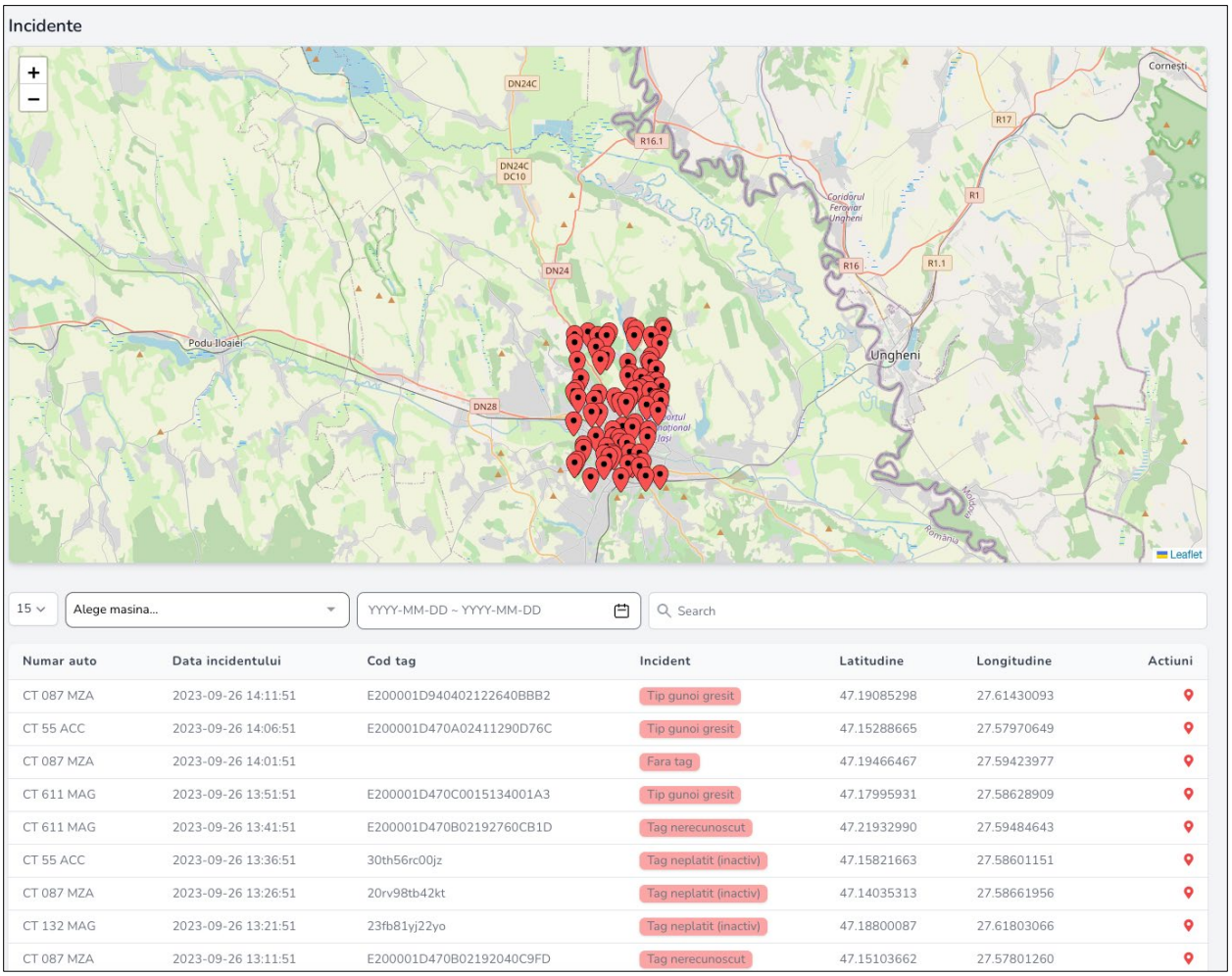


Figura 7 Localizarea recipientelor de colectare a deșeurilor

Gestiunea incidentelor

In cadrul acestui meniu vor fi sesizate incidentele cu privire la tag-urile existente pe pubele. Pot fi sesizari legate de lipsa unui tag pe o pubela sau legate de un tag care nu este inregistrat in baza de date.

Apasand pe iconita rosie ni se va deschide o alta harta cu indicarea locului precis al incidentului.



Numar auto	Data incidentului	Cod tag	Incident	Latitudine	Longitudine	Actiuni
CT 087 MZA	2023-09-26 14:11:51	E200001D940402122640BBB2	Tip gunoi gresit	47.19085298	27.61430093	
CT 55 ACC	2023-09-26 14:06:51	E200001D470A02411290D76C	Tip gunoi gresit	47.15288665	27.57970649	
CT 087 MZA	2023-09-26 14:01:51		Fara tag	47.19466467	27.59423977	
CT 611 MAG	2023-09-26 13:51:51	E200001D470C0015134001A3	Tip gunoi gresit	47.17995931	27.58628909	
CT 611 MAG	2023-09-26 13:41:51	E200001D470B02192760CB1D	Tag nerecunoscut	47.21932990	27.59484643	
CT 55 ACC	2023-09-26 13:36:51	30th56rc00jz	Tag neplatit (inactiv)	47.15821663	27.58601151	
CT 087 MZA	2023-09-26 13:26:51	20rv98tb42kt	Tag neplatit (inactiv)	47.14035313	27.58661956	
CT 132 MAG	2023-09-26 13:21:51	23fb81yj22yo	Tag neplatit (inactiv)	47.18800087	27.61803066	
CT 087 MZA	2023-09-26 13:11:51	E200001D470B02192040C9FD	Tag nerecunoscut	47.15103662	27.57801260	

Figura 8 Gestiunea incidentelor

Localizarea utilajelor de colectare a deșeurilor

Selectând numărul utilajului, in cadrul acestui meniu, utilizatorul are posibilitatea de a vizualiza locurile din care acesta a colectat deșeuri.

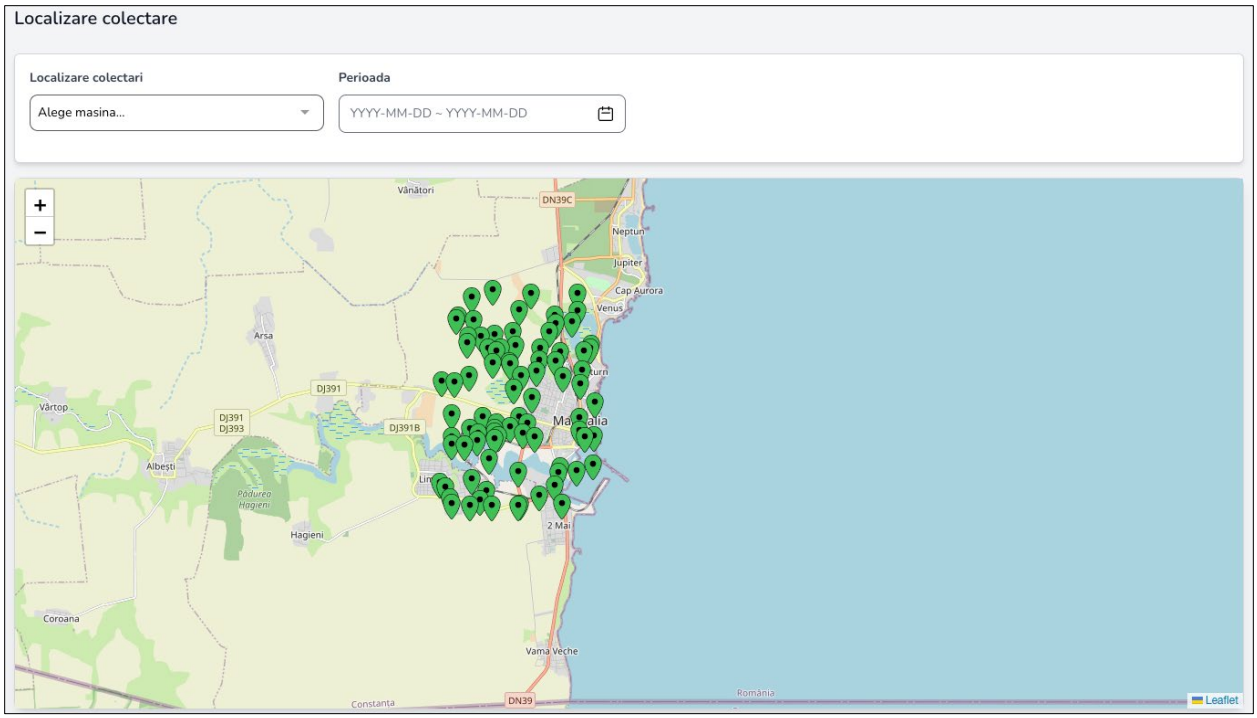


Figura 9 Vizualizarea locurilor de colectare a deșeurilor

Gestiunea generatorilor de deșeuri

Utilizatorul aplicatiei va putea sa adauge un nou generator de deșeuri (client), completând campurile obligatorii. Va fi necesar sa selecteze tipul de generator:

- asociatie de proprietari;
- institutie publica;
- persoana fizica;
- persoana juridica.

Ulterior se vor completa urmatoarele date: Denumire generator, Carte de identitate/Cod unic de inregistrare (in functie de tipul de generator), Numarul de telefon, Persoana de contact și Zona.

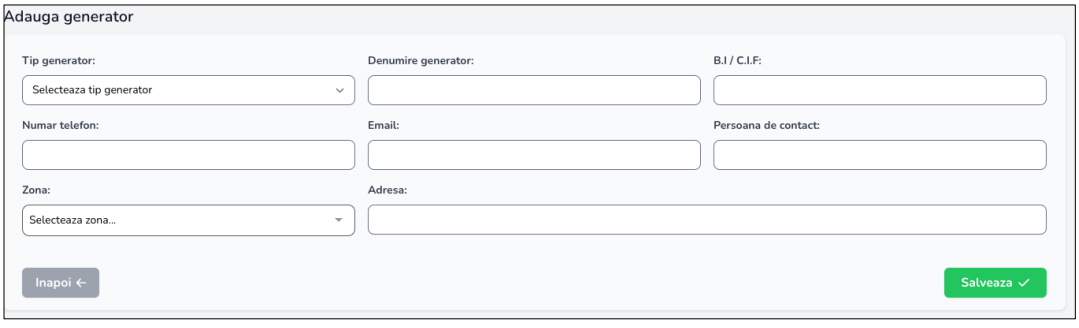


Figura 10 Adăugare generator de deșeuri

După efectuarea unui clic pe butonul “ Salveaza, acesta va fi adaugat automat in lista generatorilor de deșeuri.

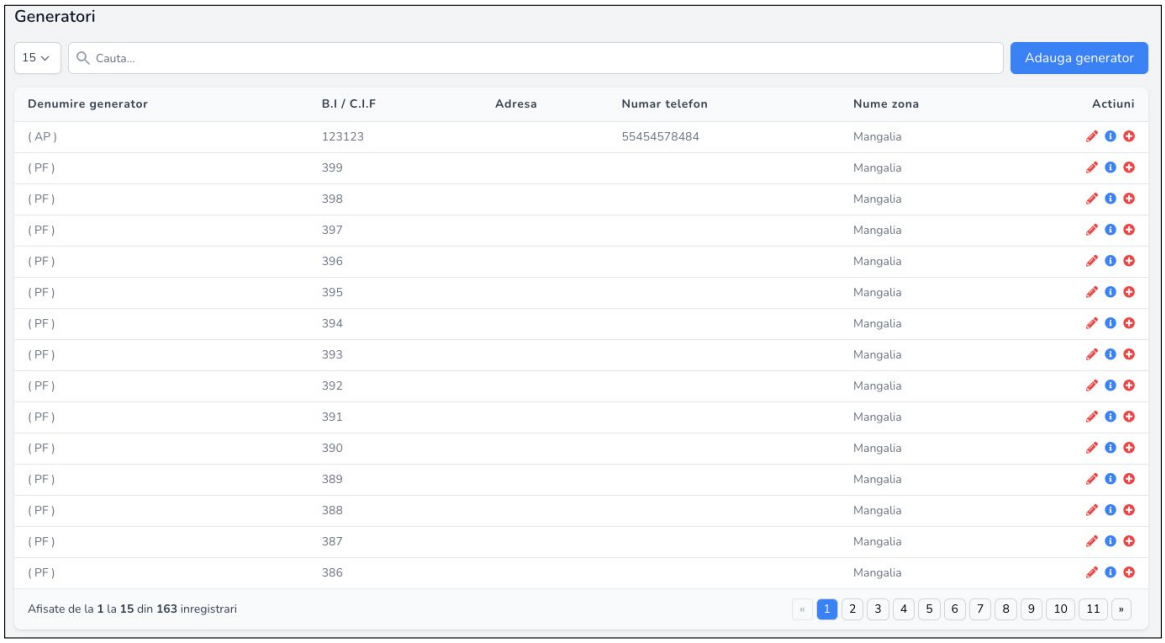


Figura 11 Lista generatorilor de deșeuri

Prin efectuarea unui clic pe se pot edita câmpurile enumerate mai sus.

Prin efectuarea unui clic pe se pot adăuga noi puncte de colectare pentru un generator de deșeuri (client).

Prin efectuarea unui clic pe se pot vizualiza toate puncte de colectare pentru un generator de deșeuri.

Gestiunea punctelor de colectare

Dupa aceleasi criterii enuntate anterior, observam lista completa cu punctele de colectare in functie de Tip client, Denumire client, Adresa, Numar de personae, Numar de telefon.

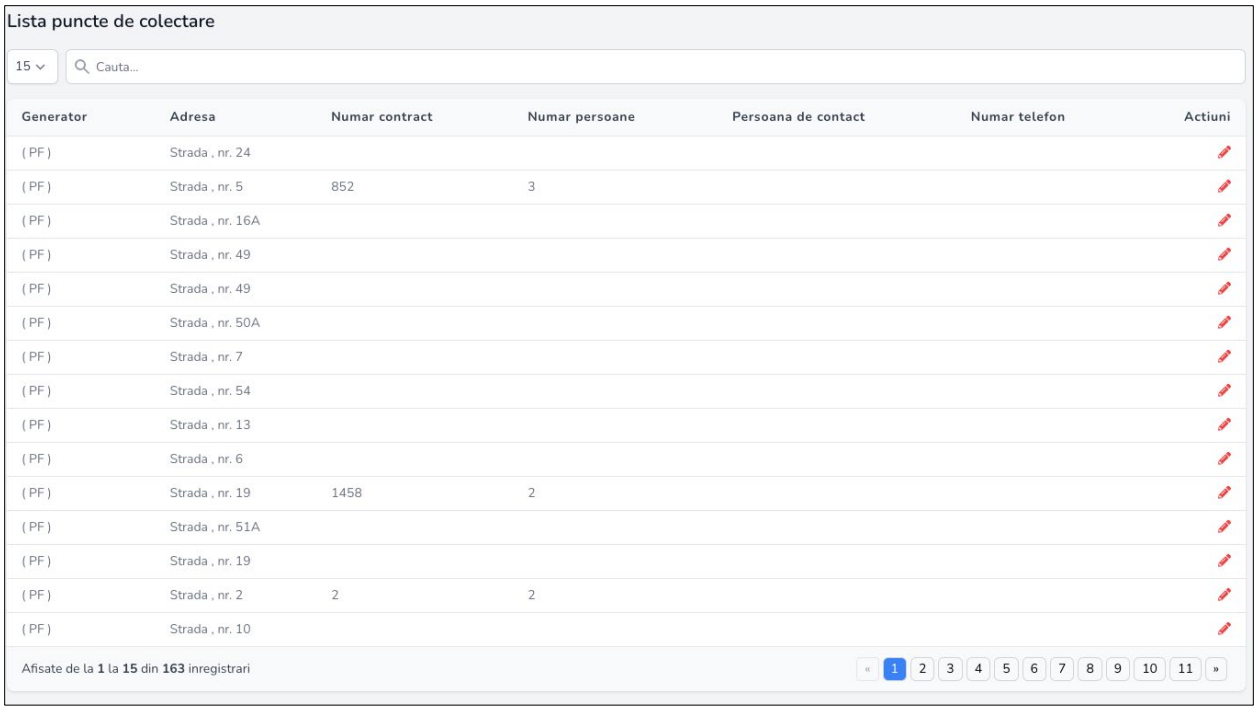


Figura 12 Lista punctelor de colectare

Prin efectuarea unui clic pe se pot actualiza datele punctului de colectare (schimbarea unor coordonate pentru ca adresa sa fie una cat mai precisa, numarul de persoane, etc.).

Gestiunea zonelor

In cadrul acestui meniu se poate adăuga o zona noua in baza de date ori se poate edita o zona existenta prin efectuarea unui clic pe .

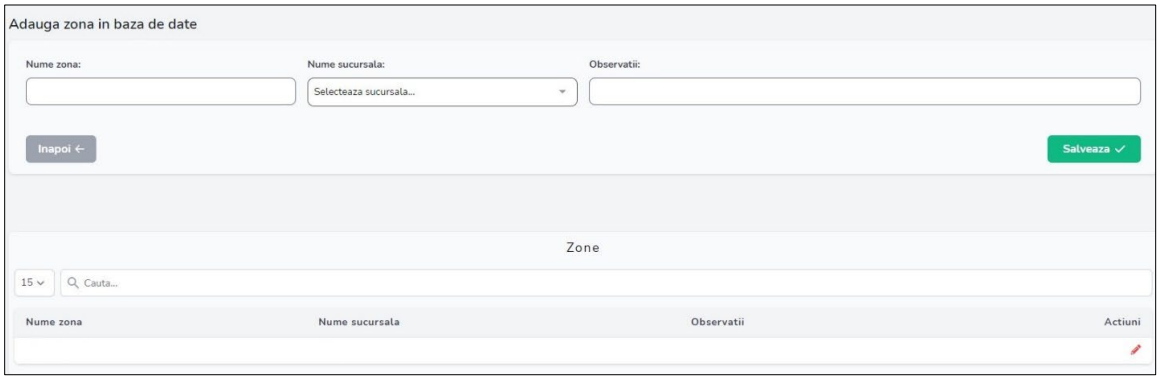


Figura 13 Adăugarea unei noi zone

Gestiunea cip-urilor

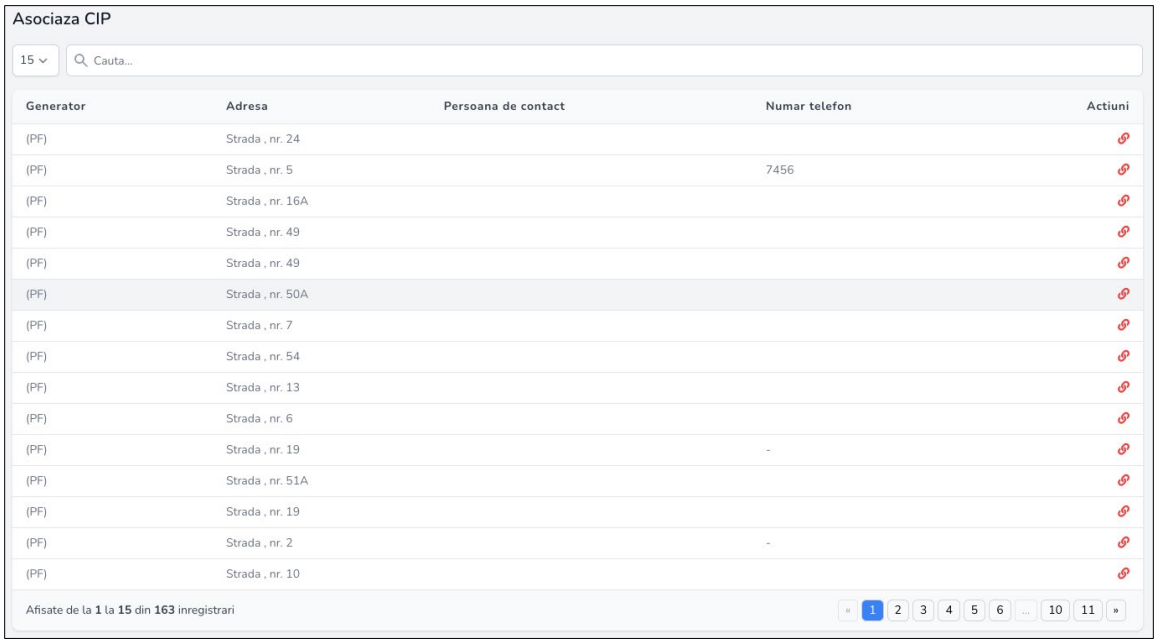


Figura 14 Gestinea cip-urilor

Cip-ul se poate asocia unui client prin efectuarea unui clic  și completarea codului acestuia în fereastra nou deschisă.

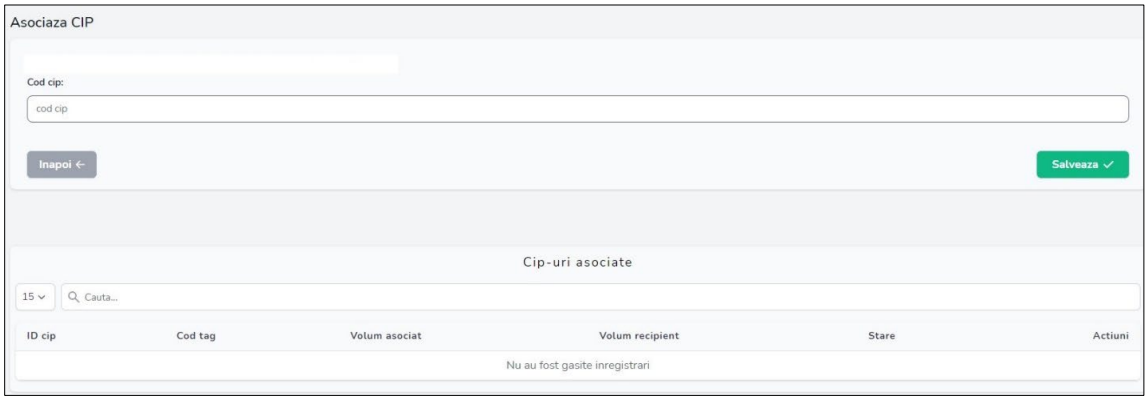


Figura 15 Asocierea cip-urilor cu generatorii de deșeuri

Gestiunea tag-urilor asociate recipientelor de colectare

Fiecare recipient pentru colectarea deșeurilor (pubelă) va avea inserat cate un TAG BIN, pe care, scanandu-l, ii vom afla atat codul unic, cat si un id pentru cip ce se va genera automat la scanare.

Pentru a adauga cip-uri, se vor completa urmatoarele campuri:

- Tip recipient (in functie de deseul propriu-zis: gunoi biodegradabil, hartie, plastic, rezidual,sticla, metal, metal/plastic);

- Volum recipient (0,12 ; 0,24 ; 1,10);
- Cod cip (care se genereaza automat la scanarea tag-ului);
- Status (activ/inactiv);
- „Observatii.

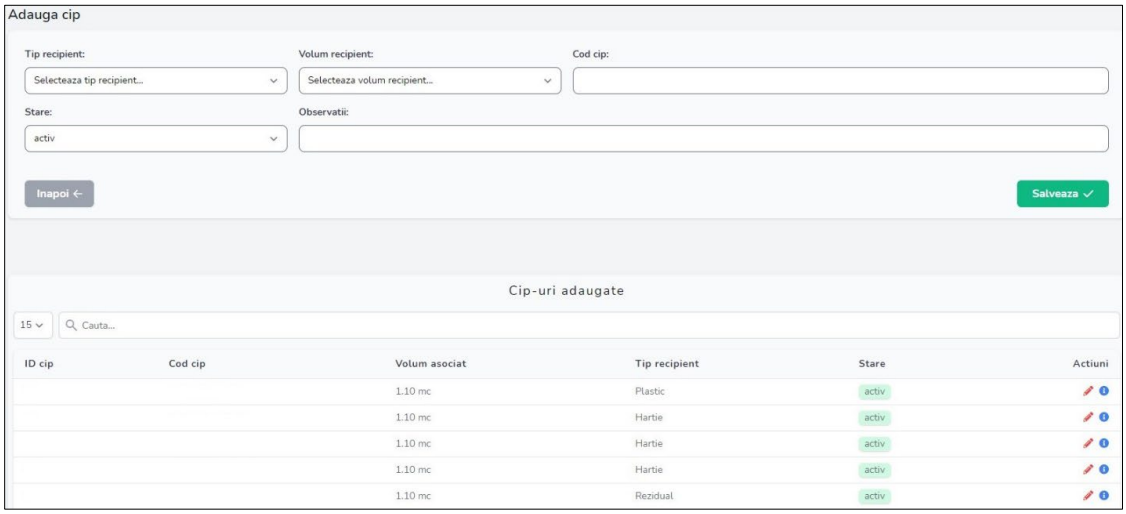


Figura 16 Asociere cip - recipient de colectare

Prin efectuarea unui clic pe se pot edita câmpurile enumerate anterior.

Prin efectuarea unui clic pe se pot vizualiza datele înregistrate.

Vizualizare recipienți

In cadrul acestui meniu se vor putea observa toți recipienții introduși in sistem cu datele aferente.

Figura 17 Recipienți

Totodata, pentru a determina recipientii asociati unei adrese, utilizatorul are la dispozitie functia de cautare a recipientilor in functie de adresa asociata acestora.

Gestiunea utilajelor de colectare a deșeurilor

Fiecărui utilaj de colectare a deșeurilor îi este atribuit un anumit tip de deseu (reciclabil/rezidual), in functie de tipul de pubela pe care o are de colectat si transportat (deșeu biodegradabil, hartie, plastic, rezidual si sticla).

Utilizatorul are optiunea de adauga un nou utilaj de colectare a deșeurilor in baza de date, completand campurile

- Numar auto
- Id auto
- Volum compactor
- Tip deseu
- Consum mediu
- Zona
- Observatii

De asemenea, se pot actualiza datele corespunzătoare utilajelor de colectare a deșeurilor introduse in sistem.

Pentru fiecare utilaj de colectare a deșeurilor se pot incarca documentele necesare, apasand butonul acesteia in baza de date, astfel:

- Numar document
- Data emitere
- Data expirarii
- Observatii

Este nevoie să fie introduse și următoarele date Volumul compactor (tone), Id auto , Numarul de inmatriculare, Consumul mediu de combustibil.

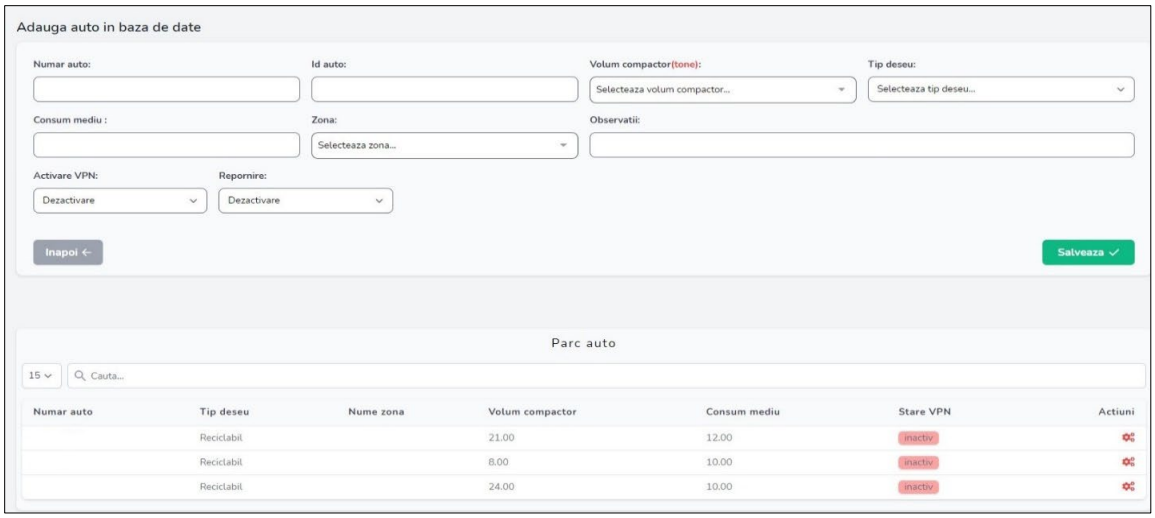


Figura 18 Gestionare utilaje de colectare a deșeurilor

Prin efectuarea unui clic pe butonul  se poate configura utilajul de colectare a deșeurilor, prin adăugarea documentelor sau actualizarea datelor existente.

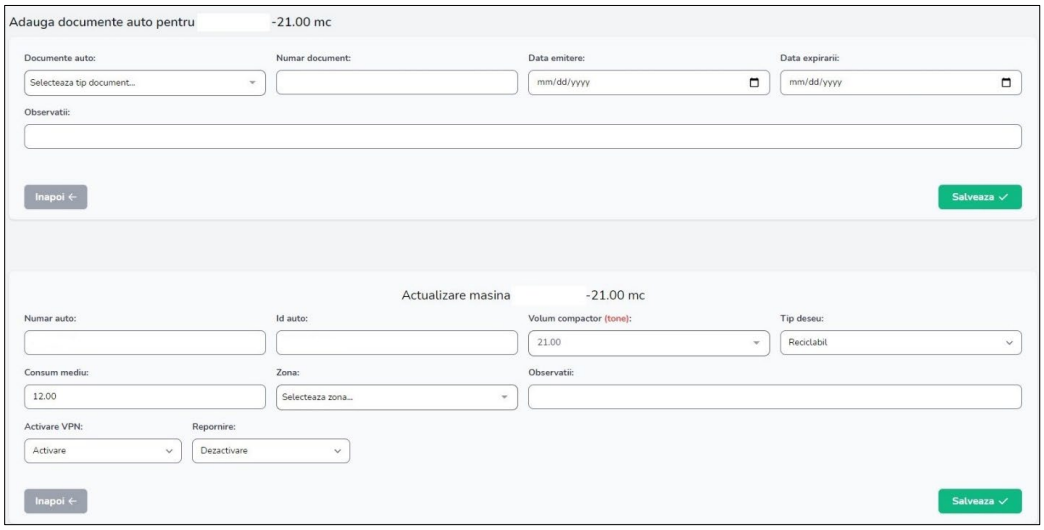


Figura 19 Configurare utilaj de colectare a deșeurilor

Gestiunea șoferilor

Se vor adauga numele, prenumele si utilajul de colectare a deșeurilor.

Figura 20 Gestiunea șoferilor

Gestiunea bonurilor carburant

Se vor completa detaliile consumului de carburant al masinilor, precum:

- Numar auto
- Volum compactor
- Consum
- Total carburant luna curenta.

Prin efectuarea unui clic pe se pot adăuga date justificative (documente) legate de consumul utilajului.

Figura 21 Consum de combustibil aferent utilajelor de colectare a deșeurilor

Gestiunea foilor de parcurs

Această secțiune permite gestionarea datelor referitoare la foile de parcurs: numele șoferului, data și ora la care a început activitatea utilajul de colectare a deșeurilor, , numărul de kilometri parcurși etc.

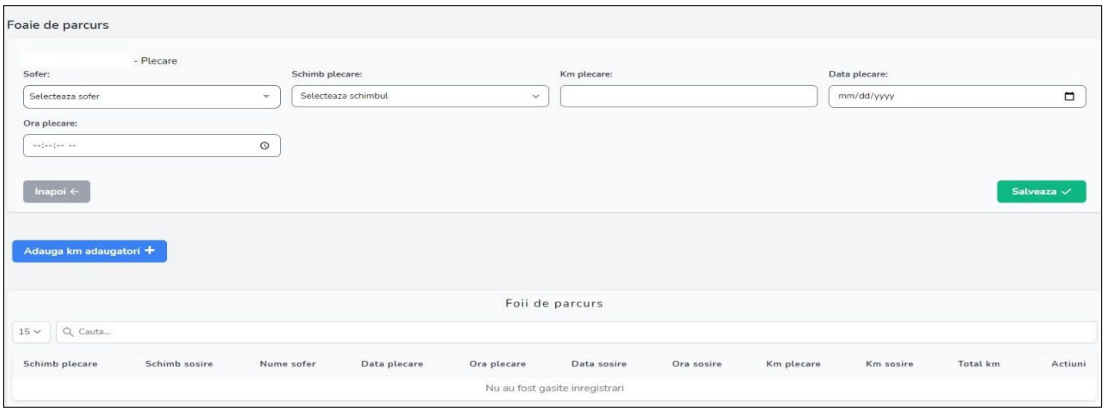


Figura 22 Gestiunea foilor de parcurs

Gestiunea tichetelor de cantar

Această secțiune permite gestionarea tichetelor de cântar prin completarea datelor referitoare la: utilajul de colectare a deșeurilor, destinația deșeurilor, tichetul de cântar, codul colectării, data și ora emiterii tichetului, greutatea deșeurilor etc.

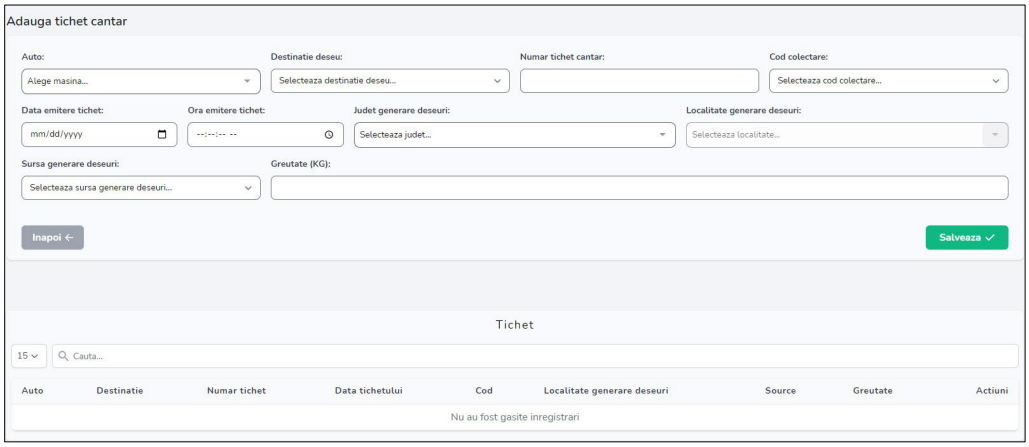


Figura 23 Gestiunea tichetelor de cântar

2 Tablouri de bord pentru analiza colectărilor și a depozitărilor la groapa de gunoi a deșeurilor

Pe baza datelor colectate din sistemele operaționale la nivel de UAT a fost construit un depozit de date care este sursa de date pentru tablourile de bord care permit analiza colectărilor, respectiv a depozitărilor la groapa de gunoi a deșeurilor la nivel de UAT.

În continuare sunt prezentate principalele tablouri de bord dezvoltate pentru analizele datelor colectate (Figura 24).

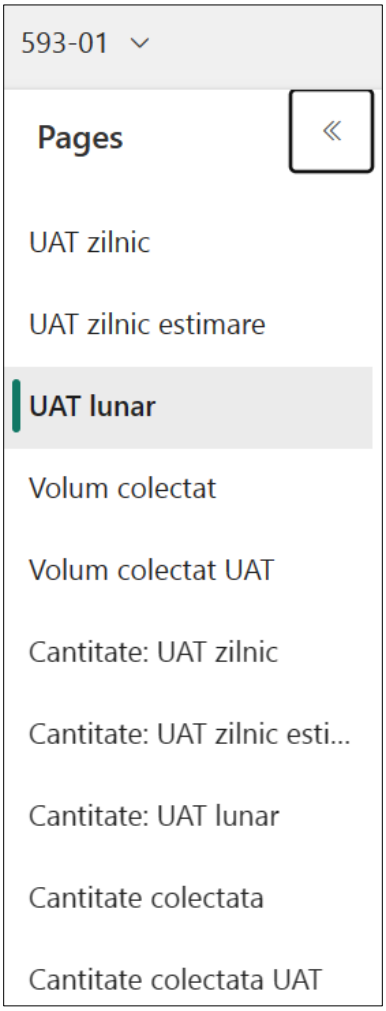


Figura 24 Tablouri de bord dezvoltate pentru analizele datelor colectate

În Figura 25 este prezentată situația colectărilor zilnice dintr-o lună la nivel de UAT, cu posibilitatea de filtrare pe tip de recipient și tip de utilaj.

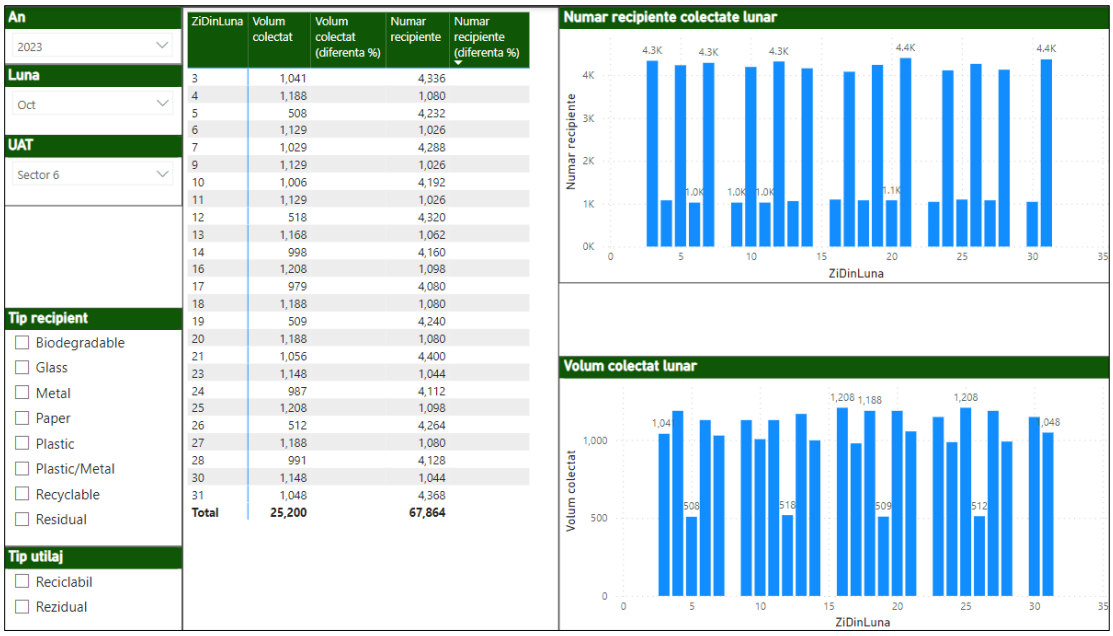


Figura 25 Situația colectărilor zilnice la nivel de UAT

În Figura 26 este prezentată estimarea colectărilor zilnice nivel de UAT, cu posibilitatea stabilirii intervalului de timp care va fi utilizat pentru previzionarea colectărilor și de filtrare pe tip de recipient și tip de utilaj.

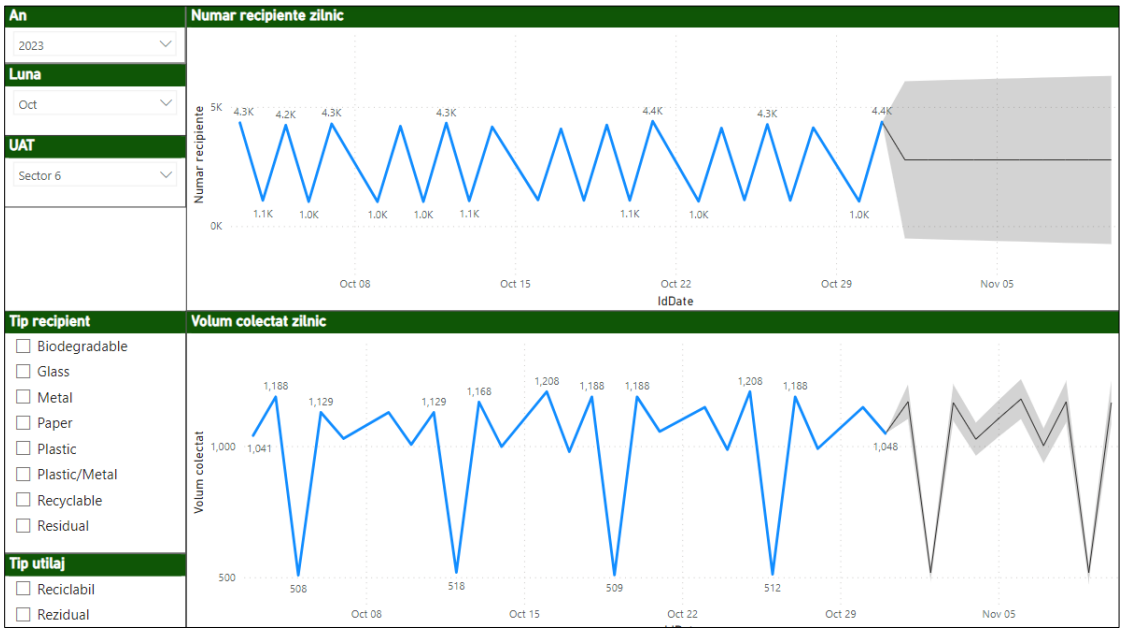


Figura 26 Estimarea colectărilor zilnice la nivel de UAT

În Figura 27 și Figura 28 sunt prezentate situațiile colectărilor lunare dintr-un an la nivel de UAT, cu posibilitatea de filtrare pe tip de recipient și tip de utilaj.

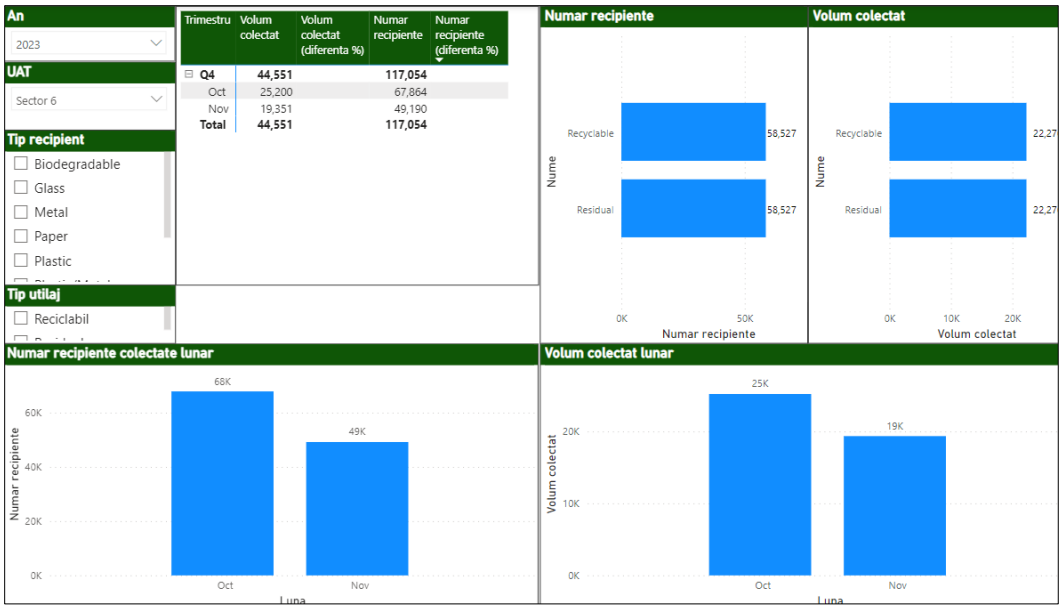


Figura 27 Situația colectărilor lunare la nivel de UAT

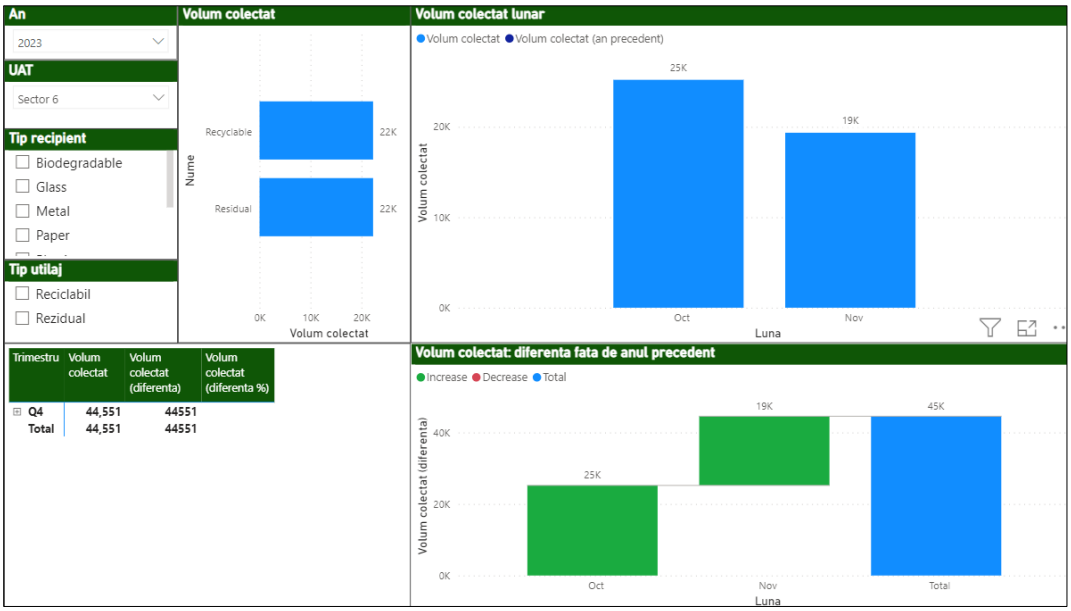


Figura 28 Situația colectărilor lunare la nivel de UAT

În Figura 29 este prezentată situația colectărilor anuale la nivel de UAT, cu posibilitatea de filtrare pe tip de recipient și tip de utilaj.

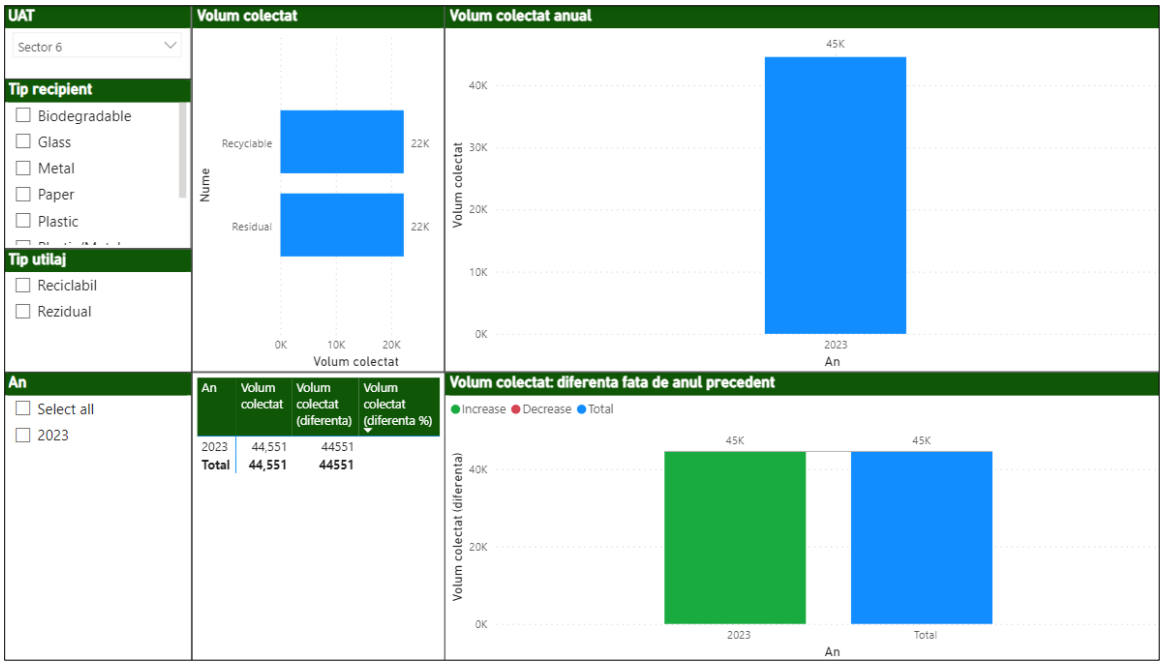


Figura 29 situația colectărilor anuale la nivel de UAT

În Figura 30 este prezentată situația depozitărilor zilnice dintr-o lună la groapa de gunoi la nivel de UAT, cu posibilitatea de filtrare pe tip de utilaj și destinație.

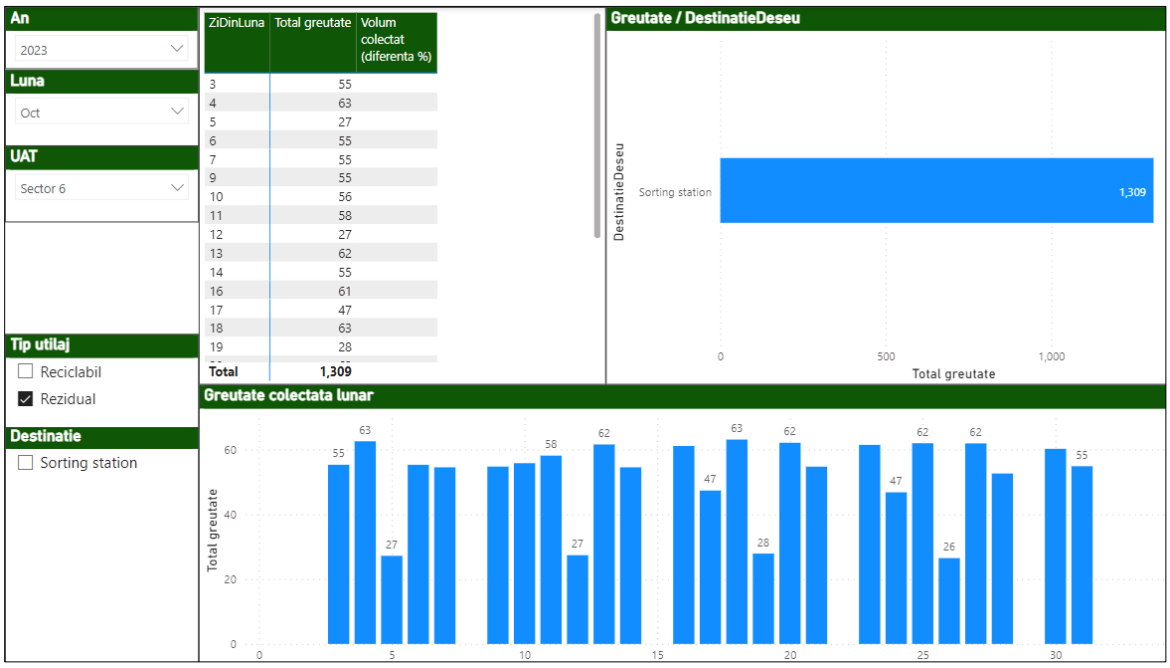


Figura 30 Situația depozitărilor zilnice la groapa de gunoi

În Figura 31 este prezentată estimarea depozitărilor zilnice la groapa de gunoi nivel de UAT, cu posibilitatea stabilirii intervalului de timp care va fi utilizat pentru previzionarea colectărilor și de filtrare pe tip de utilaj și destinație.

Figura 31 Estimarea depozitărilor zilnice la groapa de gunoi

În Figura 32 și Figura 33 sunt prezentate situațiile depozitărilor lunare la groapa de gunoi dintr-un an la nivel de UAT, cu posibilitatea de filtrare pe tip de utilaj și destinație.

Figura 32 Situația depozitărilor lunare la groapa de gunoi

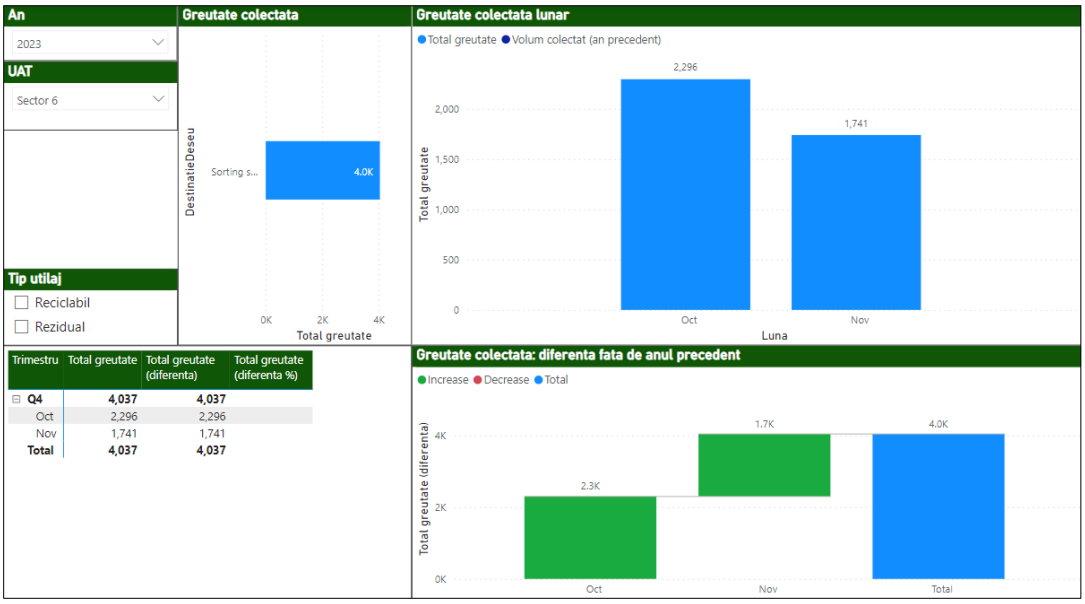


Figura 33 Situația depozitărilor lunare la groapa de gunoi

În Figura 34 este prezentată situația depozitărilor anuale la groapa de gunoi la nivel de UAT, cu posibilitatea de filtrare pe tip de utilaj și destinație.

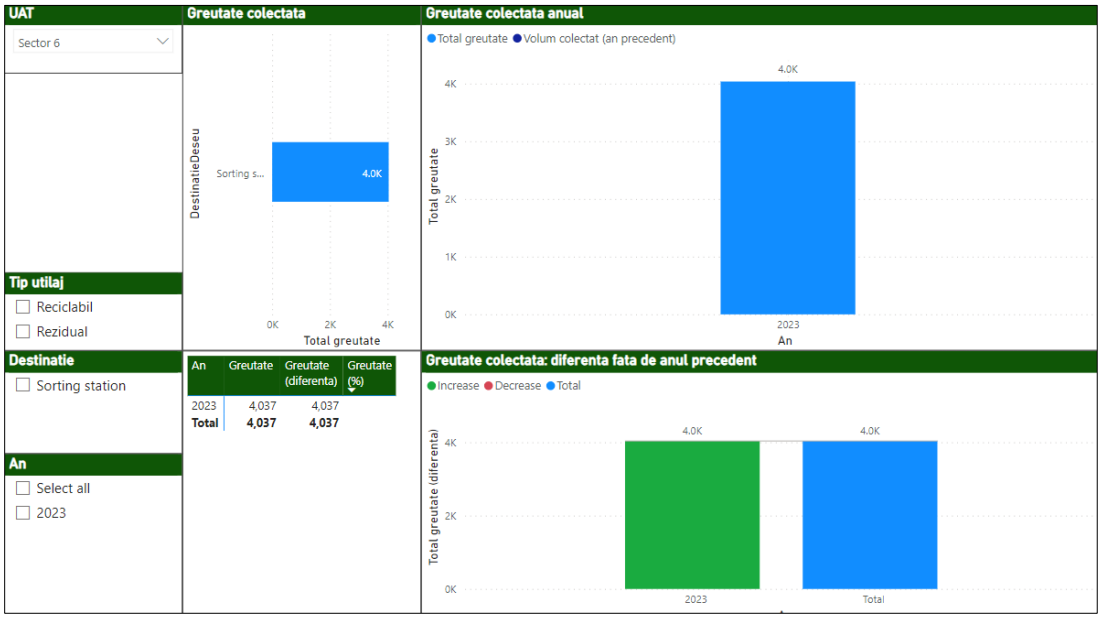


Figura 34 Situația depozitărilor anuale la groapa de gunoi

Pe lângă tablourile de bord prezentate pot fi dezvoltate o serie de noi tablouri de bord care să vina în sprijinul fundamentării politicilor la nivel național sau regional privind gestionarea deșeurilor.

3 Considerente privind performanța sistemului informatic pentru colectarea și analiza datelor prin programul PPCA

Sistemul informatic pentru colectarea și analiza datelor privind deșeurile prin programul PPCA trebuie să adopte și să faciliteze scalabilitatea, agilitatea și fiabilitatea. La această etapă este dificil de estimat un număr de utilizatori simultani care vor accesa componenta web a sistemului informatic. Scalarea sistemului trebuie să țină seama nu doar de creșterea numărului de accesări concurente ale diverselor componente ale sistemului informatic, ci și de posibilitatea unei interacțiuni mai intense cu aceasta.

Capacitatea sistemului informatic pentru colectarea datelor privind deșeurile prin programul PPCA de a permite ca scalarea pe verticală și pe orizontală să răspundă corect și la timp cererilor informaționale este o cerință obligatorie. Adoptarea principiilor de scalabilitate și toleranță la defect atât pentru componentele de back-end cât și pentru cele de front-end poate să aducă beneficii, în general, în ceea ce privește performanța și disponibilitatea ridicată (High Availability).

Întrucât sistemul are o arhitectură deschisă care poate fi dezvoltată mai departe pentru integrări suplimentare cu alte sisteme sau pentru dezvoltarea de noi componente și funcționalități suplimentare, scalabilitatea sistemului informatic trebuie abordată pe mai multe niveluri. Astfel, sistemul trebuie să adopte scalabilitatea atât pe verticală, cât și pe orizontală, urmărind:

1. scalabilitatea funcțională, acceptând adăugarea, în orice moment, de noi funcționalități fără să perturbe activitățile existente. Adăugarea de noi module software, înlocuirea sau modificarea funcțiilor acestora fără a afecta funcționarea operațională existentă la momentul actualizării software ar trebui să fie o prioritate absolută în proiectarea acestor noi module. Indiferent dacă noile componente vor fi dezvoltate ca module în cadrul sistemelor informatice existente sau ca noi aplicații software interconectate, ar trebui să se adopte principiul modularizării software cuplate în mod flexibil.
2. Scalabilitatea resurselor, având în vedere, în subsidiar, și scalabilitatea sarcinii, sistemul trebuind să dețină capacitatea să crească performanța și să scadă costurile ca răspuns la schimbările provenind din cererile de procesări la nivel de aplicații sau la nivel de sistem de operare, să se adapteze dinamic la sarcini mai mici sau mai mari, în funcție de activitatea operațională, fără a ridica probleme de performanță.

3.1 Instalarea sistemului informatic pentru colectarea și analiza datelor prin programul PPCA

3.1.1 Opțiuni pentru instalarea și distribuirea platformei PPCA

Se pot aborda trei opțiuni tehnice:

1 Instalarea platformei utilizând infrastructura hardware și software a Ministerului Mediului (opțiunea ”on-premise”). Întreaga activitate desfășurată prin această platformă va fi gestionată intern. Pentru această opțiune, întregul proces privind instalarea, configurarea, personalizarea, inițializarea sistemelor, utilizarea și întreținerea platformei va fi gestionat de specialiștii IT ai

Ministerului Mediului sau ar putea fi externalizat prin contractarea de servicii specializate de la unul sau mai mulți furnizori.

2 Solicitarea de resurse hardware de la STS și găzduirea acestei platforme în centrul de date al STS. STS poate să ofere o infrastructură securizată, care ar îndeplini nevoile hardware și software pe care le-ar presupune o astfel de platformă în ceea ce privește conținutul și accesul utilizatorilor. Între Ministerul Mediului și Serviciul de Telecomunicații Speciale (STS) există deja parteneriate în derulare[1]. Soluția găzduirii complete sau parțiale a componentelor software ale sistemului privind colectarea și analiza datelor prin programul PPCA ar aduce beneficii reale în termeni de reducere de costuri asociate operaționalizării sistemului, dar și de reducere a riscurilor legate de securitate.

3 Găzduirea platformei în cadrul unei infrastructuri de tip cloud: Infrastructure-as-a-Service(SaaS) sau Platform-as-a-Service (PaaS). Atât IaaS, cât și PaaS, permit platformei de colectare a informațiilor privind deșeurile din programul PPCA să ofere cele mai bune opțiuni de livrare de conținut, prin instalarea și configurarea acesteia într-un mediu de găzduire în cloud. Acest lucru va scădea semnificativ complexitatea întreținerii, profitând de Amazon Web Services, Google Cloud, Microsoft Azure sau alți furnizori de soluții de găzduire cloud, pentru a elimina necesitatea de a achiziționa servere dedicate, care pot fi costisitoare. Găzduirea în cloud este o soluție web care folosește servere diferite pentru a echilibra încărcarea resurselor și a maximiza performanța și timpul de disponibilitate pentru o anumită platformă sau un anumit site web. În această privință, sistemul informatic pentru programul PPCA poate beneficia de un „cluster" de servere care folosește resurse dintr-un sistem centralizat. Găzduirea în cloud constă din mai multe servere virtualizate, care sunt sincronizate, permițând astfel o echilibrare de resurse. Echilibrarea resurselor și intermedierea de servicii sunt mecanisme-cheie care minimizează timpul de răspuns al platformei la cererile în creștere ale utilizatorilor finali. De asemenea, echipa responsabilă pentru gestionarea operaționalizării platformei va primi acces la un panou de comandă care permite mutarea resurselor pe un alt server și să continue activitatea în caz de eroare.

În ceea ce privește performanța, pentru găzduirea on-premise, aceasta va depinde de resursele hardware disponibile, de tehnologia higher-end și de reducerea timpilor de încărcare. Găzduirea în cloud se bazează pe locația furnizorului, iar deținerea unor astfel de centre de date locale în apropiere se transpune, în cele mai multe cazuri, într-o performanță mai bună. În general, găzduirea on-premise poate fi considerată ca fiind mai rapidă decât găzduirea în cloud, datorită localizării sale mai aproape de utilizatorul final.

În ceea ce privește scalabilitatea, găzduirea în cloud oferă posibilitatea de a aloca resurse suplimentare în funcție de necesități. Deținătorul platformei va decide câte resurse sunt necesare și va plăti doar pentru ceea ce se va utiliza.

[1] Proiectul SIPOCA 596 „Dezvoltarea capacității administrative a Ministerului Mediului privind gestionarea situațiilor de urgență generate de riscurile specifice ministerului și a situațiilor privind starea mediului" a fost derulat de către Ministerul Mediului în parteneriat cu Serviciul de Telecomunicații Speciale.

Din punctul de vedere al managementului serverului, serviciile de întreținere pentru găzduire web sunt externalizate furnizorului de servicii cloud. Costurile finale vor depinde de furnizor, iar accesul la servicii de suport 24/7 va crește costul soluției de găzduire. În cazul unei găzduiri dedicate (on-premise), problemele de funcționalitate, întreținere, planificarea backup-urilor, planificarea recuperării în urma dezastrelor trebuie să fie sarcini ale personalului intern IT al Ministerului Mediului sau, de asemenea, aceste activități se pot externaliza.

Deși găzduirea într-un cloud public ar putea poate fi privită cu reticență din cauza inconvenientului de a nu putea asigura pe deplin securitatea datelor colectate de administrația publică (D. Sampson and M. M. Chowdhury, 2021), există un set de beneficii privind această soluție care o fac atractivă în decizia de a o alege ca opțiune de găzduire. Furnizorii de servicii de cloud folosesc un model de tip „plătește la plecare" (pay-as-you-go), asimilat unui abonament, sau un model de tip „plătește pentru resurse", care facturează serviciile de găzduire ale fiecărui client. Găzduirea în cloud asigură o scalabilitate automată (în sus sau în jos, pe orizontală sau pe verticală), în funcție de nevoile în timp real cu privire la resursele hardware și software necesare și la traficul înregistrat. Conform datacentermap.com, există 48 de centre de date în colocație[2] și 9 furnizori de găzduire în cloud în România, cu o gamă completă de servicii de cloud (infrastructura ca serviciu - IaaS). De asemenea, nu trebuie trecută cu vederea urmărirea oportunității ca, în viitor, sistemul informatic pentru programul PPCA să fie găzduit în cloud-ul guvernamental, planificat a fi implementat conform propunerilor din Planul național de redresare și reziliență (PNRR).[3]

O altă opțiune posibilă privită ca alternativă la reticența în utilizarea de servicii de cloud public ar implica stocarea de date sensibile într-un mediu sigur și nu expunerea acestora în mediu public, care este mai dificil de controlat din punct de vedere al securității, în paralel cu utilizarea unei soluții de găzduire în cloud pentru datele și serviciile mai puțin senzitive. Această variantă de găzduire poartă denumirea de cloud hibrid. Soluțiile de tip cloud hibrid sunt din ce în ce mai frecvente, fiind adoptate cu precădere mai ales în sectorul privat. Varianta cloud-ului hibrid poate aduce ca beneficiu utilizarea puterii de calcul a unui cloud public în tandem cu soluția găzduirii interne (on-premise) a sistemului pentru colectarea și analiza datelor prin programul PPCA. În consecință, activitățile cele mai sensibile din punctul de vedere al datelor vehiculate se pot desfășura într-un centru de date local (privat), iar pentru găzduirea resurselor mai puțin critice se poate folosi un cloud public, de la un furnizor agreat de servicii (AWS, Google, Microsoft, etc.). Alternativ, se poate opta pentru un alt model de design de cloud hibrid, și anume acela potrivit căruia datele să fie stocate local și să folosească cloud-ul public doar pentru procesarea acestora. În acest concept, datele sunt găzduite "on-premise", în centre de date cu costuri reduse, iar sarcina semnificativă de prelucrare de date și servicii este transferată în cloud (Hashim, Farooq, & Piatti-Fünfkirchen, 2020). Aceasta este opțiunea în care datele sunt trimise înapoi la centrele de date după prelucrarea activităților în cloud-ul public.

Deși sistemul informatic dezvoltat ca rezultat al prezentului proiect permite găzduirea în cloud, considerăm că decizia de a opta pentru această soluție de distribuire a diferitelor componente software care intră în arhitectura acestuia aparține beneficiarului. De aceea, în continuare vom prezenta etapele

[2] https://www.datacentermap.com/romania/

[3] https://mfe.gov.ro/pnrr/

instalării și punerii în funcțiune a sistemului informatic potrivit opțiunii de găzduire după modelul “on-premise”.

4 Etape în instalarea și punerea în funcțiune a platformei PPCA

Arhitectura funcțională a sistemului sistemului informatic pentru colectarea și analiza datelor prin programul PPCA a for prezentată pe larg în livrabilul *Platformă funcțională de colectare a datelor privind PPCA*. Componentele majore ale acestei soluții tehnice sunt sistemul operațional de colectare a datelor privind deșeurile, componenta de integrare și componenta de analiză a datelor și de raportare prin instrumentul de tip talou de bord. Fiecare dintre aceste componente deține caracteristici tehnice diferite. Întrucât sistemul operațional de colectare a datelor primare este în apanajul UAT-urilor, în cadrul prezentului raport sunt enunțate principalele etape necesare operaționalizării sistemului informatic de analiză a datelor, respectiv componenta de integrare și componenta de analiză și vizualizare prin tablou de bord.

4.1 Instalarea și pregătirea serverelor de baze de date

Structurile de date de tip DataLake și DataWarehouse care deservesc soluția informatică PPCA au fost dezvoltate utilizând sistemul de gestiune a bazelor de date Microsoft SQL Server. Pentru construirea acestora s-a utilizat de către echipa de dezvoltare SGBDul SQL Server Express Edition, o versiune care poate fi utilizată cu titlu gratuit, însă prezintă o serie de limitări care nu o fac eligibilă pentru mediul de producție. În acest sens, se recomandă instalarea la sediul beneficiarului a suitei de aplicații Microsoft SQL Server, alegând fie versiunea Standard, fie Enterprise Edition[4].

- Enterprise Edition este o alegere bună pentru aplicațiile pentru care performanța în memorie este esențială, asigurând cel mai înalt nivel de securitate și disponibilitate ridicată (High Availability);
- Standard Edition oferă baze de date cu caracteristici complete, fiind calibrată pentru aplicații de nivel mediu și structuri de date de tip magazii de date (*data marts*).

Ambele versiuni prezintă politici de licențiere diferite[5]. În vederea unei decizii informate cu privire la costurile de licențiere pe care beneficiarul ar trebui să le suporte pentru a migra platforma PPCA în mediul de producție prezentăm cele două modele principale de licențiere practicate de Microsoft pentru utilizarea produselor aferente versiunilor SQL Server 2019 și SQL Server 2022:

- **Licențierea per nucleu (*per core*)**: Oferă clienților o măsură mai precisă a puterii de calcul și o măsură de licențiere mai consistentă, indiferent dacă soluțiile sunt implementate "*on-premise*" pe servere fizice, în medii virtuale sau cloud. Licențierea per-nucleu este adecvată atunci când clienții nu pot număra utilizatorii/dispozitivele, baza de date trebuie să permită o sarcină de lucru pentru Internet/Extranet, sau sarcini de lucru pentru sisteme care se integrează cu sisteme din

[4] O prezentare a diferențelor care există între diversele versiuni ale sistemelor de gestiune a bazelor de date utilizate în dezvoltarea platformei PPCA a făcut subiectul livrabilului *Platformă funcțională de colectare a datelor privind PPCA*.

[5] O descriere mai detaliată a versiunilor și modelelor de licențiere specifice Microsoft SQL Server poate fi vizualizată în https://download.microsoft.com/download/f/0/d/f0d7004e-9e39-4991-853b-2aa09e4ce456/SQL%20Server%202019%20%20Licensing%20Datasheet.pdf

exterior. Pentru a licenția un server fizic toate nucleele fizice ale serverului trebuie să fie licenţiate. Microsoft impune o limită de minim 4 licențe per nucleu pentru fiecare procesor fizic al serverului de bază de date.

- **Licențierea per server și de tip CAL (*Client Access Licence*)**: Oferă opțiunea de a licenția utilizatori și/sau dispozitive, cu acces la costuri reduse pentru implementări incrementate privind bazele de date SQL Server. Fiecare server care rulează software SQL Server necesită o licență de tip server. Fiecare utilizator și/sau dispozitiv care accesează un server SQL licențiat necesită o licență de tip CAL SQL Server pentru aceeași versiune. Există, prin urmare, două tipuri principale de licențe CAL Microsoft SQL Server:
  - Licențierea CAL pentru utilizator (user CAL): Câte o licență necesară pentru fiecare utilizator, permițând acestuia să acceseze SQL Server. Sunt recomandate dacă sistemul va fi accesat de mai multe dispozitive și mai puțini utilizatori.
  - Licențierea CAL pentru dispozitiv (device CAL): câte o licență necesară pentru fiecare dispozitiv (de exemplu, un client care accesează serverul). Acestea permit tuturor utilizatorilor care utilizează dispozitivul de tip client să acceseze SQL Server.

Se recomandă ca serverul de baze de date să fie instalat pe o mașină independentă, cu următoarele caracteristici tehnice minimale:

A. Specificatii hardware minimale:

- Procesor x64 tip Intel Xeon cu suport Intel EM64T, minim 4 nuclee (cores) 4 threads, minim 3,3 GHz;
- SSD minim 700Gb cu doua partitii: 200Gb (pentru sistemul de operare) si 500 Gb (pentru date)
- RAM 32 GB;
- Retea 100/1000 NIC

B. Specificatii software:

- Windows Server 2022 Standard Edition
- SQL Server Developer Edition 2019 sau SQL Server Enterprise 2019 (Database Engine, Analysis Services, Integration Services, Reporting Services) și SQL Agent instalat.
- .NET Framework 4.8
- Network software
- Access prin Remote Desktop Connection - 2 conexiuni simultane cu drept de administrator
- SQL Server Management Studio si Azure Data Studio
- Mail Server sau utilizarea unui alt server de mail pentru trimiterea de mesaje prin aplicație sau prin SQL Server.
- IIS 10 instalat.
- Acces prin VPN securizat.

În funcție de politicile de securitate privind infrastructura la sediul beneficiarului, administratorul de sistem poate decide reguli specifice privind modul de protejare a serverului de baze de date. Se recomandă separarea nivelului arhitectural de baze de date de nivelul arhitectural al aplicației, ceea ce

presupune existența a două servere diferite, în cadrul cărora cele două niveluri vor funcționa autonom, dar interconectate între ele.

4.2 Inițializarea bazelor de date

Sistemul informatic pentru colectarea și analiza datelor prin programul PPCA necesită două baze de date având roluri diferite: una care joacă rol de DataLake cu rol de colectare a datelor provenind de la sistemul informatic operațional, și cea care joacă rol de depozit de date (DataWarehouse) care va da sursa informațională pentru analiza multidimensională pe baza datelor colectate.

Pentru ca sistemul să fie funcțional este necesară definirea structurilor aferente acestor baze de date înainte de instalarea serviciului web și a componentei de tip BI pentru analiza datelor. Două abordări pot dicta modul de inițializare a bazelor de date, spre a fi realizată trecerea de la mediul de dezvoltare la mediul de producție:

1. Inițializarea bazelor de date fără preluarea datelor culese în perioada de testare. În acest caz generarea structurilor de date specifice celor două baze de date, a procedurilor stocate și a funcțiilor definite de utilizatori se poate realiza pe baza a câte unui fișier de tip script. Ulterior instanța SQL Server instalată pe serverul de baze de date de producție la sediul beneficiarului va putea genera bazele de date prin simpla executare a acestor scripturi.
2. Inițializarea bazelor de date cu preluarea datelor culese în perioada de testare. Pentru acest caz, se recomandă în prealabil, o operațiune de curățare a datelor de test, prin păstrarea celor care trebuie să fie reținute pe serverul de producție, după punerea în funcțiune a platformei PPCA. Deși pot fi apelate pentru aceasta mai multe metode, cea mai sigură dintre acestea ar fi generarea unei copii de siguranță pentru fiecare bază de date, urmată de restaurarea fiecărei copii de siguranță ca bază de date nouă pe serverul de producție.

Structurile de date aferente bazelor de date necesare pentru componentele de colectare și analiză a datelor prin programul PPCA pot fi vizualizate în cadrul scripturilor generate și expuse în Anexa 1.

4.3 Instalarea și pregătirea serverului web

Cu privire la serverul web necesar componentei de integrare a sistemelor, se recomandă existența unui server de tip Windows 2022, fie fizic, fie virtual, care să dețină IP fizic accesibil din exterior și preferabil asociat unui domeniu de internet sau subdomeniu al domeniului beneficiarului.

Cu privire la serverul care va deservi componenta de analiză și tablou de bord, deși acesta poate fi instalat pe aceași mașină (fizică sau virtuală) pe care se va instala serverul de aplicație care va găzdui serviciul web de interconectare cu sistemul operațional pentru programul PPCA, se recomandă separarea acestuia de nivelul de integrare, prin plasarea sa într-o zonă de tip DMZ și nu în zona de intranet.

A. Specificatii hardware minimale:

- Procesor x64 tip Intel Xeon, minim 2,4 GHz, 4 nuclee;
- SSD minim 500 GB cu doua partitii: 200Gb (pentru sistemul de operare) si 300 Gb (pentru date)
- RAM 16 GB;

- Retea 100/1000 NIC

B. Specificatii software:

- OS: Windows Server 2022 Standard Edition
- IIS 10 (daca este posibil, cu instalarea unui certificat SSL);
- Adresa IP dedicata accesibila din exterior;
- .NET Framework 4.8 si Visual Studio 2022 instalat (cu posibilitatea adaugarii de noi componente ulterior).
- Access prin Remote Desktop Connection - 2 conexiuni simultane cu drept de administrator
- Acces direct la serverul SQL de mai sus prin aplicații web sau desktop
- Acces prin VPN securizat.

4.4 Migrarea sistemului informatic din mediul de dezvoltare pe serverele de producție

Pentru instalarea și punerea în funcțiune a sistemului informatic pentru colectarea și analiza datelor prin programul PPCA pregătirea serverelor presupune, în esență, încercarea de replicare a mediului de dezvoltare în care sistemul a fost realizat. Însă migrarea sistemului informatic presupune inclusiv instalarea componentelor software necesare găzduirii "on-premise", incluzînd întreaga suită de configurări specifice acestora, pentru a asigura compatibilitatea cu mediul de dezvoltare.

4.4.1 Instalarea serverului web IIS

Un server web poate furniza informații utilizatorilor în mai multe forme, cum ar fi prin intermediul paginilor web statice codificate prin limbajul HTML, prin schimburi de fișiere ca descărcări (downloads) și încărcări (uploads) sau documente text, fișiere multimedia, etc. Serverul web IIS este o componentă a sistemului de operare Windows ce permite găzduirea de situri sau aplicații web scrise în limbaje de scripting precum ASP.NET, MVC, WebApi, etc., acceptând solicitări de la dispozitivele client la distanță (*HTTP requests*) și returnând răspunsul corespunzător (*HTTP Response*). Principala funcționalitate permite serverului web IIS să partajeze și să furnizeze informații prin rețelele locale (LAN), cum ar fi intranet-urile corporative și rețelele de zonă largă (WAN), cum ar fi Internetul.

Serviciul web prin care s-a realizat interconectarea sistemelor operațional și de analiză a datelor aferente programului PPCA a fost dezvoltat sub forma unei aplicații web (web application) utilizând tehnologia Windows Communication Foundation (WCF) de la Microsoft. În vederea asigurării compatibilității, serverul Microsoft IIS este capabil să găzduiască nativ un astfel de serviciu și gestioneze corect cererile provenite de la clienți, apoi să emită răspunsurile către aceștia.

Etapele principale privind instalarea componentei Windows IIS vor fi prezentate în continuare:

Pasul 1: Se va specifica numele serverului web care va găzdui serviciul web.

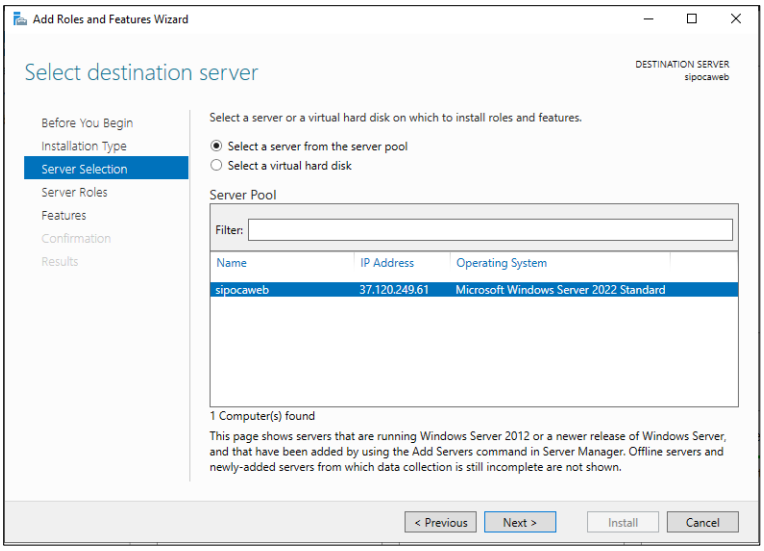


Figura 35 Instalarea serverului web IIS
(selectarea serverului din cadrul colecției de servere gestionate de Windows Server 2022)

Pasul 2: Selectarea tipului de instalare pe bază de roluri sau caracteristici:

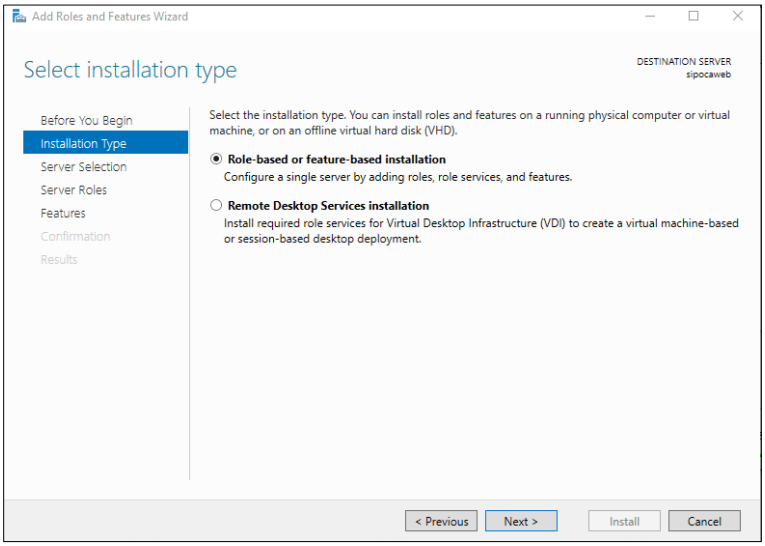


Figura 36 Selectarea tipului de instalare a componentei IIS din Windows Server 2022.

Pasul 3: Selectarea componentelor necesare în vederea instalării serverului web IIS:

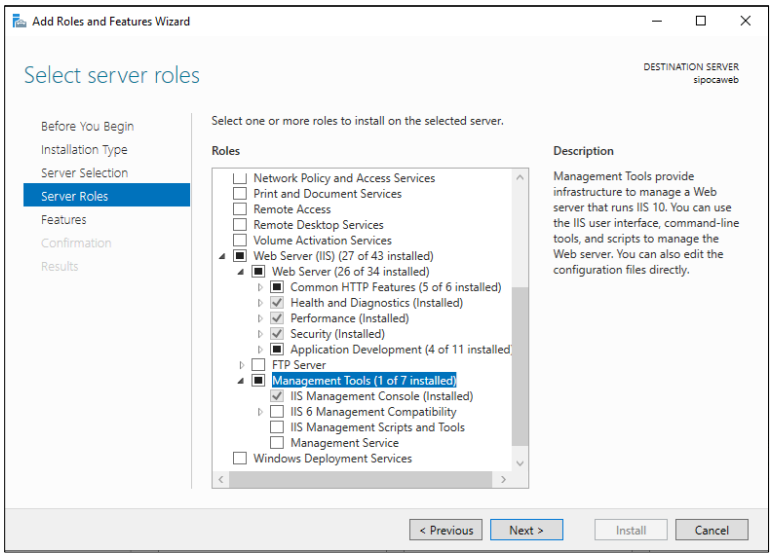


Figura 37 Selectarea componentelor Web Server și Management Tools, cu subcategoriile necesare

Pasul 4: Precizarea tipurilor de aplicații suplimentare necesare găzduirii de servicii web de tip WCF:

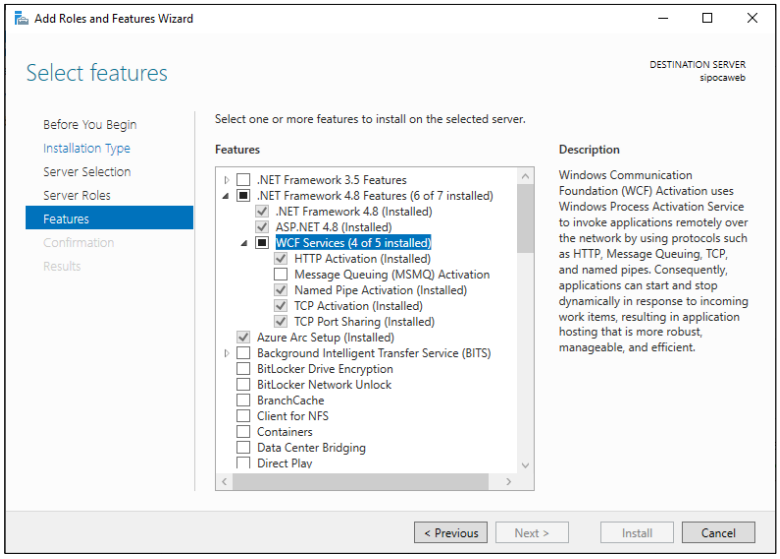


Figura 38 Selectarea opțiunii WCF services aferente cadrului de dezvoltare .NET Framework 4.8.

Odată instalat, serverul web IIS poate accepta găzduirea unui mare număr de aplicații web ce vor fi apelate prin intermediul directoarelor virtuale pe care IIS le creează pentru fiecare dintre ele, prin punerea în corespondență a numelui subdomeniului aferent cu un director fizic situat pe server.

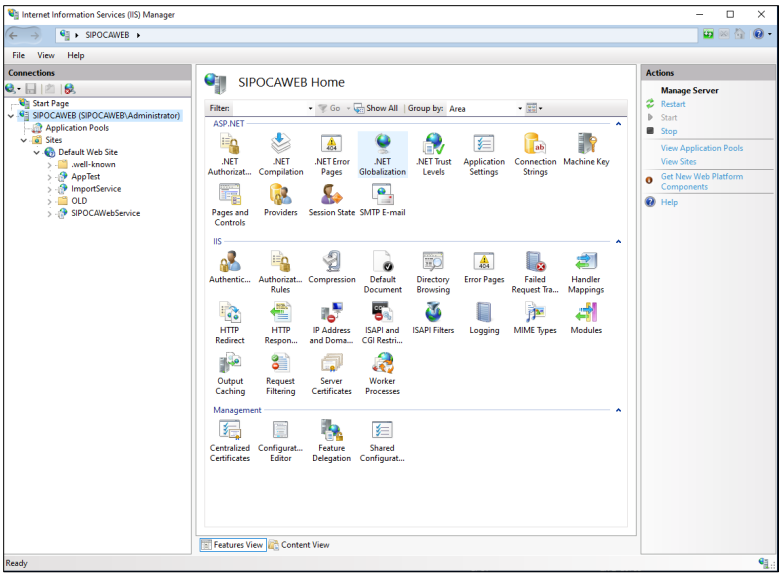


Figura 39 Fereastra de afișare a serverului care găzduiește aplicațiile web necesare funcționării platformei PPCA

Pasul 5: Instalarea unui certificat de tip SSL pentru serviciul web

În vederea asigurării securității transportului mesajelor transmise între client și serverul web s-a utilizat practica securizării printr-un certificat de tip SSL (Secure Socket Layer). Un certificat de server SSL urmărește două scopuri principale: confirmarea identității serverului înainte de a-l autentifica și stabilirea unui canal criptat de comunicare între server (site-ul web) și client (browserul utilizatorului final care se conectează la acesta).

Certificatul SSL poate fi achiziționat de la o autoritate de certificare, o organizație de încredere care emite certificate digitale pentru site-uri web și alte entități. Pentru testarea funcționalității de securizare a transportului serviciului web dezvoltat, a fost generat un certificat gratuit prin intermediul instrumentului CertBot. Acesta este un utilitar software de tip open-source ce utilizează certificate de tip Let's Encrypt pe site-uri web administrate manual pentru a activa protocolul HTTPS (Hyper Text Transfer Protocol Secure).

Exemplu de procedură pentru generarea certificatului de tip SSL cu extensia .pfx de către utilitarul CertBot

Se deschide CMD cu drepturi de administrator:

Se execută în ordine următoarele comenzi:

net stop W3SVC

certbot certonly --standalone

Certificatele plătite oferă niveluri de criptare îmbunătățite față de opțiunile gratuite, cum ar fi Let's Encrypt și oferă caracteristici suplimentare, cum ar fi validarea extinsă (EV), suport pentru wildcard și sigiliile site-ului pentru a dovedi autenticitatea. De aceea, se recomandă ca pe serverul web de producție ce va fi pus în funcțiune să se instaleze un certificat achiziționat de la o autoritate de certificare.

Conform (Techopedia, 2023), IIS acceptă orice fel de certificat criptografic utilizat în infrastructura de chei publice a Internetului care este gestionat de Serviciul de informare Internet al Microsoft – software-ul de server utilizat împreună cu serverele care rulează sisteme de operare Microsoft Windows.

Gestionarea certificatelor IIS este gestionată în mod special de consola MMC (Microsoft Management Console) care lucrează împreună cu o aplicație web. Cu aceste programe, administratorii pot vizualiza cererile de certificate emise, în așteptare, revocate și eșuate.

Fie în urma generării unui certificat gratuit, prin utilitarul CertBot, fie a achiziționării unui certificat de tip SSL de la o autoritate de certificare, acesta trebuie inclus în colecția de certificate înregistrate în cadrul serverului web IIS. Aceasta presupune importul prin intermediul interfeței IIS, utilizând opțiunea Server Certificates a numelui serverului web.

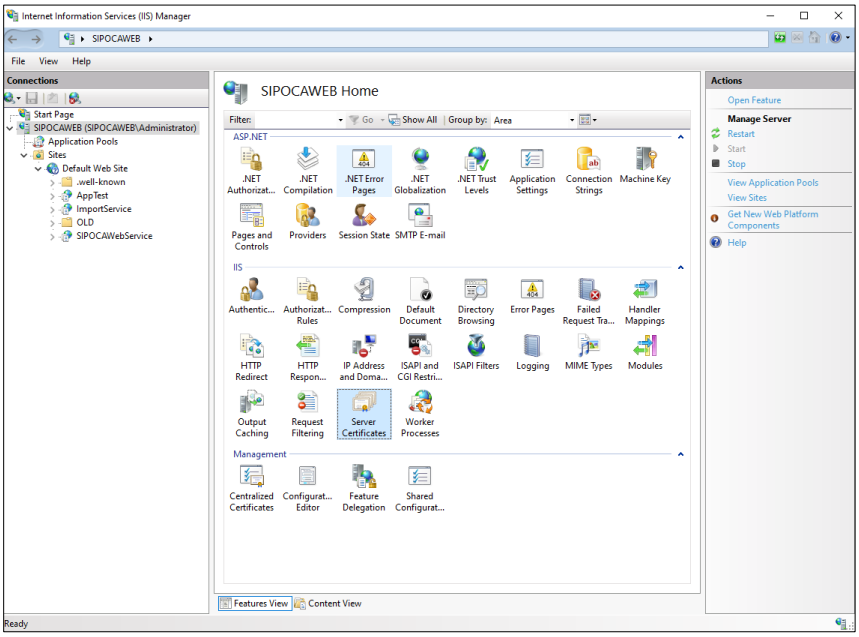
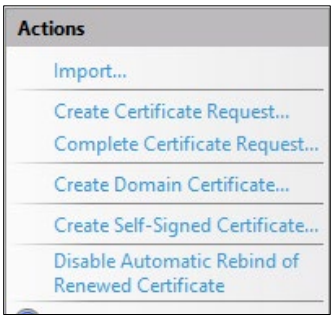


Figura 40 Accesarea certificatelor înregistrate pe serverul web IIS

Importul certificatului se va realiza prin opțiunea Import ... din cadrul listei de acțiuni disponibile, ceea ce presupune selectarea fișierului cu extensia .pfx aferent certificatului care urmează a fi importat.



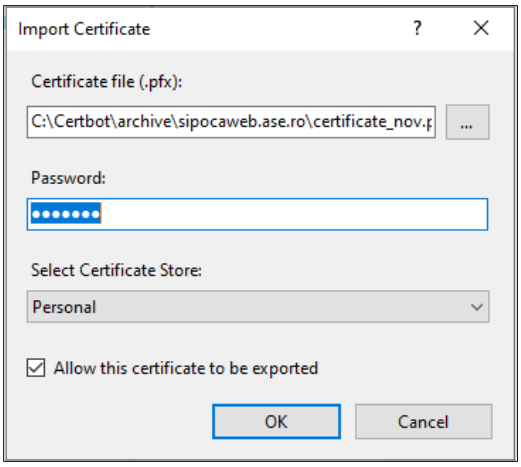


Figura 41 Procedura de import a certificatului de tip SSL în cadrul serverului web IIS

Pentru a stabili care este combinația de nume de host, numărul portului, și adresa de IP la care un site web este disponibil în rețeaua Internet, IIS utilizează conceptul de Legături (Bindings). Această secțiune este cea în care trebuie precizat protocolul HTTPS, pentru a fi recunoscut de către toate aplicațiile web găzduite.

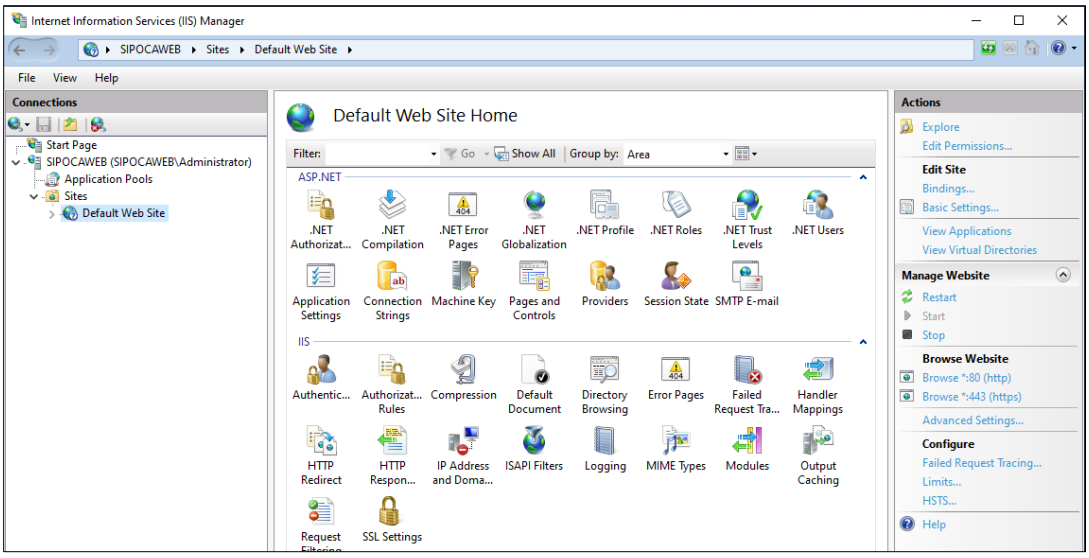


Figura 42 Adăugarea certificatului ca parte a protocolului HTTPS ce va fi acceptat de site-ul web

La adăugarea unei noi legături (Binding) de tip HTTPS trebuie specificat numele certificatului instalat anterior în IIS cu precizarea parolei stabilite la emiterea acestuia.

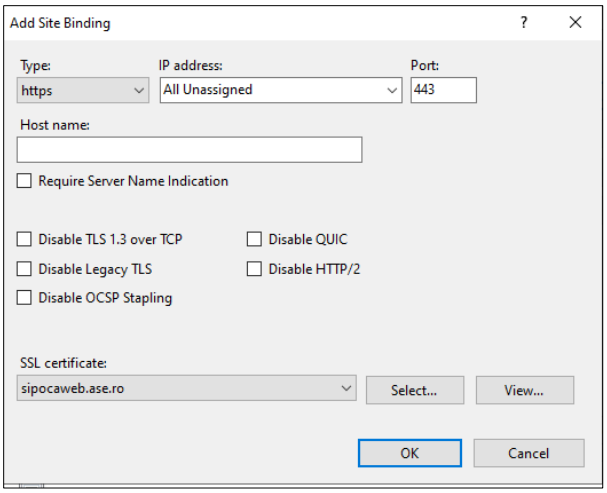


Figura 43 Adăugarea unei noi legături de tip HTTPS securizate prin certificatul instalat în IIS

Pasul 6: Utilizarea protocolului HTTPS la apelarea tuturor aplicațiilor web de pe serverul IIS:

Implicit, IIS utilizează protocolul HTTP pentru a permite primirea cererilor de tip HTTP Request către aplicațiile web. În scenariul de față securizarea transportului schimbului de mesaje între aplicațiile client și serviciul web trebuie să se realizeze utilizând protocolul HTTPS prin certificatul de tip SSL instalat. Pentru a obliga clientul să folosească acest protocol în loc de HTTP, trebuie selectată opțiunea *SSL Settings* din fereastra principală cu caracteristicile site-ului web, apoi bifată opțiunea Require SSL.

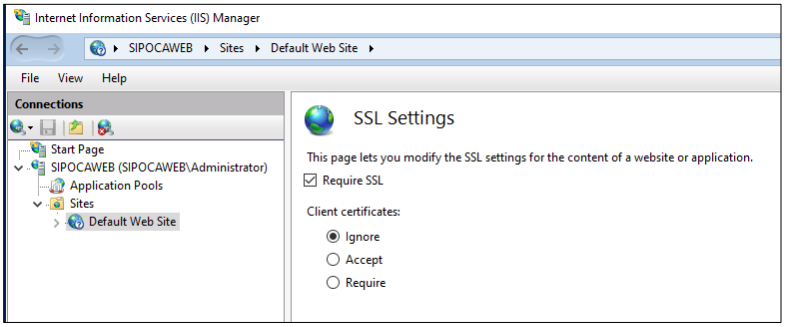


Figura 44 Stabilirea protocolului HTTPS prin utilizarea obligatorie a certificatului de tip SSL pentru aplicațiile web

4.5 Distribuirea aplicației informatice privind colectarea datelor prin programul PPCA

Pentru a putea fi accesat de către sistemul informatic operațional, potrivit arhitecturii software și funcționalităților detaliate în livrabilul *Platformă funcțională de colectare a datelor privind PPCA*, serviciul web trebuie publicat pe serverul web pregătit în etapele anterioare. Serviciul web dezvoltat va fi distribuit pe Internet pentru utilizare sub forma unei aplicații web apelate prin protocolul HTTPS, numele domeniului (sau adresa de IP) expus (ă) în Internet și numele serviciului web. În acest sens trebuie creată aplicația web care va crea corespondența între numele directorului virtual aferent ei (care va defini calea către numele serviciului web cu extensia .svc) și directorul fizic unde rezidă varianta compilată a serviciului web.

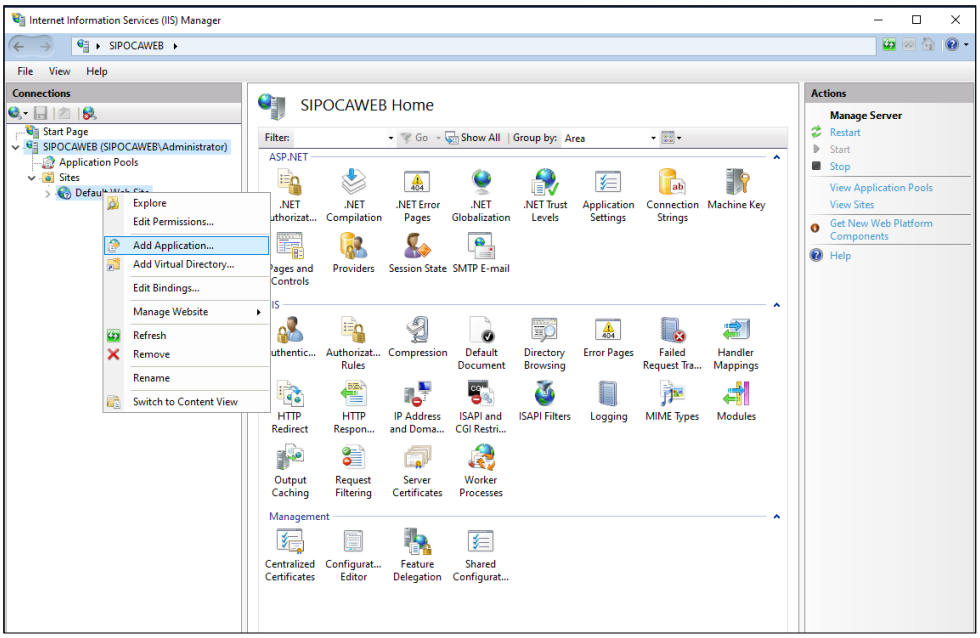


Figura 45 Modalitatea de creare a unei noi aplicații web pe serverul web IIS

Înainte de a genera versiunea compilată a serviciului web, trebuie specificate noile coordonate ale serverului bazei de date, precum și datele de autentificare și autorizare la baza de date de tip DataLake. Acestea pot fi precizate în fișierul web.config din cadrul proiectului Visual Studio în secțiunea connectionStrings, prin modificarea informațiilor privind numele serverului (data source), numele bazei de date (initial catalog), numele utilizatorului (user id), parola (password), așa cum se pot ele observa în caseta de mai jos:

```
<connectionStrings>

    <add name="WCFImportEntities"
connectionString="metadata=res://*/DAL.mdlImport.csdl|res://*/DAL.mdlImport.s
sdl|res://*/DAL.mdlImport.msl;provider=System.Data.SqlClient;provider
connection string=&quot;data source=192.168.4.107;initial
catalog=ENVDataLakeImp;persist security info=True;user
id=……;password=……;MultipleActiveResultSets=True;App=EntityFrame
work&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>
```

Generarea versiunii compilate a serviciului web prin împachetarea codului sursă sub forma unui fișier de tip .dll se va realiza prin utilizarea meniului Publish <<Nume serviciu web>> din meniul Build al aplicației Visual Studio.

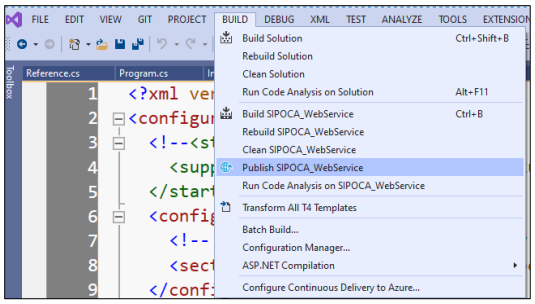


Figura 46 Generarea versiunii compilate a serviciului web.

Fișierele rezultate în urma compilării serviciului web vor fi copiate în cadrul directorului fizic situat pe serverul web. Este necesar, pentru aceasta, comutarea în modul de vizualizare Content View pentru directorul virtual al aplicației web create anterior, care va afișa directorul fizic unde fișierele pot fi copiate pentru a fi recunoscute de către IIS la accesarea serviciului web.

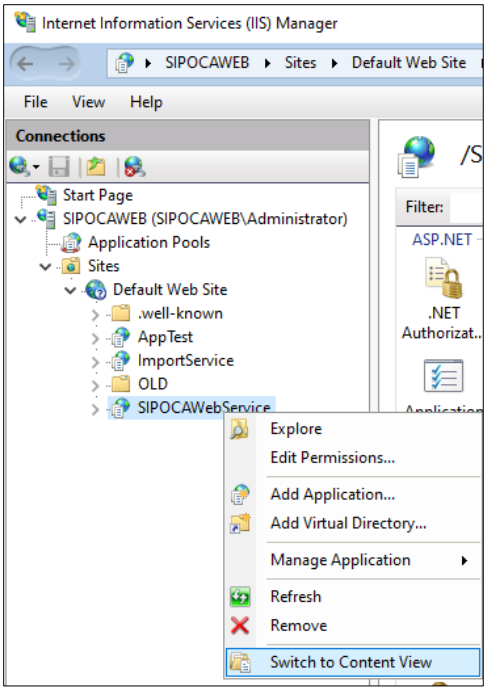


Figura 47 Modalitatea de publicare a codului sursă aferent serviciului web în directorul virtual specific din IIS

În urma acestei etape poate fi testată funcționarea serviciului web prin apelarea sa în cadrul unui browser web (pentru exemplificare am utilizat adreas serverului web IIS utilizat în etapa de dezvoltare *https://sipocaweb.ase.ro/SIPOCAWebService/ImportService.svc*)

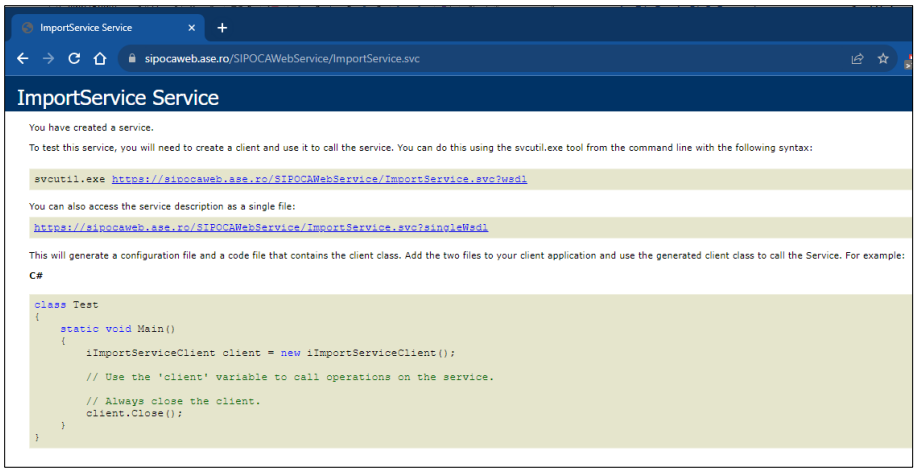


Figura 48 Testarea publicării serviciului web pe serverul IIS

Publicarea tablourilor de bord dezvoltate în Power BI

Pentru publicarea tablourilor de bord construite în Power BI este necesar ca utilizatorul să fie conectat la contul său, după care să efectueze clic pe pictograma *Publish* din Power BI Desktop (Figura 49).

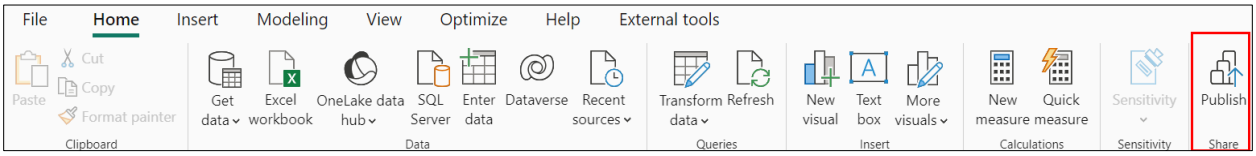


Figura 49 Meniul Power BI Desktop

Pasul următor constă în selectarea spațiului de lucru unde urmează să fie publicate tablourile de bord (Figura 50) și efectuarea unui clic pe butonul *Select*.

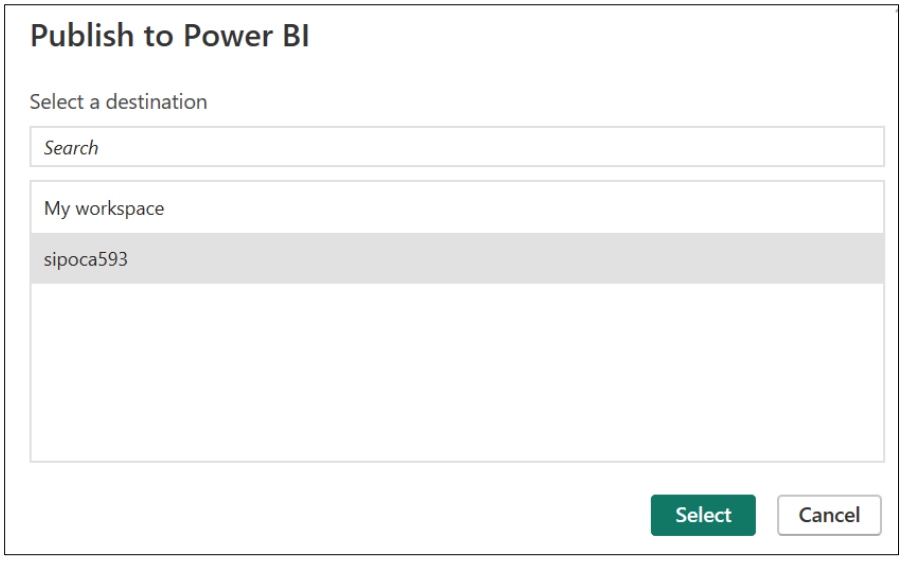


Figura 50 Selectarea spațiului de lucru unde vor fi publicate tablourile de bord

În Figura 51 este prezentată legătura Server SQL – Model semantic – Raport Power BI pentru sistemul dezvoltat.

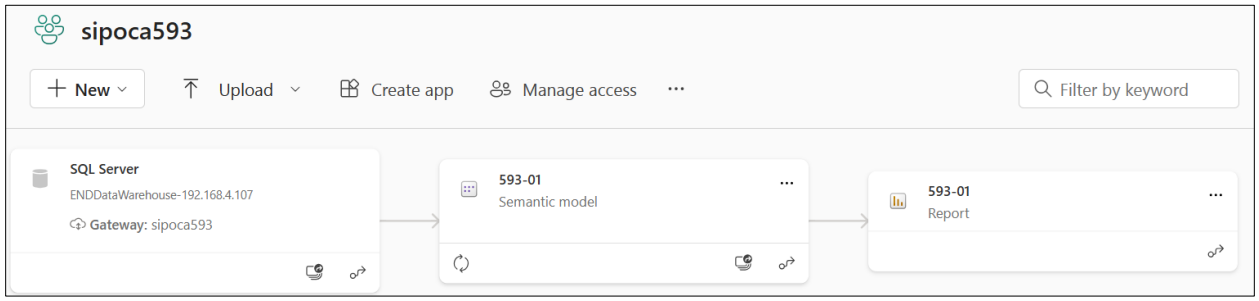


Figura 51 Prezentarea legăturii Server SQL – Model semantic – Raport Power BI

4.6 Actualizarea surselor de date ale tablourilor de bord

Pentru actualizarea surselor de date ale tablourilor de bord este necesară parcurgerea următoarelor etape:

- Instalarea gateway-ului de date pe serverul local
- Configurarea actualizării datelor în Power BI

Instalarea gateway-ului de date pe serverul local

Gateway-ul de date acționează ca o punte care oferă un transfer al datelor rapid și sigur între serverul SQL și serviciile Power BI din cloud.

Principalii pași care trebuie parcurși pentru utilizarea gateway-ului sunt:

- Descărcarea și instalarea gateway-ul pe serverul de date.
- Configurarea gateway-ul în funcție de politicile de securitate existente la nivelul firewall-ului.
- Adăugarea de noi administratori ai gateway-ului care pot gestiona.
- Utilizarea gateway-ul pentru actualizarea datelor din cloud cu cea mai recentă versiune a datelor locale.

Descărcarea și instalarea gateway-ul pe serverul de date

(i) Descărcarea kit-ului de instalare de la adresa https://go.microsoft.com/fwlink/?LinkId=2116849&clcid=0x409

(ii) Alegerea folderului unde urmează să fie instalată aplicația

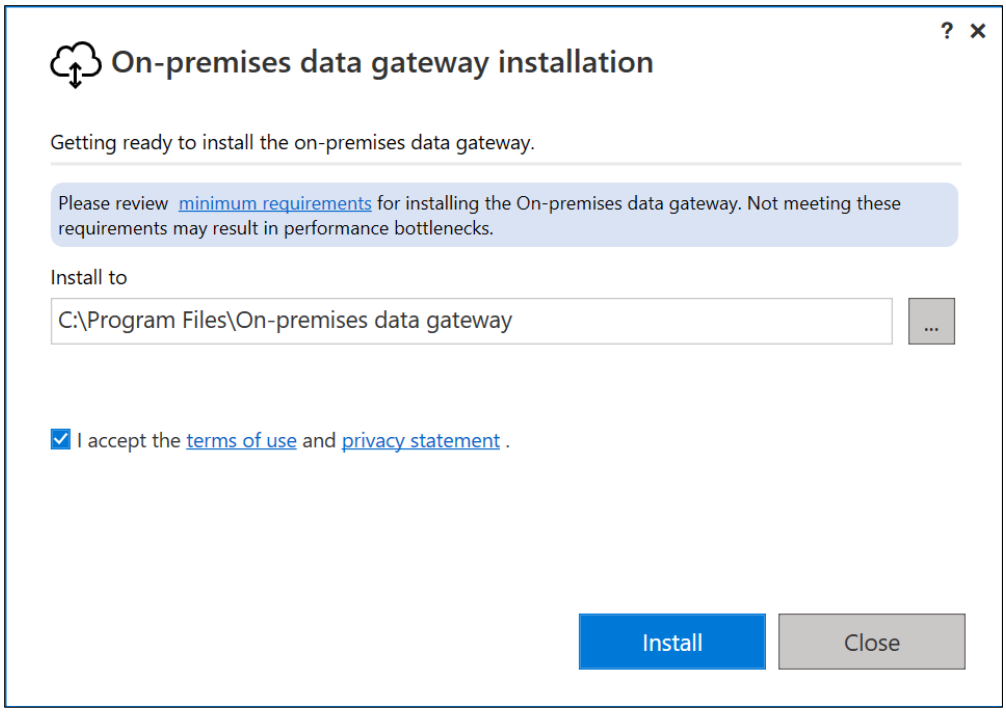


Figura 52 Instalarea gateway-ului de date

(iii) Introducerea adresei de e-mail pentru contul de instituțional Office 365

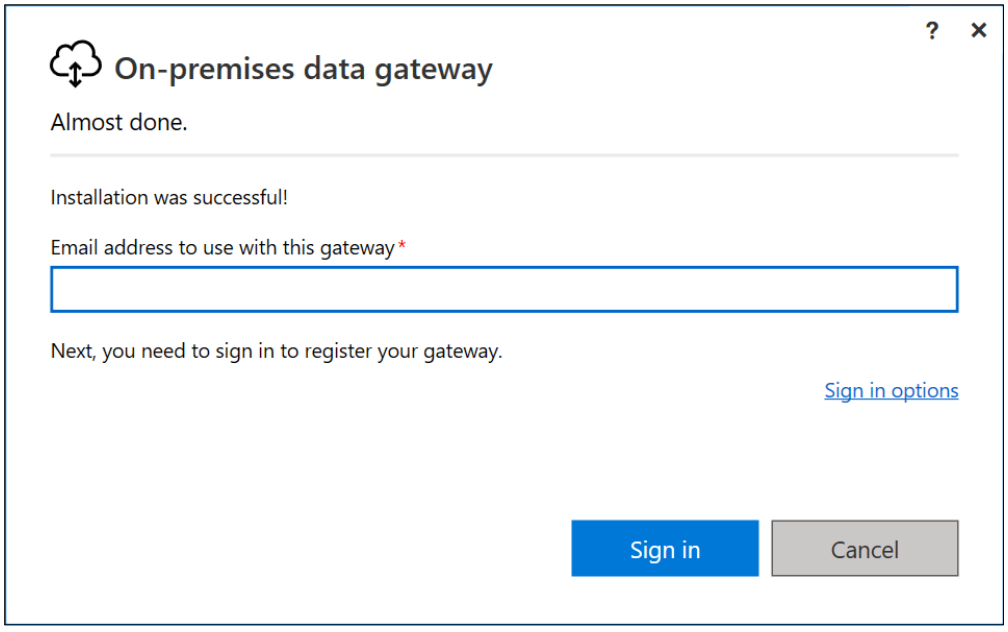


Figura 53 Instalarea gateway-ului de date

(iv) Selectarea opțiunii Register a new gateway on this computer

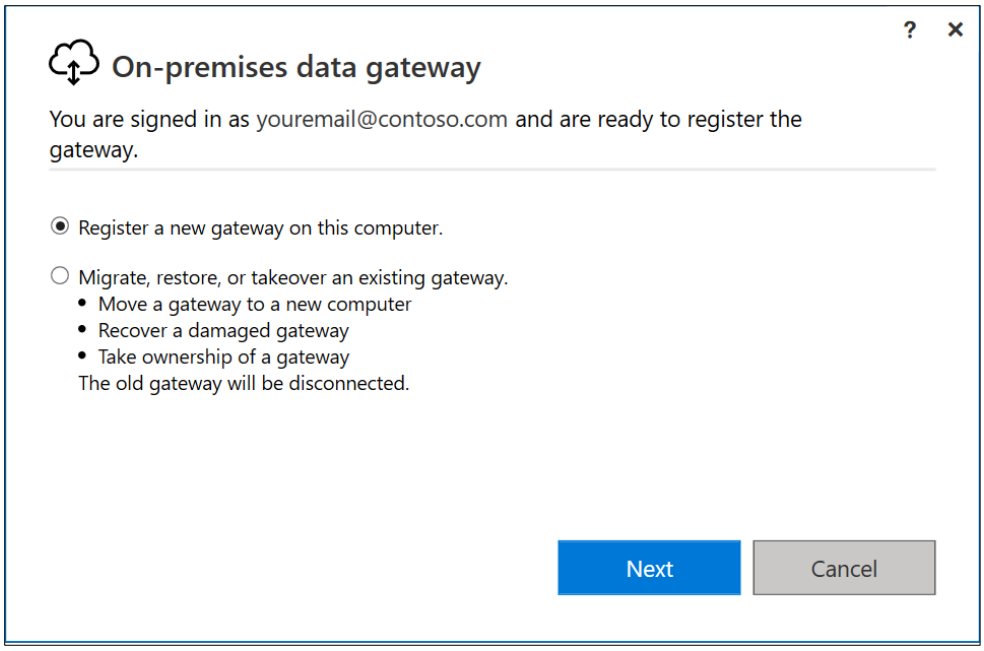


Figura 54 Instalarea gateway-ului de date

(v) Stabilirea numelui gateway-ului și a cheii de recuperare (*recovery key*). Cheia de recuperare este necesară dacă se dorește recuperarea sau mutarea gateway-ului.

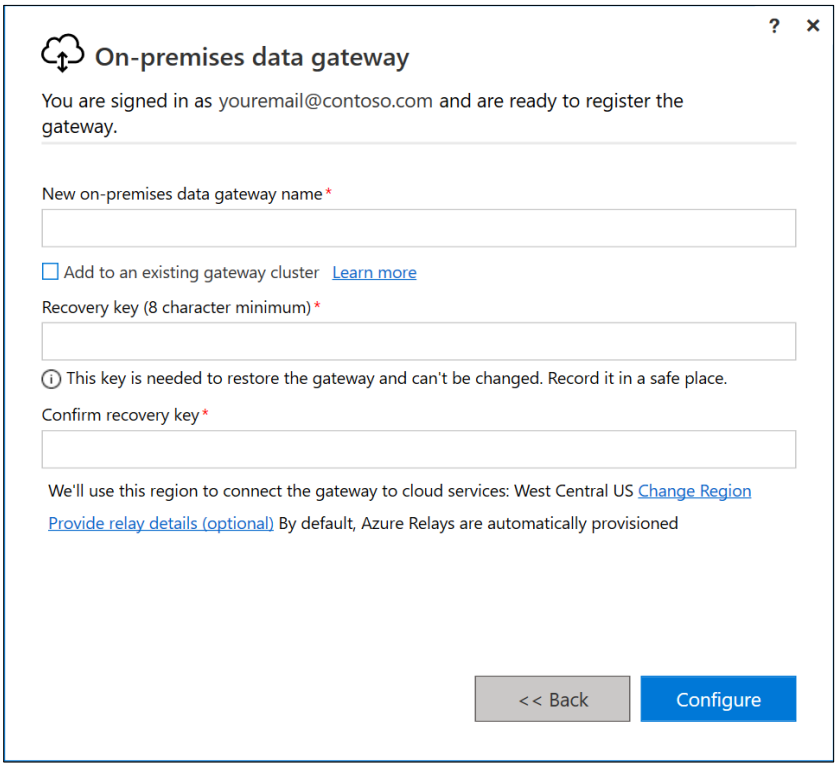


Figura 55 Instalarea gateway-ului de date

Configurarea actualizării datelor în Power BI

Actualizarea datelor înseamnă reimportarea datelor din sursele de date originale (i) pe baza unui program de actualizare sau (ii) la cerere. În cazul modificării frecvente a surselor de date, Power BI oferă posibilitatea mai multor actualizări zilnice, numărul acestora fiind limitat la opt actualizări zilnice. Cele opt valori de timp sunt stocate în baza de date backend și se bazează pe fusul orar local care a fost selectat pe pagina de configurare, iar planificatorul verifică modelul și ora la care trebuie actualizat. În Figura 56 este prezentată fereastra care permite configurarea actualizărilor în Power BI.

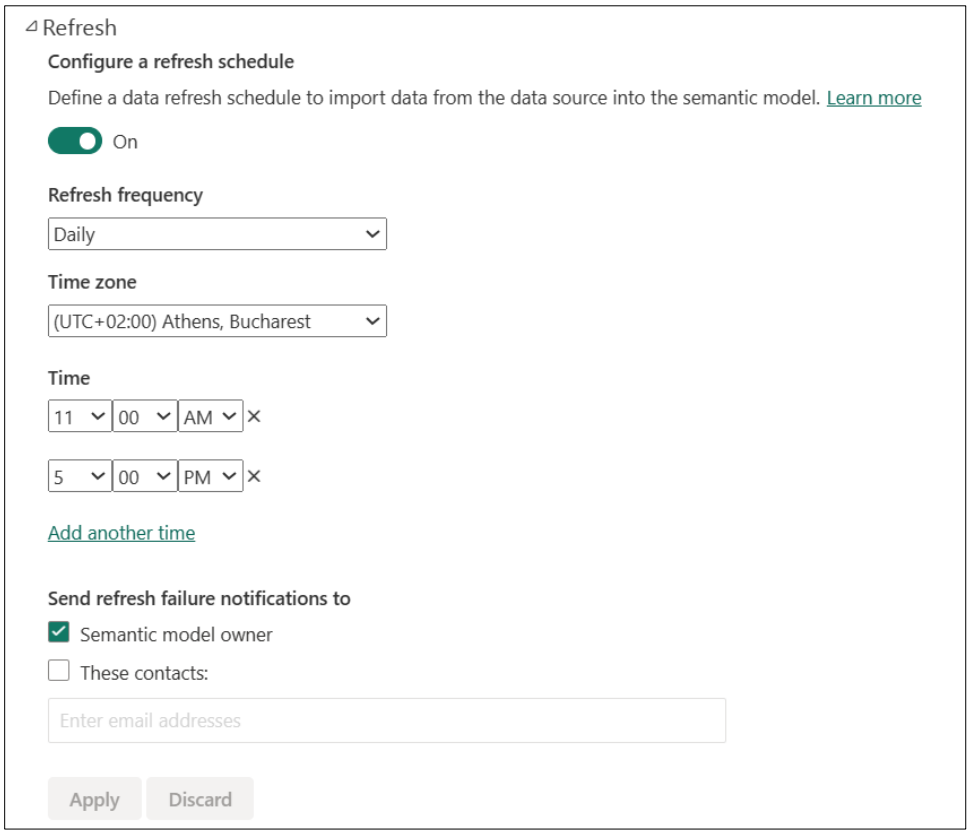


Figura 56 Configurarea intervalului de actualizare a datelor în Power BI

Gestionarea rolurilor in Power BI cloud

- Rolurile permit gestionarea utilizatorilor și acțiunilor pe care aceștia le pot face ce într-un spațiu de lucru (workspace), astfel încât echipele să poată colabora. Spațiile de lucru permit atribuirea de roluri utilizatorilor și grupurilor de utilizatori, cum ar fi grupurile de securitate, grupurile Microsoft 365 și listele de distribuție.
- Pentru a acorda acces la un spațiu de lucru, este necesară atribuirea unuia dintre următoarele roluri: Admin, Member, Contributor sau Viewer.
- Toată lumea dintr-un grup de utilizatori primește rolul atribuit, iar dacă cineva se află în mai multe grupuri de utilizatori, primește cel mai înalt nivel de permisiune oferit de rolurile care-i sunt atribuite.

Un utilizator cu o licență Power BI Pro poate fi membru a maxim 1.000 de spații de lucru.

Acordarea accesului/rolului în spațiul de lucru

Acordarea accesului în spațiul de lucru se realizează prin selectarea opțiunii *Manage access.*

Figura 57 Acordarea accesului în spațiul de lucru

Adăugarea grupurilor de securitate, listelor de distribuție, grupurilor Microsoft 365 sau utilizatorilor individuali la aceste spații de lucru ca Admin, Member, Contributor sau Viewer, urmata de selectarea butonului *Add*.

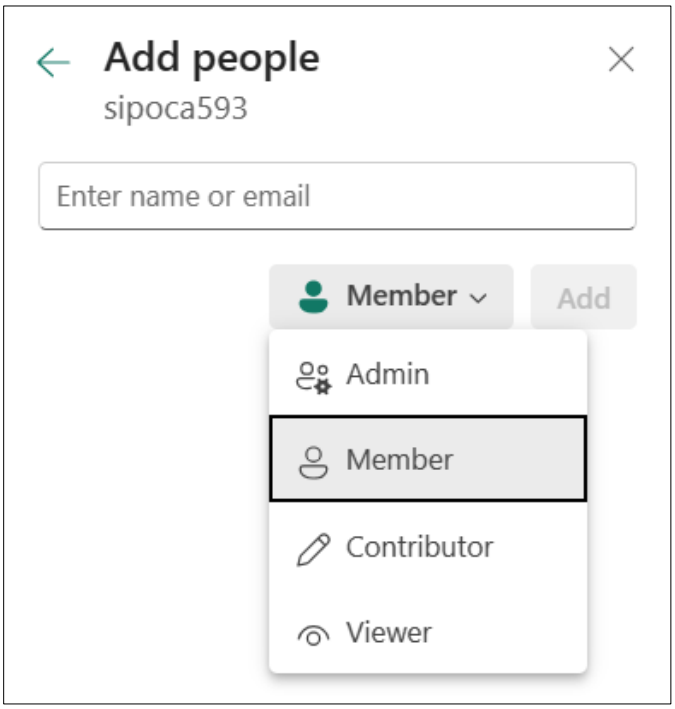


Figura 58 Adăugarea utilizatorilor

4.7 Stabilirea de politici de backup și restaurare în caz de incidente

Business Continuity Planning (BCP) este procesul de creare a unor sisteme de prevenire și recuperare pentru a face față amenințărilor potențiale la adresa unei organizații. Aceasta implică analiza impactului unei întreruperi, identificarea funcțiilor și resurselor critice și elaborarea unor planuri de urgență. Business Continuity Planning (BCP) face parte din teoria managementului riscurilor și a rezilienței.

Conceptul Business Continuity Planning (BCP) se bazează pe ideea că organizațiile ar trebui să fie capabile să își mențină sau să își reia operațiunile normale în cazul unui dezastru, cum ar fi o calamitate naturală, un atac cibernetic, o eroare umană, o defecțiune hardware etc. Dezastrele pot avea consecințe grave pentru organizații, cum ar fi pierderea reputației, a satisfacției clienților, a avantajului competitiv

etc. Prin urmare, organizațiile trebuie să aibă un plan pentru a-și proteja personalul și bunurile și pentru a se asigura că pot funcționa rapid și eficient atunci când are loc un dezastru.

Business Continuity Planning implică, de obicei, următoarele etape:

- Identificarea și evaluarea riscurilor potențiale care pot afecta activitatea, cum ar fi incendii, inundații, cutremure, pandemii, întreruperi de energie electrică, încălcarea securității datelor etc.
- Evaluarea impactului fiecărui risc asupra proceselor, funcțiilor și obiectivelor afacerii, cum ar fi vânzările, producția, distribuția etc.
- Determinarea obiectivului punctului de recuperare (Recovery Point Objective - RPO) și a obiectivului timpului de recuperare (Recovery Time Objective - RTO) pentru fiecare funcție, care reprezintă pierderea maximă acceptabilă de date și, respectiv, timpul de nefuncționare.
- Elaborarea și punerea în aplicare a strategiilor și procedurilor de prevenire, atenuare și recuperare a riscurilor, cum ar fi backup-ul, redundanța, failover-ul, relocarea etc.
- Testarea și revizuirea periodică a planului pentru a asigura eficacitatea și validitatea acestuia și pentru a-l actualiza, dacă este necesar.

Business Continuity Planning este important pentru orice organizație, indiferent de mărimea sau sectorul de activitate. Aceasta ajută organizațiile să se pregătească pentru situații neprevăzute, să reducă la minimum impactul întreruperilor și să își sporească rezistența și competitivitatea. Business Continuity Planning ajută, de asemenea, organizațiile să respecte cerințele legale și să răspundă așteptărilor părților interesate, cum ar fi clienții, angajații, furnizorii etc.

SQL Server este un sistem de gestiune a bazelor de date relaționale care stochează și procesează date pentru diverse aplicații. Bazele de date SQL Server sunt active valoroase pentru orice organizație, deoarece sprijină procesul de asistare a deciziilor. Prin urmare, este esențial să existe o strategie de backup și restaurare care să protejeze datele de orice amenințări potențiale și să permită recuperarea datelor în caz de dezastru.

O copie de rezervă este o copie a datelor care este stocată într-un loc diferit de locul în care sunt stocate datele originale. O restaurare este procesul de înlocuire a datelor curente cu cele existente în copia de rezervă. O strategie de backup și restaurare trebuie să ia în considerare următoarele aspecte:

- Obiectivele de recuperare: Acestea sunt obiectivele care definesc cât de multe pierderi de date și cât timp de nefuncționare sunt acceptabile în cazul unui dezastru. Obiectivul punctului de recuperare (Recovery Point Objective - RPO) reprezintă valoarea maximă a pierderii de date care poate fi tolerată. Obiectivul privind timpul de recuperare (Recovery Time Objective - RTO) reprezintă timpul maxim care poate fi permis pentru restaurarea datelor. Obiectivele de recuperare depind de cerințele de afaceri și de caracterul critic al datelor.
- Tipuri de copii de rezervă: Acestea sunt diferite metode de backup al datelor, în funcție de cantitatea și frecvența modificărilor. SQL Server acceptă trei tipuri de copii de rezervă: completă, diferențială și transaction log. O copie de rezervă completă copiază întreaga bază de date și oferă o recuperare completă la un moment dat. O copie de rezervă diferențială copiază numai datele care s-au modificat de la ultima copie de rezervă completă și oferă o copie de rezervă mai rapidă și mai mică. O copie de rezervă a jurnalului de tranzacții (transaction log) copiază înregistrările din

jurnal care surprind modificările aduse bazei de date și asigură o recuperare la un moment dat sau până la ultimul minut.

- Frecvența de backup: Acesta este programul de efectuare a copiilor de rezervă, bazat pe obiectivele de recuperare și pe tipurile de copii de rezervă. Frecvența de backup ar trebui să echilibreze compromisul dintre RPO și dimensiunea și performanța backup-ului. De exemplu, o frecvență de backup mai mare poate reduce RPO, dar poate crește dimensiunea backup-ului și impactul asupra performanței sistemului. Frecvența de backup ar trebui să ia în considerare, de asemenea, rata de modificare a datelor și fereastra de backup.
- Locul unde sunt salvate copiile de rezervă: reprezintă destinația de stocare a copiilor de rezervă, cum ar fi: calculatorul local, un echipament din rețea sau un serviciu de stocare în cloud. O locație de backup ar trebui să fie sigură și accesibilă și ar trebui să dispună de suficient spațiu și lățime de bandă pentru a găzdui copiile de rezervă. De asemenea, locația de backup ar trebui să fie separată geografic de datele originale, pentru a evita riscul de a pierde atât datele, cât și copiile de rezervă în același dezastru. SQL Server suportă backup-ul și restaurarea din Azure Blob Storage, care oferă stocare de backup la distanță și scalabilă.
- Verificarea copiilor de rezervă: reprezintă procesul de verificare a validității și consistenței copiilor de rezervă și de asigurare a faptului că acestea pot fi restaurate cu succes. SQL Server oferă diverse metode de verificare a copiilor de rezervă, cum ar fi sumele de control (checksum), restaurarea anteturilor (restore headers), restaurarea doar cu verificare (restore verify only) și restaurările de test (test restores). Sumele de control reprezintă o modalitate de a detecta orice corupere sau modificare a datelor de backup. Restaurarea anteturilor este o modalitate de a verifica metadatele copiei de rezervă, cum ar fi tipul de copie de rezervă, data copiei de rezervă și dimensiunea copiei de rezervă. Restore verifyonly este o modalitate de a verifica integritatea logică și fizică a copiei de rezervă fără a o restaura efectiv. Test restores reprezintă o modalitate de a efectua o restaurare completă sau parțială a copiei de rezervă într-o locație sau bază de date diferită și de a verifica rezultatele.
- Criptare copiilor de rezervă: reprezintă procesul de protejare a copiilor de rezervă împotriva accesului neautorizat sau a falsificării. SQL Server oferă diverse metode de criptare a copiilor de rezervă, cum ar fi criptarea transparentă a datelor (Transparent Data Encryption - TDE), criptarea copiilor de rezervă sau instrumente de criptare de la terți. TDE este o modalitate de criptare a datelor, atât în baza de date, cât și în copia de rezervă. Criptarea copiilor de rezervă este o modalitate de criptare a datelor în timpul procesului de backup, folosind un certificat sau o cheie asimetrică. Instrumentele de criptare ale terților sunt aplicații externe care pot cripta sau decripta fișierele de backup.

Pe lângă aspectele de mai sus, o strategie de backup și restaurare trebuie să ia în considerare și următoarele strategii care pot îmbunătăți disponibilitatea și performanța bazelor de date SQL Server:

- Point-in-time recovery: Aceasta este o tehnică care permite restaurarea bazei de date la un anumit moment în timp. Acest lucru poate fi util pentru recuperarea în cazul unor erori logice, cum ar fi modificări accidentale sau coruperea datelor. SQL Server suportă recuperarea punctuală prin utilizarea copiilor de rezervă ale jurnalului de tranzacții, care conțin înregistrările jurnalului modificărilor efectuate în baza de date. Prin aplicarea în succesiune a copiilor de rezervă ale

jurnalului de tranzacții, baza de date poate fi restaurată la orice moment din perioada de păstrare a copiilor de rezervă.

- Log shipping: Aceasta este o tehnică care implică două sau mai multe instanțe SQL Server și copierea copiilor de rezervă ale jurnalului de tranzacții de la o instanță la alta. Copiile de rezervă ale jurnalului de tranzacții sunt aplicate la fiecare dintre instanțele secundare în mod individual. Expedierea logurilor oferă o soluție de recuperare în caz de dezastru pentru o singură bază de date principală și una sau mai multe baze de date secundare, fiecare pe o instanță SQL Server separată. Log shipping suportă, de asemenea, accesul limitat doar pentru citire la bazele de date secundare, în timpul intervalului dintre lucrările de restaurare. Log shipping poate fi configurat și monitorizat folosind SQL Server Management Studio sau SQL Server Agent jobs.
- Mirroring: Aceasta este o tehnică care implică două instanțe SQL Server și transferul înregistrărilor din jurnalul de tranzacții de la o instanță la alta. Înregistrările din jurnalul de tranzacții sunt aplicate instanței secundare în mod sincron sau asincron. Mirroring oferă o soluție de înaltă disponibilitate pentru o singură bază de date primară și o singură bază de date secundară, fiecare pe o instanță SQL Server separată. De asemenea, mirroringul suportă failoverul automat și accesul complet numai pentru citire la baza de date secundară. Mirroring-ul poate fi configurat și monitorizat utilizând SQL Server Management Studio sau SQL Server Agent jobs.

5 Monitorizarea funcționării sistemului informatic după etapa de punere în funcțiune

Monitorizarea unui sistem informatic aflat în exploatare implică urmărirea constantă și analiza unor parametri ce furnizează informații extrem de valoroase privind modul de funcționare al sistemului și maniera concretă în care acesta este folosit de beneficieri. Monitorizarea acestor aspecte permite detectarea anumitor tipare de utilizare a unei aplicații și abordarea proactivă a diferite probleme sau disfuncționalități, ce pot fi identificate și rezolvate cu implicații minime asupra utilizatorilor finali, ca și asupra proceselor și activităților curente sau infrastructurii informatice a organizației. Monitorizarea unei aplicații reclamă utilizarea de instrumente și tehnici specifice, care diferă în funcție de aspectele vizate și care pot varia de la soluții pentru jurnalizarea erorilor, până la platforme de monitorizare sofisticate, care generează alerte în timp real și dispun de instrumente de natură grafică pentru analiza performanței sistemului.

Într-o prezentare succintă, principalele argumente pentru monitorizarea funcționării unui sistem informatic sunt următoarele:

- Detectarea problemelor în timp real – Monitorizarea continuă permite detectarea timpurie a unor probleme precum erorile de sistem sau performanța nesatisfăcătoare, care pot fi remediate înainte ca acestea să cauzeze discontinuități în utilizarea sistemului, cu evidente efecte negative în plan operațional și al costurilor.
- Monitorizarea erorilor – Prin jurnalizarea sistematică și analiza erorilor se pot identifica și rezolva potențiale erori sau omisiuni cauzate de aspecte specifice mediului de exploatare, care nu au putut fi anticipate și tratate adecvat în etapa de dezvoltare a sistemului. Drept urmare, monitorizarea erorilor constituie o premisă a stabilității și fiabilității în exploatarea curentă a oricărei aplicații.
- Creșterea fiabilității sistemului – Monitorizarea continuă a modului de funcționare a unei aplicații permite detectarea și rezolvarea potențialelor puncte de eșec (a oricăror cauze și condiții care pot determina discontinuități în utilizare), conducând la o fiabilitate crescută a sistemului și la minimizarea timpului de nefuncționare.
- Optimizare performanței sistemului – Monitorizarea vizează și urmărirea sistematică a unor indicatori de performanță asociați utilizării unei aplicații, precum timpii de răspuns la cererile de acces și de prelucrare a datelor sau consumul de resurse la nivelul sistemului. Este astfel posibilă identificarea unor blocaje cauzate de interogările lente ale bazelor de date sau algoritmii ineficienți, care pot cauza probleme de performanță la nivelul sistemului.
- Detectarea vulnerabilităților de securitate – În condițiile în care securitatea reprezintă un aspect critic pentru siguranța exploatării aplicațiilor, vulnerabilitățile și amenințările de securitate sunt vizate în mod explicit de instrumentele de monitorizare, pentru detectarea rapidă și gestionarea adecvată a breșelor de securitate și a potențialelor riscuri de acest tip.

În esență, calitatea oricărui produs software depinde de gradul în care acesta satisface setul cerințelor funcționale și nefuncționale definite anterior proiectării și implementării sistemului. Drept

consecință, monitorizarea constantă a aplicațiilor aflate în exploatare devine o necesitate obiectivă pentru asigurarea unui suport informatic adecvat proceselor de la nivelul organizației beneficiare. Într-un sens foarte strict, monitorizarea este procesul de colectare a oricăror date asociate cu funcționarea unui sistem sau cu folosirea efectivă a sistemului de utilizatorii cu diferite roluri. Pe de altă parte, obținerea acestor date nu reprezintă un scop în sine, fiind necesară analiza lor pentru a aprecia în ce măsură sistemul se conformează cerințelor definite anterior dezvoltării sale. Drept urmare, monitorizarea funcționării unei soluții software se poate descrie sub forma unui proces, cu o etapizare specifică, descrisă succint în paragrafele următoare:

- *Identificarea aspectelor care fac obiectul monitorizării* – Acestea pot include diverse componente ale infrastructurii IT, precum servere, baze de date, rețele etc. Este important să fie considerate toate elementele critice pentru funcționarea unei aplicații, având în vedere arhitectura acesteia și interacțiunile cu alte sisteme.
- *Configurarea instrumentelor și proceselor de monitorizare* – Presupune instalarea și configurarea unor instrumente dedicate pentru urmărirea performanței și disponibilității sistemelor și componentelor software vizate. Aceasta poate implica instalarea de agenți software pe servere, configurarea tablourilor de bord pentru monitorizare și definirea unor mecanisme de alertare și notificare. Desigur, caracteristicile instrumentelor și proceselor de monitorizare sunt influențate de cele ale sistemelor pe care le vizează, soluțiile software complexe impunând recurgerea la instrumente de monitorizare sofisticate și la procese de monitorizare complexe.
- *Colectarea și analiza datelor* – Instrumentele de monitorizare trebuie configurate pentru a colecta și stoca date privind modul de funcționare al sistemelor monitorizate. Prin analiza acestor date se pot identifica potențialele problemele, dar și contextele de factură IT sau organizațional-operațională sau anumite tipare de utilizare a aplicațiilor care favorizează apariția acestor probleme. Urmărirea sistematică a datelor furnizate de instrumentele de monitorizare permite ca eventualele aspecte problematice detectate să fie gestionate în mod proactiv, evitându-se apariția unor deficiențe majore în planul performanței sau disponibilității unui sistem informatic.
- *Rezolvarea potențialelor probleme* – Identificarea anumitor probleme prin monitorizarea IT trebuie să conducă la acțiuni corective prompte, pentru a limita consecințele negative asupra sistemelor informatice. Aceasta implică remedierea problemelor constatate, dar și implementarea unor măsuri preventive, menite să reducă riscul apariției unor probleme similare în viitor. Rezolvarea rapidă și eficientă a problemelor detectate este esențială pentru asigurarea unei fiabilități crescute a sistemelor informatice, pentru a minimiza timpul de nefuncționare sau, ideal, pentru a evita discontinuitățile în exploatarea acestora.

Soluțiile de monitorizare pot viza diferite niveluri de abstractizare, de la cel accesibil utilizatorilor finali ai aplicațiilor, până la cel al infrastructurii-suport (sistem de operare, server web, server de date, API, middleware etc). Din perspectiva utilizatorilor finali, sistemul informatic care asigură suport programului *Plătești pentru cât arunci* (PPCA) este reprezentat de setul tablourilor de bord Microsoft Power BI; pe de altă parte, disponibilitatea și gradul de actualitate al datelor privind colectarea deșeurilor sunt condiționate de funcționarea corespunzătoare a fiecărei componente din arhitectura acestui sistem.

Secțiunea de față se concentrează asupra instrumentelor de monitorizare care vizează componentele esențiale, sub aspect structural și funcțional, de la nivelul soluției informatice destinată programului PPCA:

- Bazele de date care corespund structurii de tip data lake ce colectează datele de la nivelul UAT, respectiv depozitului de date populat pe baza data lake și utilizat pentru prezentarea și analiza datelor;
- Serviciul web care permite transmiterea datelor de la nivelul UAT.

5.1 Monitorizarea serviciului web

În condițiile în care transmiterea datelor privind colectarea deșeurilor de către UAT se realizează prin utilizarea unui serviciu web, monitorizarea funcționării corespunzătoare a serverului web și a serviciului este esențială pentru asigurarea unei funcționări optime a soluției informatice aferente programului PPCA în ansamblul său. Având în vedere tehnologia utilizată la implementare, monitorizarea serviciului web beneficiază de instrumentarul specific Windows Communication Foundation (WCF) și IIS.

În contextul tehnologiei WCF, suportul specific monitorizării sistemului vizează următoarele aspecte :

- End to End (e2e) Tracing – Permite urmărirea execuției codului în infrastructura WCF, pentru depanarea unei aplicații și investigarea eventualelor erori; de asemenea, în cazul aplicațiilor funcționale, urmărirea poate fi activată pentru planificarea capacității sistemului și analiza performanței. WCF permite urmărirea tuturor aspectelor de interes pentru diagnoza unei aplicații, precum apelurile care vizează metodele expuse de servicii, excepțiile definite la nivelul codului, interceptarea evenimentelor semnificative.
- Jurnalizarea mesajelor – Urmărește înregistrarea conținutului mesajelor de intrare și de ieșire, pentru consum offline; este astfel posibilă identificarea mesajelor malformate și examinarea elementelor de securizare utilizate și a conținutului transmis, respectiv părțile criptate și semnate și părțile rămase intacte. Deoarece jurnalizarea nu are legătură cu formatul de transfer al mesajului, vizând mesajele decodate, configurarea corectă a înregistrării mesajelor va genera conținut în format text. În condițiile în care ieșirea XmlWriterTraceListener este un fișier care conține o secvență de fragmente XML, fără a reprezenta cod XML valid, în documentația oficială WCF se recomandă utilizarea instrumentul Service Trace Viewer (SvcTraceViewer.exe) pentru consultarea fișierelor în care au fost jurnalizate mesajele.
- Jurnalizarea evenimentelor – Permite urmărirea evenimentelor interne și a eventualelor probleme în jurnalul de evenimente Windows. Înregistrarea evenimentelor este activată în mod implicit și nu există mecanisme care să o dezactiveze; evenimentele asociate WCF pot fi vizualizate folosind instrumentul Windows Event Viewer, fiind grupate în secțiunile Application și Security. Jurnalul evenimentelor de securitate conține evenimente înregistrate în urma auditului de securitate realizat de WCF. Jurnalul evenimentelor aplicației conține evenimentele generate de WCF, majoritatea intrărilor indicând faptul că o anumită caracteristică nu a funcționat corespunzător pentru o aplicație (de exemplu, când urmărirea și înregistrarea mesajelor eșuează, sau în cazul evenimentelor critice și de eroare, ca erorile de pornire și blocările) sau semnalând

diverse situații specifice, precum activarea jurnalizării mesajelor (pentru a anunța administratorul că informațiile sensibile de la nivelul aplicației pot fi înregistrate în antetul și corpul mesajelor).

- Monitorizarea performanței – WCF dispune de un număr important de contoare pentru evaluarea performanței aplicației, inspectarea datelor colectate realizându-se-se prin intermediul utilitarului Performance Monitor (perfmon.exe), disponibil la nivelul platformei Windows. Contoarele de performanță WCF necesită nu doar o configurarea adecvată, în funcție de necesități, dar și ajustarea corespunzătoare a setărilor privind utilizarea memoriei serverului pentru realizarea efectivă a monitorizării. În funcție de aspectele vizate, contoarele de performanță pot opera pe trei niveluri diferite: Service (vizează serviciul, în ansamblul său), Endpoint (vizează modul în care sunt recepționate mesajele), respectiv Operation (vizează apelurile de invocare a operațiilor și modul în care funcționează acestea).
- Accesul la datele privind utilizarea serviciilor, pe baza Windows Management Instrumentation (WMI) – WMI reprezintă implementarea Microsoft a standardului Web-Based Enterprise Management (WBEM) și vizează modul în care pot fi accesate instrumentele de administrare, specifice unui anumit tip de aplicație. În cazul de față, WCF expune datele aferente utilizării serviciilor printr-un furnizor WMI, iar acestea pot fi accesate în multe moduri, în condițiile în care Microsoft oferă API WMI pentru scripturi, .NET Framework, aplicații Visual Basic sau C++.
- Alte opțiuni și instrumente WCF – WCF furnizează mai multe instrumente de tip GUI și linie de comandă pentru a facilita implementarea și gestionarea aplicațiilor WCF. Spre exemplu, instrumentul Service Trace Viewer (SvcTraceViewer.exe) permite vizualizarea, gruparea și filtrarea mesajelor de urmărire, în vederea identificării, diagnosticării și gestionării adecvate a problemelor care pot apărea în exploatarea curentă a serviciilor WCF.

În completarea elementelor enumerate mai sus, se impune observația că utilizarea oricărui serviciu web este condiționată de resursele disponibile la nivelul platformei care îl găzduiește. Este deci important să se monitorizeze utilizarea resurselor pe serverul IIS, cele mai relevante aspecte vizând:

- Memoria disponibilă la nivelul Windows Server;
- Utilizarea procesorului (consumul CPU de către procesele care rulează pe server);
- Cozile de așteptare aferente serviciilor (numărul cererilor HTTP care trebuie gestionate de IIS);
- APP_POOL_WAS (starea curentă a pool-ului de aplicații) – Windows Process Activation Service (WAS) permite gestionarea proceselor care vizează aplicațiile ce găzduiesc servicii WCF.

Toate aceste elemente pot vizualizate în timp real sau jurnalizate, cu ajutorul utilitarului Windows dedicat (Performance Monitor).

5.2 Monitorizarea bazelor de date

În condițiile în care atât depozitul de date care facilitează analizele privind colectarea deșeurilor, cât și structura de tip data lake care stochează datele furnizate de UAT și alimentează depozitul, sunt găzduite și gestionate în mediul Microsoft SQL Server, monitorizarea corespunzătoare a stării serverului de date și a activații înregistrate la nivelul acestuia este esențială pentru exploatarea curentă a sistemului informatic aferent programului PPCA. În acest scop se poate utiliza setul amplu de funcționalități și instrumente dedicate, disponibile în SQL Server, care permit administrarea adecvată a bazelor de date și detectarea rapidă a diferite blocaje și probleme de performanță. Fundamentul acestor instrumente este reprezentat de un set de vederi (Dynamic Management Views – DMV) și funcții (Dynamic Management Functions - DMF), esențiale pentru administrarea curentă a serverului și bazelor de date, care returnează informații vitale pentru a monitoriza starea unei instanțe SQL Server, pentru a identifica și diagnostica eventualele problemele și pentru a regla performanța. În plus, multe dintre instrumentele disponibile pentru monitorizarea activității serverului permit vizualizarea acestor informații de stare în mod interactiv și în dinamică, prin intermediul a diferite tablouri de bord sau rapoarte, în funcție de necesitățile informaționale existente la un moment dat.

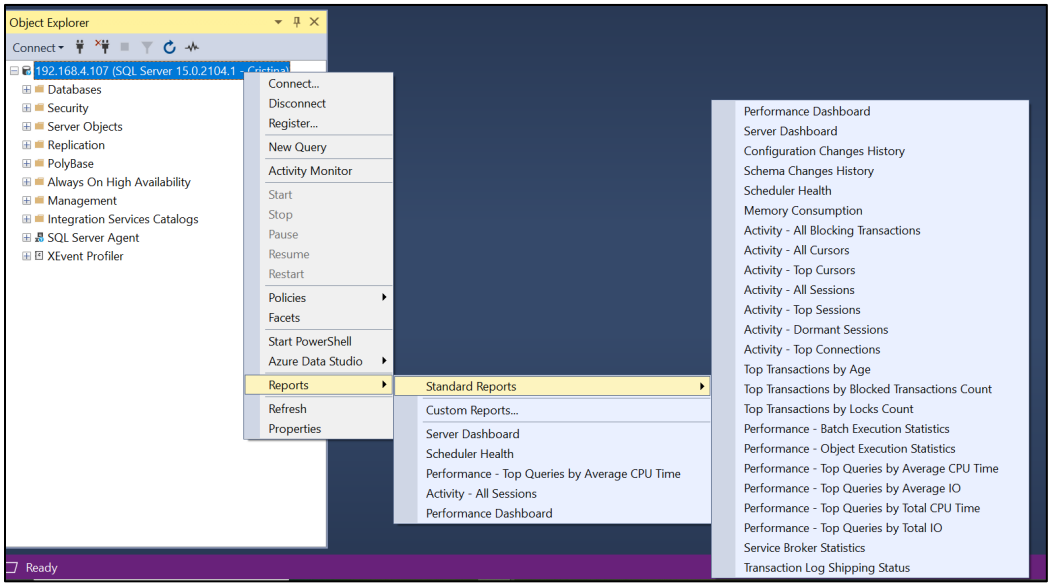


Figura 59 Monitorizarea serverului de date - rapoarte predefinite disponibile în SQL Server Management Studio

Întrucât setul instrumentelor care permit monitorizarea și analiza activității la nivelul serverului de date este extrem de amplu, cu titlu de exemplu este redată o captură de ecran aferentă tabloului de bord (Server Dashboard) ce oferă informații despre starea curentă a instanței SQL Server, respectiv modul de configurare și activitatea înregistrată la nivelul acesteia.

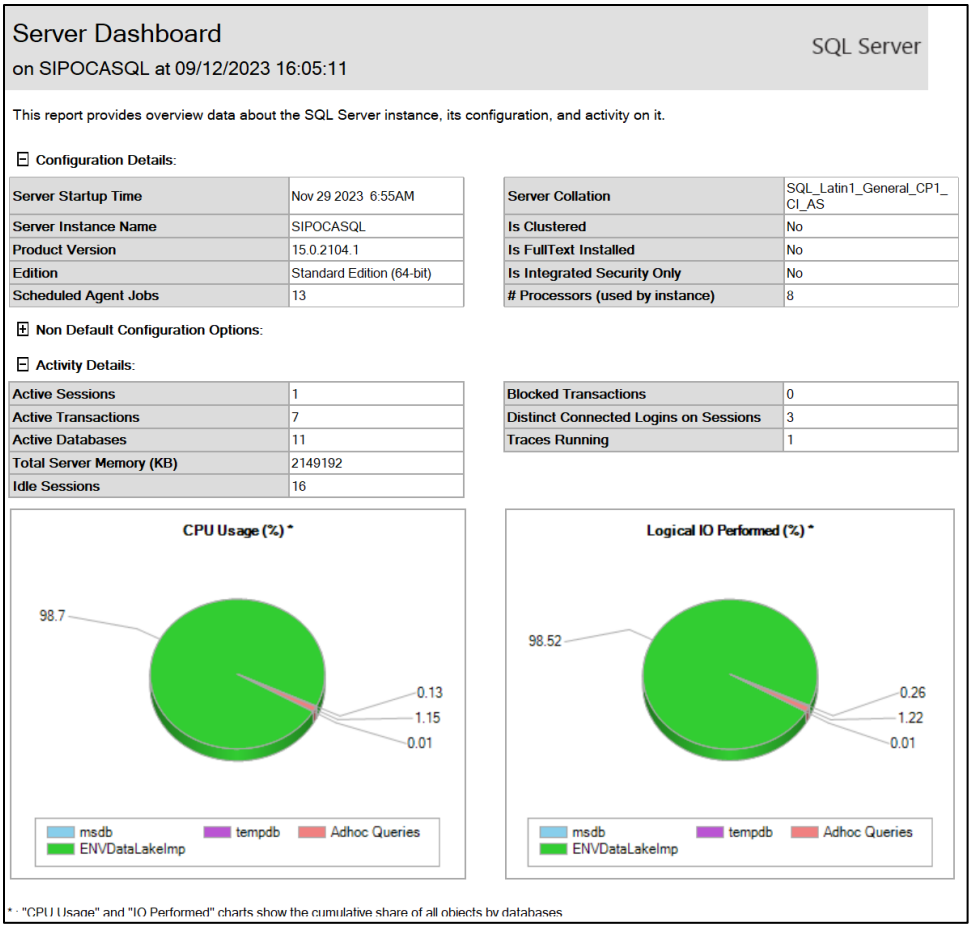


Figura 60 Starea serverului de date, redată de instrumentul SQL Server Dashboard

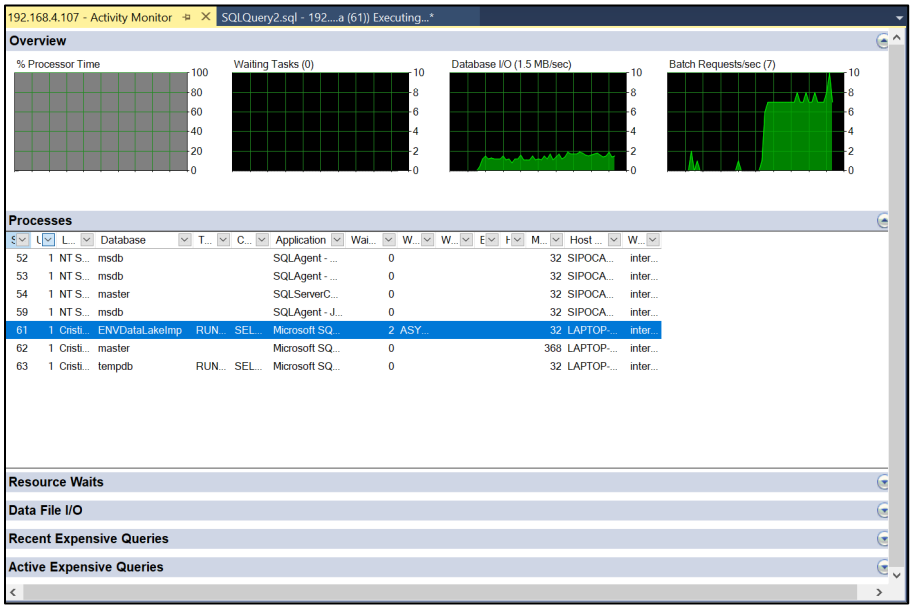
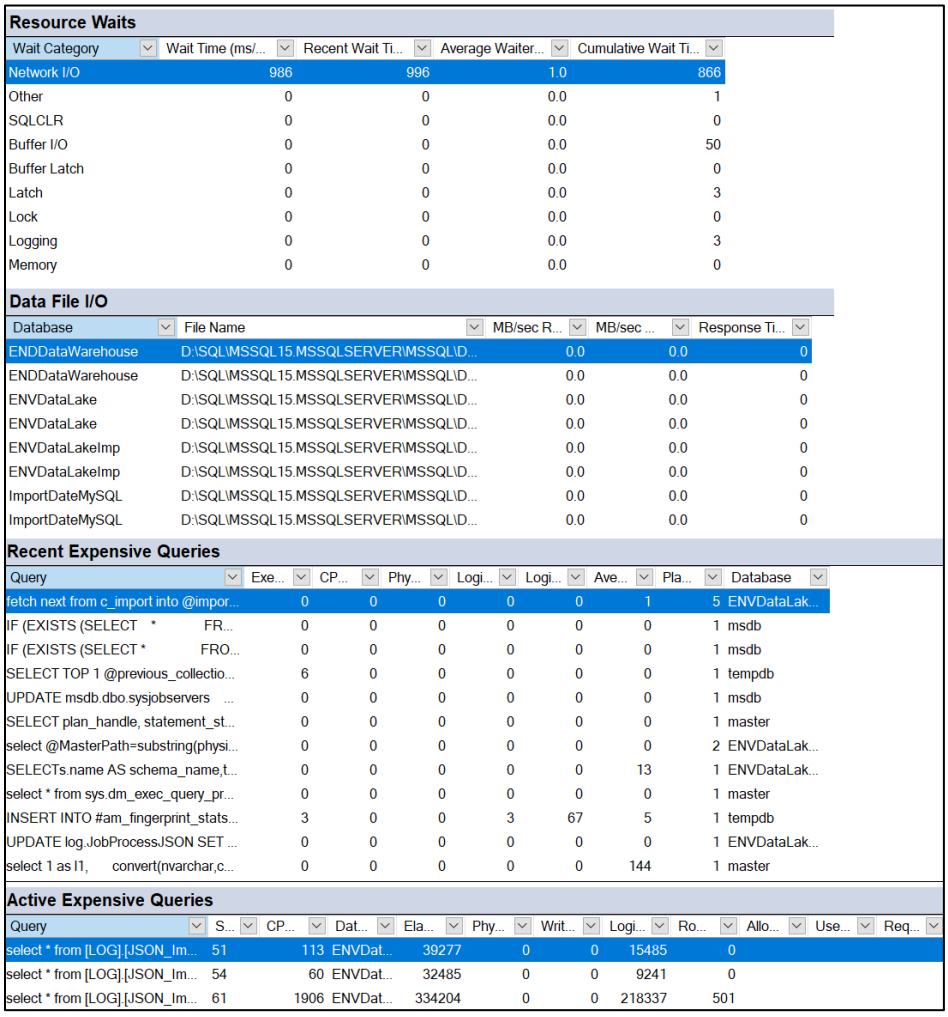


Figura 61 Monitorizarea serverului de date folosind SQL Server Activity Monitor

Un utilitar SQL Server extrem de util este Activity Monitor, care permite monitorizarea în timp real a proceselor ce rulează pe server, furnizând totodată informații detaliate privind timpii de așteptare pentru accesul la resurse (Resources Waits), operațiile I/O la nivelul fișierelor de date (Data File I/O), precum și Interogările recente sau active (Recent/Active Expensive Queries), care reclamă un consum ridicat de resurse (memorie, procesor, activitate pe disc).



Resource Waits

Wait Category	Wait Time (ms/...	Recent Wait Ti...	Average Waiter...	Cumulative Wait Ti...
Network I/O	986	996	1.0	866
Other	0	0	0.0	1
SQLCLR	0	0	0.0	0
Buffer I/O	0	0	0.0	50
Buffer Latch	0	0	0.0	0
Latch	0	0	0.0	3
Lock	0	0	0.0	0
Logging	0	0	0.0	3
Memory	0	0	0.0	0

Data File I/O

Database	File Name	MB/sec R...	MB/sec ...	Response Ti...
ENDDataWarehouse	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ENDDataWarehouse	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ENVDataLake	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ENVDataLake	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ENVDataLakeImp	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ENVDataLakeImp	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ImportDateMySQL	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0
ImportDateMySQL	D:\SQL\MSSQL15.MSSQLSERVER\MSSQL\D...	0.0	0.0	0

Recent Expensive Queries

Query	Exe...	CP...	Phy...	Logi...	Logi...	Ave...	Pla...	Database
fetch next from c_import into @impor...	0	0	0	0	0	1	5	ENVDataLak...
IF (EXISTS (SELECT * FR...	0	0	0	0	0	0	1	msdb
IF (EXISTS (SELECT * FRO...	0	0	0	0	0	0	1	msdb
SELECT TOP 1 @previous_collectio...	6	0	0	0	0	0	1	tempdb
UPDATE msdb.dbo.sysjobservers ...	0	0	0	0	0	0	1	msdb
SELECT plan_handle, statement_st...	0	0	0	0	0	0	1	master
select @MasterPath=substring(physi...	0	0	0	0	0	0	2	ENVDataLak...
SELECTs.name AS schema_name,t...	0	0	0	0	0	13	1	ENVDataLak...
select * from sys.dm_exec_query_pr...	0	0	0	0	0	0	1	master
INSERT INTO #am_fingerprint_stats...	3	0	0	3	67	5	1	tempdb
UPDATE log.JobProcessJSON SET ...	0	0	0	0	0	0	1	ENVDataLak...
select 1 as l1, convert(nvarchar,c...	0	0	0	0	0	144	1	master

Active Expensive Queries

Query	S...	CP...	Dat...	Ela...	Phy...	Writ...	Logi...	Ro...	Allo...	Use...	Req...
select * from [LOG].[JSON_Im...	51	113	ENVDat...	39277	0	0	15485	0			
select * from [LOG].[JSON_Im...	54	60	ENVDat...	32485	0	0	9241	0			
select * from [LOG].[JSON_Im...	61	1906	ENVDat...	334204	0	0	218337	501			

Figura 62 Detalii privind activitatea serverului de date, furnizate de SQL Server Activity Monitor

În cazul fiecărei interogări care solicită intens resursele serverului, poate fi analizat planul de execuție, astfel încât să poată fi detectate operațiile sau prelucrările problematice din perspectiva performanței.

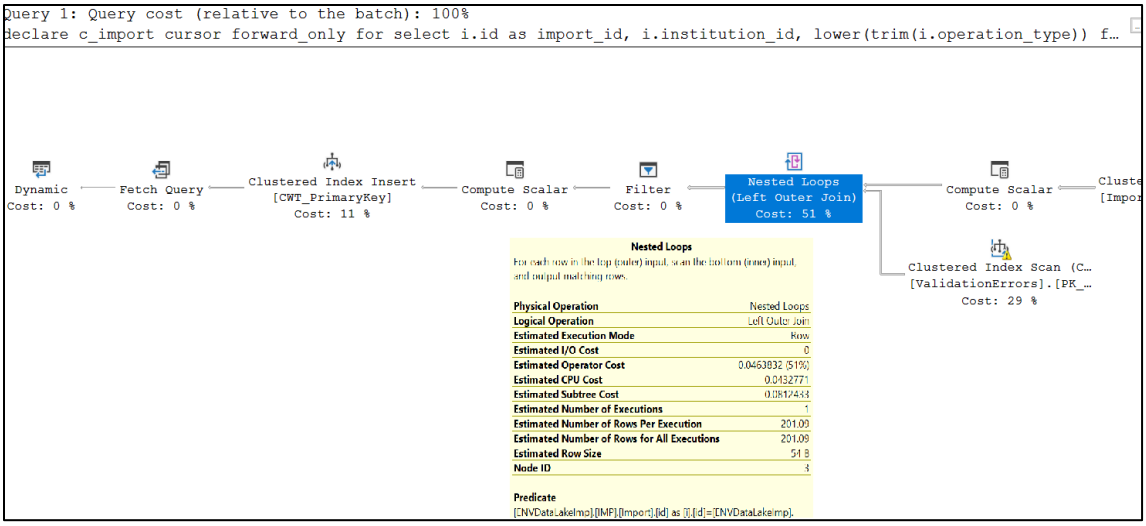


Figura 63 Planul de execuție al unei interogări, accesat prin intermediul SQL Server Activity Monitor

Instrumentele care vizează serverul de date sunt completate de statisticile aferente bazelor de date individuale și de rapoarte care permit monitorizarea evenimentelor și activității de la nivelul acestora: tranzacțiile, utilizarea indecșilor, realizarea copiilor de siguranță și restaurarea bazelor de date, utilizarea discului etc.

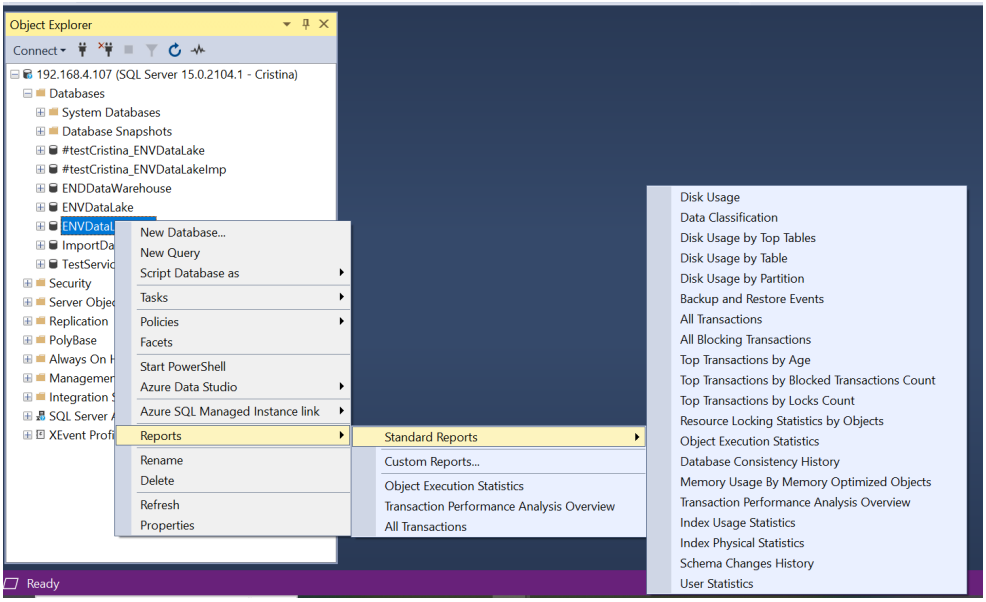


Figura 64 Monitorizarea bazelor de date - rapoarte predefinite disponibile în SQL Server Management Studio

În completarea instrumentelor menționate mai sus, un rol important în procesul de monitorizare al activității la nivelul serverului și bazelor de date revine vederilor și funcțiilor SQL Server de tipul Dynamic Management Views (DMV), respectiv Dynamic Management Functions (DMF), lista completă a acestora fiind obținută prin următoarea interogare:

```
SELECT name, type, type_desc
FROM sys.system_objects so
WHERE so.name LIKE 'dm_%'
ORDER BY so.name;
```

Din ansamblul DMV și DMF, cele mai frecvent utilizate vizează următoarele aspecte:

- Cererile executate pe server:
  - *sys.dm_exec_connections* – Conexiunile la server
  - *sys.dm_exec_sessions* – Sesiunile autentificate
  - *sys.dm_exec_requests* – Cererile curente de executat
- Execuția interogărilor:
  - *sys.dm_exec_cached_plans* – Planurile de execuție stocate în cache
  - *sys.dm_exec_query_stats* – Statisticile de performanță
  - *sys.dm_exec_sql_text* – Codul SQL aferent interogărilor din cache
- Indecșii:
  - *sys.dm_db_index_physical_stats* – Dimensiunea și fragmentarea indecșilor
  - *sys.dm_db_index_usage_stats* – Utilizarea indecșilor pentru optimizarea interogărilor
  - *sys.dm_db_missing_index_details* – Identificarea indecșilor lipsă
- Platforma pe care rulează serverul de date (sistemul de operare - OS):
  - *sys.dm_os_performance_counters* – Lista contoarelor și valorilor de performanță SQL Server
  - *sys.dm_os_schedulers* – Gradul de utilizare al CPU
  - *sys.dm_os_waiting_tasks* – Sarcinile aflate în așteptarea resurselor necesare execuției
  - *sys.dm_os_wait_stats* – Informații și statistici privind toate tipurile de așteptări
- Operațiile de scriere și de citire (intrare - ieșire I/O):
  - *sys.dm_io_virtual_file_stats* – Statisticile I/O care vizează fișierele de date și cele de tip log
  - *sys.dm_io_pending_io_requests* – Solicitările I/O aflate în așteptare

În afara aspectelor care vizează performanța, mediul Microsoft SQL Server oferă instrumente pentru urmărirea tuturor activităților care au loc pe serverul de date, cu scopul detectării unor potențiale amenințări și vulnerabilități; aceste aspecte sunt gestionate în contextul auditului SQL Server. Pe lângă monitorizarea și raportarea modificărilor critice efectuate pe server, SQL Server Audit oferă o imagine cuprinzătoare a informațiilor de securitate SQL Server, orientând acțiunile de securizare a bazelor de date și a serverului. Auditul SQL Server permite monitorizarea și înregistrarea modificărilor aduse setărilor serverului; în plus, pot fi urmărite toate activitățile serverului, până la nivelul acțiunilor concrete realizate de utilizatori asupra datelor dintr-un anumit tabel al unei baze de date. Aceste informații sunt apoi înregistrate după caz, în jurnalul de securitate Windows, jurnalul aplicației sau într-un fișier extern, fiind astfel posibilă identificarea și analiza tuturor evenimentelor suspecte.

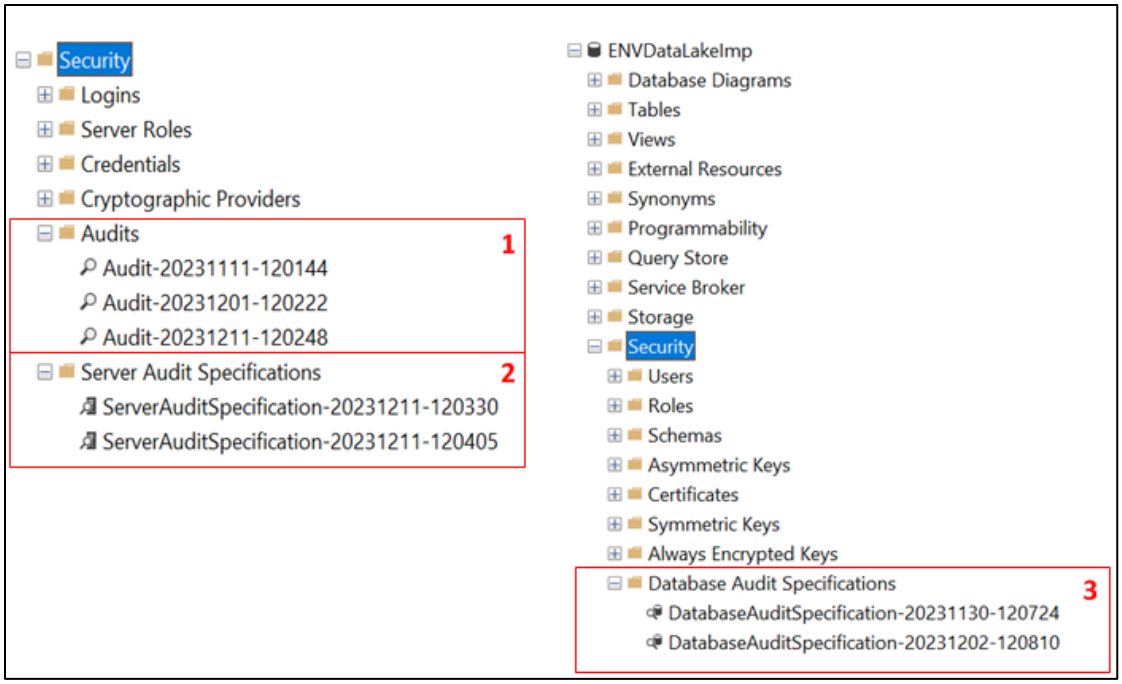


Figura 65 Auditul SQL Server, în funcție de aspectele vizate:
realizarea efectivă a auditului (1), instanța SQL Server (2) și bazele de date (3)

Acțiunile de audit pot fi definite utilizând SQL Server Management Studio, Transact-SQL sau SQL Server Management Objects (SMO). În funcție de natura evenimentelor vizate, auditul SQL Server se poate desfășura pe trei niveluri, cărora le corespund diferite acțiuni, organizate în grupuri:

- Acțiuni la nivel de server (SUCCESSFUL_LOGIN_GROUP, FAILED_LOGIN_GROUP, DBCC_GROUP etc.);
- Acțiuni la nivel de bază de date (DATABASE_OPERATION_GROUP, DATABASE_PRINCIPAL_CHANGE_GROUP etc.) – în acest caz, auditul poate viza aspecte foarte specifice legate de acțiunile pe care utilizatorii le realizează asupra obiectelor unei baze de date;
- Acțiuni care vizează însăși desfășurarea procesului de audit SQL Server (AUDIT_CHANGE_GROUP).

Dacă informațiile de audit sunt înregistrate într-un fișier extern, acestea sunt memorate în format binar, pentru accesarea lor fiind disponibilă funcția tabelară fn_get_audit_file(), cu parametri pentru localizarea fișierului sau fișierelor vizate, ca și a setului datelor ce trebuie returnate:

```
SELECT * FROM fn_get_audit_file('D:\SQLAudits\*', default, default)
```

Informațiile de audit înregistrate la nivelul sistemului (Windows Security Log, respectiv Application Log) pot fi accesate prin intermediul instrumentului Windows Event Viewer. De asemenea, sunt disponibile și în cadrul SQL Server Management Studio (nodul Audit din secțiunea Security).

6 Mentenanța sistemului informatic pentru colectarea datelor prin programul PPCA

6.1 Tipuri de mentenanță aplicabile sistemului informatic

Mentenanța sistemului informatic este esențială pentru a asigura funcționarea eficientă, securitatea și performanța sistemului în timp. Există mai multe tipuri de mentenanță aplicabile sistemelor informatice, iar acestea pot varia în funcție de nevoile specifice ale organizației sau a utilizatorului. Dintre tipurile de mentenanță aplicabile amintim:

- Mentenanță corectivă (reparativă): Acest tip de mentenanță intervine după ce a apărut o problemă sau o defecțiune în sistem. Scopul este de a remedia problema și de a readuce sistemul la starea funcțională normală.
- Mentenanță preventivă: Se efectuează în mod regulat pentru a preveni apariția problemelor sau a defecțiunilor. Include activități precum verificarea și curățarea hardware-ului, actualizări de software, scanări antivirus, backup-uri regulate și alte măsuri care previn degradarea performanței sistemului.
- Mentenanță planificată: Aceasta implică activități planificate în avans, cum ar fi actualizările de sistem, instalarea de patch-uri, optimizarea performanței și alte activități de întreținere care sunt programate pentru a minimiza impactul asupra utilizatorilor.
- Mentenanță neplanificată: Acest tip de mentenanță are loc în situații de urgență, cum ar fi o cădere a sistemului sau o problemă critică care necesită intervenție imediată.
- Mentenanța hardware: Include activități precum curățarea și îngrijirea hardware-ului, înlocuirea componentelor defecte sau învechite și monitorizarea stării hardware-ului pentru a preveni defecțiunile.
- Mentenanța software: Se referă la gestionarea actualizărilor de software, instalarea de patch-uri de securitate, actualizarea aplicațiilor și sistemele de operare pentru a asigura funcționarea cu cele mai recente funcționalități și pentru a remedia eventuale vulnerabilități de securitate.
- Mentenanța rețelei: Includ activități precum monitorizarea traficului de rețea, optimizarea performanței rețelei, gestionarea securității rețelei și implementarea actualizărilor de firmware pentru echipamentele de rețea.
- Mentenanța bazelor de date: Se concentrează pe optimizarea performanței bazelor de date, backup-uri regulate, monitorizarea integrității datelor și actualizarea sistemelor de gestionare a bazelor de date.
- Mentenanța de securitate: Acest tip de mentenanță are ca scop asigurarea securității sistemului, inclusiv implementarea de soluții antivirus, firewalls, actualizări de securitate și auditarea sistemului pentru a identifica și remedia eventuale vulnerabilități.
- Mentenanța de mediu: Se referă la asigurarea condițiilor optime de funcționare pentru echipamentele IT, cum ar fi controlul temperaturii, gestionarea umidității și asigurarea alimentării cu energie electrică stabilă.

Fiecare tip de mentenanță joacă un rol crucial în menținerea sănătății și performanței unui sistem informatic. Planificarea și implementarea unui program de mentenanță bine structurat pot ajuta la prevenirea problemelor majore și la menținerea eficienței operaționale a sistemului pe termen lung.

6.2 Politici de update al componentelor sistemului informatic (versionare, publicare, etc.)

Elaborarea și implementarea unei politici eficiente de actualizare a componentelor sistemului informatic este crucială pentru asigurarea securității, stabilității și performanței sistemului. În continuare sun prezentate câteva aspecte importante de care politicile de actualizare a componentelelor sitemului informatic trebuie să țină seama:

- Calendar de actualizări. Se recomandă stabilirea unui calendar regulat pentru actualizările sistemului, inclusiv actualizări de securitate, patch-uri și upgrade-uri de software. În cadrul acestui calendar ar trebui să se dea prioritate actualizărilor critice și celor legate de securitate pentru a le implementa în cel mai scurt timp posibil.
- Versionare. Este importantă adoptarea unei politici clare de versionare pentru toate componentele sistemului, inclusiv sistemul de operare, aplicațiile și orice alt software folosit. Se recomandă stabilirea unui un proces de gestionare a versiunilor și un sistem de urmărire a modificărilor pentru a monitoriza actualizările și upgrade-urile.
- Testare prealabilă. Se recomandă implementarea unui mediu de testare pentru a evalua impactul actualizărilor înainte de implementarea lor pe sistemele de producție precum și interacțiunea dintre diferite componente ale sistemului pentru a evita conflictele și problemele de compatibilitate.
- Backup-uri și restaurare. Este obligatorie realizarea de backup-uri complete înainte de a implementa actualizări majore. Este importantă asigurarea că procesele de restaurare sunt testate și funcționale pentru a putea reveni rapid la o versiune anterioară în caz de probleme.
- Securitate. Trebuie acordată o atenție deosebită actualizărilor de securitate și implementarea în cel mai scurt timp posibil. De asemenea se recomandă verificarea periodică și analiza politicilor de securitate pentru a le adapta la schimbările de amenințări.
- Comunicare internă. Se recomandă crearea unui proces de comunicare internă pentru a anunța toate actualizările și modificările planificate. Toate departamentele sau echipele afectate asupra schimbărilor programate și a impactului acestora trebuie informate din timp.
- Documentare. Trebuie să existe o documentația actualizată cu privire la toate actualizările, versiunile de software și modificările aduse sistemului.
- Monitorizare și raportare. Se recomandă implementarea unui sistem de monitorizare continuă pentru a detecta orice anomalii sau probleme după implementarea actualizărilor.
- Formare și conștientizare. Echipa tehnică trebuie să urmeze activități de formare cu privire la procesele de actualizare și noile funcționalități introduse. Întreaga echipă trebuie să fie conștientă de importanța actualizărilor și de procedurile asociate.
- Evaluare post-implementare. Trebuie evaluată performanța sistemului și feedback-ul utilizatorilor după implementarea actualizărilor.

Implementarea unei politici bine structurate de actualizare a componentelor sistemului informatic contribuie la menținerea unui mediu de lucru eficient, securizat și actualizat.

6.3 Mentenanța mediului de lucru (politici de update pentru sistemele de operare, sistemele de gestiune ale bazelor de date, certificatele utilizate etc.)

Implementarea unor politici de actualizare corespunzătoare pe un sistem de operare Windows Server este esențială pentru menținerea securității, stabilității și performanței serverului. În continuare sunt prezentate câteva politici și recomandări pentru gestionarea actualizărilor pe un server Windows:

- Automatizare: Se recomandă utilizarea serviciului Windows Update pentru a automatiza procesul de actualizare. Acesta poate fi configurat să descarce și să instaleze automat actualizările critice și de securitate.
- Planificare actualizări: Se recomandă configurarea unui program de actualizare regulat, astfel încât să nu afecteze perioadele critice de utilizare a serverului. De exemplu, se pot programa actualizările să se efectueze în weekend sau în orele de noapte.
- Actualizări selectate: În situația în care se dorește un control mai mare asupra serverului se poate opta pentru efectuarea de actualizări manuale. Această abordare poate fi utilă în medii în care trebuie să se testeze mai întâi actualizările într-un mediu de pre-producție.
- Grupuri de actualizări: Se poate opta pentru împărțirea actualizărilor în grupuri pentru a le gestiona mai eficient. De exemplu, se poate crea un grup pentru actualizări de securitate și un altul pentru actualizări non-securitate.
- Setarea de politici de respingere: În cazuri excepționale se pot configura politici pentru a amâna sau respinge anumite actualizări. Acest lucru poate fi important atunci când se suspectează faptul că anumite actualizări pot afecta anumite aplicații sau configurații.
- Instrumente de monitorizare: Se recomandă utilizarea de instrumente de monitorizare pentru a ține evidența stării actualizărilor și pentru a fi alertat în cazul unor probleme. Microsoft oferă instrumente precum Windows Server Update Services (WSUS) pentru a gestiona și monitoriza actualizările.
- Backup înainte de actualizări majore: Se recomandă un backup complet al sistemului pentru restaurarea acestuia în cazul unor probleme.
- Testare în mediu de pre-producție: Se recomandă testarea actualizărilor într-un mediu de pre-producție pentru a obține asigurări că acestea nu vor afecta negativ funcționarea aplicațiilor critice sau configurațiile serverului.

Mentenanța serverului web IIS (Internet Information Services) este o activitate esențială pentru asigurarea funcționării corecte și securizate a serviciului web găzduit. Cele mai importante aspecte care ar trebui luate în considerare în timpul mentenanței unui server web IIS sunt:

- Actualizări de securitate ale sistemulului de operare, serverul web și toate componentele auxiliare
- Securitatea aplicațiilor web. Aceasta se realizează prin monitorizarea regulată a vulnerabilităților aplicațiilor web gazduite și actualizarea acestora la versiunile cele mai recente.
- Implementarea și actualizeazarea regulilor de filtrare a traficului web pentru a proteja împotriva atacurilor.

- Monitorizarea performanței. Aceasta se realizează prin instrumente de monitorizare care permit urmărirea performanței serverului și a aplicațiilor.
- Verifică logurile pentru a depista orice anomalii și pentru a lua măsuri preventive.
- Backup-uri regulate ale aplicațiilor web, bazelor de date și setărilor de server. De asemenea trebuie luate măsuri prin care să se asigure faptul că procedurile de restaurare a datelor sunt testate și funcționale.
- Gestionarea resurselor. Se recomandă monitorizează utilizării resurselor, cum ar fi CPU, memorie și spațiu pe disc.
- Optimizarea setărilor serverului în funcție de nevoile aplicațiilor web gazduite.
- Firewall și reguli de securitate. Se recomandă configurarea unui firewall pentru a controla traficul către și dinspre server.
- Jurnale și auditare. Se recomandă activarea jurnalelor de securitate și de acces pentru a monitoriza activitățile precum și examinează periodică a jurnalelor pentru a detecta orice activitate suspectă.

În vederea asigurării securității transportului mesajelor transmise între client și serverul web (IIS-Internet Information Services) s-a utilizat un certificat de tip SSL (Secure Socket Layer).

Mentenanța certificatelor SSL pe un server IIS (Internet Information Services) este importantă pentru menținerea securității și bunei funcționări a conexiunilor securizate către serverul web. Mentenanța certificatului SSL instalat pe serverul Web presupune:

- Monitorizare expirare certificat. În acest scop se pot configura notificări care să fie activate înainte de expirarea certificatului. Acest lucru poate fi realizat cu ajutorul unor servicii sau scripturi care verifică periodic data de expirare a certificatului.
- Actualizare și reînnoire certificat. Dacă autoritatea care a emis certificatul permite acest lucru, se recomandă configurarea unui proces de reînnoire automată a certificatului înainte de expirare. Dacă reînnoirea automată nu este posibilă, certificatul trebuie reînnoit manual înainte de expirare.
- Verificare periodică a configurației SSL. Se recomandă verificarea configurării SSL pentru asigurarea concordanței cu cele mai bune practici de securitate și cu schimbările de reglementare.
- Backup și recuperare. Se recomandă crearea de copii de siguranță la intervale regulateale certificatelor și cheilor private. Acest lucru vă permite recuperarea rapidă în caz de pierdere sau corupție a datelor.
- Actualizarea cheilor criptografice. Se recomandă actualizarea periodică a algoritmilor și lungimilor cheilor criptografice conform standardelor de securitate curente.
- Audit și conformitate: Este important să se realizeze audituri periodice în vedere astfel încât să existe serverul IIS respectă standardele de securitate recomandate.

Prin aplicarea acestor practici, se menține un mediu sigur și actualizat pentru conexiunile securizate pe serverul IIS. Pe baza unui plan bine stabilit pentru gestionarea și actualizarea certificatelor SSL se vor evita problemele de securitate și se va menține funcționalitatea corectă a serviciilor web.

7 Direcții viitoare de integrare a sistemului informatic pentru colectarea și analiza datelor privind deșeurile prin programul PPCA cu alte sisteme informatice

Deși sistemul informatic pentru colectarea și analiza datelor privind deșeurile prin programul PPCA a fost dezvoltat având în vedere principiul funcționării autonome, independent de alte sisteme din cadrul Ministerului Mediului, integrarea sa în peisajul IT existent poate aduce beneficii vizibile în sensul unei analize centralizate a datelor colectate, sau chiar a unui schimb de informații automatizat sau la cerere. Acest obiectiv ar putea fi urmărit în viitor, necesitând dezvoltări software în direcția implementării unei comunicări permanente cu alte sisteme și platforme informatice, în mod sincron sau asincron, facilitând schimbul de date între soluțiile existente folosite de Ministerul Mediului și sistemul informatic pentru colectarea și analiza datelor privind deșeurile prin programul PPCA.

Pot fi analizate, de asemenea, oportunități de integrare cu sisteme informatice externe, de accesare a unor baze de date de tip open-data, a unor nomenclatoare de date publice, etc. Totuși, din această perspectivă, ar putea fi necesară încheierea unor parteneriate inter-instituționale privind schimbul de date și stabilirea unor protocoale de transfer standardizat de informații într-o manieră sigură și stabilă în timp.

Dacă în unele scenarii sistemele informatice pot comunica în timp real, în special sistemele ale căror dependențe sunt motivate de necesitatea de a executa procesele operaționale, în alte scenarii va fi suficientă comunicarea asincronă prin transferuri de date la anumite intervale de timp. În acest context, sistemul informatic pentru programul PPCA poate juca atât rol de furnizor de date, cât și de consumator. Depozitul de date dezvoltat ca element central în colectarea și modelarea multidimensională a datelor ar putea fi extins spre a putea fi populat și cu alte date provenind din surse informaționale primite de la alte sisteme existente. Tabloul de bord dezvoltat ar putea fi utilizat în analize de date complexe pe baza unor surse informaționale provenind de la alte sisteme informatice. Pe de altă parte, sistemul informatic suport pentru programul PPCA ar putea să exporte informații printr-un schimb sincron sau asincron de mesaje stabilit cu alte sisteme informatice suport. Aceasta ar presupune, din punct de vedere tehnic, dezvoltări software viitoare de soluții de tip servicii web pentru integrări de tip punct la punct (point-to-point) sau de interfețe API pentru integrări de tip multicerere.

Luând în considerare proliferarea tehnologică, complexitatea crescută a domeniului IT și nevoia de integrare a informațiilor din domeniul administrației publice, în general, este de așteptat ca în viitor un număr tot mai mare de interacțiuni cu sistemul informatic pentru programul PPCA să fie necesare. Pe măsură ce mai multe surse vor extrage informații din sistemul pentru programul PPCA, integrarea punct la punct nu va mai fi eficientă. În acest context se recomandă folosirea interfețelor software-to-software (API) dezvoltate o singură dată și care pot fi utilizate pentru orice nevoie de integrare cu alte sisteme informatice sau aplicații software.

La integrarea sistemelor informatice trebuie să se aibă în vedere infrastructura hardware și de rețea existentă în cadrul Ministerului Mediului**.** Pentru respectarea politicilor de securitate definite la nivelul Ministerului privind protejarea datelor manipulate de sistemul informatic pentru programul PPCA, ar putea fi necesar ca diferite componente web care să faciliteze integrarea cu alte sisteme sau baze de

date să fie instalate în zone de acces public (precum DMZ) diferite de componenta *back-end* a sistemului informatic folosit pentru colectarea și analiza datelor privind deșeurile. Aceasta ar putea afecta modalitatea de integrare a sistemelor, indiferent ce componente software sunt implicate în procesul de integrare.

8 Implementarea unui sistem de tip help-desk pentru utilizatorii sistemului informatic

Deși instruirea utilizatorului final în vederea utilizării eficiente a platformei software pentru programul PPCA este obligatorie, studiile au demonstrat că satisfacția utilizatorilor crește proporțional cu suportul acordat la problemele pe care le întâmpină aceștia în utilizarea sistemelor informatice (Sheehan, 2008).

Help desk este un centru de asistență pentru clienții dintr-o organizație care oferă informații, asistență administrativă și tehnică utilizatorilor, cu scopul de a rezolva problemele pe care aceștia le-au întâlnit în timpul utilizării resurselor sau facilităților organizației (Masongsong, 2016). Un sistem de tip help-desk automatizat permite companiilor să accepte, să urmărească și să răspundă la solicitările de asistență într-un mod organizat. Multe programe de tip help-desk oferă, de asemenea, baze de cunoștințe (knowledge management), portaluri de autoservire (self-service), management SLO (Service Level Objective) și raportare.

Un sistem de emitere de tichete (*ticketing*) ar contribui la eficientizarea procesului de raportare a erorilor și de depanare, în special atunci când se interacționează cu utilizatorii finali. Ar trebui să existe o echipă de specialiști în dezvoltarea de software și de analiști de date care să se ocupe de gestionarea problemelor tehnice apărute fie din cauza unor erori în utilizarea sistemelor informatice, fie din cauza neînțelegerii funcționării corecte a acestora. Aceste solicitări trebuie rezolvate în funcție de prioritățile și de ordinea în care au fost semnalate. Astfel, un sistem centralizat de tip *ticketing* gestionat de specialiști IT din Ministerul Mediului ar ajuta la gestionarea în mod corespunzător a priorității în remedierea erorilor respective. În acest scop, beneficiarul ar putea lua în considerare utilizarea unor soluții de tip *ticketing* de tip COTS sau *open-source*, bazate pe cloud sau găzduite “on-premise”.

Deși o soluție la cheie (COTS) asigură o securitate și eficiență maximă, precum și un suport din partea dezvoltatorului, totuși putem observa că există, în prezent, pe piață mai multe sisteme informatice de tip ticketing de tip *open-source* care ar putea fi adoptate în gestionarea problemelor de operaționalizare a platformei PPCA (cu titlu de exemplu, fără a recomanda vreo opțiune aparte, am putea enunța aici soluții informatice precum Altera, osTicket, Uvdesk, Faveo, FreeScout, etc.). Fiecare dintre acestea posedă o listă de funcționalități care ar trebui analizate comparativ pentru a decide care va fi opțiunea aleasă. Printre principalele funcționalități necesare ar trebui să fie incluse ergonomia interfeței cu utilizatorul, posibilitatea emiterii de tichete de suport pe grupuri de acțiuni, posibilitatea de a stabili priorități în rezolvarea problemelor raportate prin tichetele emise de utilizatori, exportul datelor în fișiere de tip .CSV,. json, .xml, .pdf, etc., alocarea de responsabili pentru problemele raportate, etc.

Bibliografie selectivă

D. Sampson and M. M. Chowdhury. (2021). The Growing Security Concerns of Cloud Computing. *IEEE International Conference on Electro Information Technology (EIT)*, 50-55.

Hashim, A., Farooq, K., & Piatti-Fünfkirchen, M. (2020). *ENSURING BETTER PFM OUTCOMES WITH FMIS INVESTMENTS.* Washington, DC: World Bank Group.

Masongsong, R. P. (2016). Help Desk Management System. *Proceedings of the World Congress on Engineering and Computer Science*, *1.* San Francisco, USA.

Sheehan, M. (2008). The Help Desk as a Pivot Point for IT Agility. *2007 EDUCAUSE Midwest Regional Conference.* Chicago, Illinois.

Techopedia. (2023). *Internet Information Services Certificate*. Preluat pe 11 29, 2023, de pe Internet Information Services Certificate - IIS Cerificates: https://www.techopedia.com/definition/29765/internet-information-services-certificate-iis-certificate#:~:text=An%20IIS%20certificate%20is%20any,running%20Microsoft%20Windows%20operating%20systems.

ANEXA 1. Scriptul SQL pentru inițializarea bazei de date de tip DataLake

```
USE [ENVDataLakeImp]
GO
/****** Object:  Schema [IMP]    Script Date: 12/1/2023 4:10:56 PM ******/
CREATE SCHEMA [IMP]
GO
/****** Object:  Schema [LOG]    Script Date: 12/1/2023 4:10:56 PM ******/
CREATE SCHEMA [LOG]
GO
/****** Object:  Schema [medias]    Script Date: 12/1/2023 4:10:56 PM ******/
CREATE SCHEMA [medias]
GO
/****** Object:  Schema [shared]    Script Date: 12/1/2023 4:10:56 PM ******/
CREATE SCHEMA [shared]
GO
/****** Object:  Schema [Temp]    Script Date: 12/1/2023 4:10:56 PM ******/
CREATE SCHEMA [Temp]
GO
/****** Object:  UserDefinedFunction [dbo].[f_CatVerificErori]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
create function [dbo].[f_CatVerificErori] (@json nvarchar(max))
returns nvarchar(max)
as
```

```
	begin
	declare @errors nvarchar(max)=''
	exec [dbo].[spCris_#ValidareStructuraJSON] @json, @errors out
	return @errors
	end
	GO
	/****** Object:  UserDefinedFunction [dbo].[fCris_#SirStructuraTabel]    Script Date: 12/1/2023
4:10:56 PM ******/
	SET ANSI_NULLS ON
	GO
	SET QUOTED_IDENTIFIER ON
	GO
	CREATE function [dbo].[fCris_#SirStructuraTabel](
		@numeTabel sysname,
				@tipStructuraOut varchar(100)
	) returns nvarchar(max)

	as
	begin

	declare @tmp_campuriTabel table (
	   NumeCamp nvarchar(100),
		IdCamp int,
		TipCamp nvarchar(100),
		NrCaractere nvarchar(30),
		AdmiteNull bit, --nvarchar(20)
		NumeCamp_StructuraTabel nvarchar(100),
	   NumeCamp_StructuraJSON nvarchar(100)
	)
```

```
insert @tmp_campuriTabel
select *
from dbo.fCris_#StructuraTabel(@numeTabel)

declare @testIdCamp int
select @testIdCamp=IdCamp
from @tmp_campuriTabel

if @testIdCamp is null
  return null

if  @tipStructuraOut = 'structuraJson'
begin
        declare @structuraJson nvarchar(max) =''
        select @structuraJson = @structuraJson + NumeCamp_StructuraJSON + ','
        from @tmp_campuriTabel
        order by IdCamp
        SET @structuraJson = SUBSTRING(@structuraJson, 0, LEN(@structuraJson))
        --select @structuraJson as 'StructuraJSON'
        return @structuraJson
        --select  @structuraJson
end

if  @tipStructuraOut = 'structuraTabel'
begin
        declare @structuraTabel nvarchar(max)=''
        select @structuraTabel = @structuraTabel + QUOTENAME(NumeCamp) + ','
        from @tmp_campuriTabel
        order by IdCamp
```

```
		SET @structuraTabel = SUBSTRING(@structuraTabel, 0, LEN(@structuraTabel))
		--select @structuraTabel  as 'StructuraTabel'
	   return @structuraTabel
	end

	if  @tipStructuraOut = 'structuraTabel_tmp'
	begin
		declare @structuraTabel_tmp nvarchar(max)=''
		select @structuraTabel_tmp = @structuraTabel_tmp + ' ' + NumeCamp_StructuraTabel + ','
		from @tmp_campuriTabel
		order by IdCamp
		SET @structuraTabel_tmp = SUBSTRING(@structuraTabel_tmp, 0, LEN(@structuraTabel_tmp))
		return @structuraTabel_tmp
	end


	if  @tipStructuraOut = 'structuraTabel_min'
	begin
		declare @structuraTabelMin nvarchar(max)=''
		select @structuraTabelMin = @structuraTabelMin + QUOTENAME(NumeCamp) + ','
		from @tmp_campuriTabel
		where AdmiteNull=0
		order by IdCamp
		if @structuraTabelMin=''
		   set @structuraTabelMin=null
		else
			SET @structuraTabelMin = SUBSTRING(@structuraTabelMin, 0, LEN(@structuraTabelMin))
```

```
    return @structuraTabelMin
end

-- else:
return null

end
GO
/****** Object:  UserDefinedFunction [dbo].[fCris_#StructuraTabel]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE function [dbo].[fCris_#StructuraTabel](
        @numeTabel sysname
) returns @tmp_structuraTabel table (
    NumeCamp nvarchar(100),
        IdCamp int,
        TipCamp nvarchar(100),
        NrCaractere nvarchar(30),
        AdmiteNull bit,  --nvarchar(20)
        NumeCamp_StructuraTabel nvarchar(100),
    NumeCamp_StructuraJSON nvarchar(100)
)

as
begin
```

```
declare @tmp_campuriTabel table (
    NumeCamp nvarchar(100),
        IdCamp int,
        TipCamp nvarchar(100),
        NrCaractere nvarchar(30),
        AdmiteNull bit --nvarchar(20)
)
insert @tmp_campuriTabel
select
            col.name as NumeCamp,
            col.column_id as IdCamp,
            typ.name  as TipCamp,
            case typ.name
                when 'char' then '(' + cast(col.max_length as varchar(10))+ ')'
                when 'nchar' then '(' + cast(col.max_length as varchar(10))+ ')'
                when 'nvarchar' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
                when 'varbinary' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
                when 'varchar' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
                else ''
            end as NrCaractere,
                --case when col.is_nullable = 1 then '$.' else 'strict $.' end as Strict
                col.is_nullable as AdmiteNull
        from sys.columns col
            join sys.types typ on
                col.system_type_id = typ.system_type_id AND
                col.user_type_id = typ.user_type_id
        where object_id = object_id(QUOTENAME(@numeTabel))
```

```
            and col.is_identity = 0
            and col.is_computed = 0
            and col.is_hidden = 0
            and col.is_rowguidcol = 0
            and col.generated_always_type = 0

        --select * from  @tmp_campuriTabel


        insert @tmp_structuraTabel (
            NumeCamp,
                IdCamp,
                TipCamp,
                NrCaractere,
                AdmiteNull,
                NumeCamp_StructuraTabel,
            NumeCamp_StructuraJSON
        )
        select NumeCamp, IdCamp, TipCamp, NrCaractere, AdmiteNull,
          QUOTENAME(NumeCamp) + ' ' + TipCamp + NrCaractere  as NumeCamp_StructuraTabel,
          QUOTENAME(NumeCamp) + ' ' + TipCamp + NrCaractere +
                                              ' N''' + case when AdmiteNull = 1 then '$.' else 'strict $.' end + '"'
+ STRING_ESCAPE(NumeCamp, 'json') + '"'''
        from @tmp_campuriTabel



        return

        end
```

```
GO
/****** Object:  UserDefinedFunction [dbo].[fCris_ExtrageDateJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE function [dbo].[fCris_ExtrageDateJSON](@pSirJSON nvarchar(max))
returns @tabelRezultat table (
	IdTabel int,
	NumeTabel nvarchar(100),
	SirJSON nvarchar(Max)
)
as
begin

if ISJSON(@pSirJSON)=0
  return

------------------------
declare @startCautare nvarchar(max) = '$."'
if json_query(@pSirJSON,'$."original"') is not null
   set @startCautare = '$."original"."'



declare cDateJSON cursor forward_only
for
select t.object_id  as IdTabel,
     t.name as NumeTabel
```

```
	from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
	where s.name = 'dbo' and t.name not in (select [table_name] from [IMP].[IgnoredTables])

	declare @idTabel int
	declare @numeTabel nvarchar(100)

	open cDateJSON
	fetch next from cDateJSON into @idTabel, @numeTabel
	while @@FETCH_STATUS=0
	begin
		insert @tabelRezultat (IdTabel,NumeTabel,SirJSON)
		select  @idTabel, @numeTabel, json_query(@pSirJSON,@startCautare + @numeTabel + '"') as Test

		fetch next from cDateJSON  into @idTabel, @numeTabel
	end
	close cDateJSON
	deallocate cDateJSON

	----------------------------
	update @tabelRezultat
	set SirJSON=null
	where REPLACE(SirJSON,' ','')='[]'

	return
	end
	GO
	/****** Object:  UserDefinedFunction [dbo].[fCris3_Test_DateValidare]    Script Date: 12/1/2023 4:10:56 PM ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE function [dbo].[fCris3_Test_DateValidare](
        @numeTabel sysname,
                @tipStructuraOut varchar(100)
) returns nvarchar(max)

as
begin

declare @tmp_campuriTabel table (
   NumeCamp nvarchar(100),
        IdCamp int,
        TipCamp nvarchar(100),
        NrCaractere nvarchar(30),
        Strict nvarchar(20)
)
insert @tmp_campuriTabel
select
                col.name as NumeCamp,
                col.column_id as IdCamp,
                typ.name  as TipCamp,
                case typ.name
                        when 'char' then '(' + cast(col.max_length as varchar(10))+ ')'
                        when 'nchar' then '(' + cast(col.max_length as varchar(10))+ ')'
                        when 'nvarchar' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
```

```
                                when 'varbinary' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
                                when 'varchar' then (IIF(col.max_length=-1, '(MAX)', '(' +
cast(col.max_length as varchar(10))+ ')'))
                                else ''
                        end as NrCaractere,
                                case when col.is_nullable = 1 then '$.' else 'strict $.'
                        end as Strict
                from sys.columns col
                        join sys.types typ on
                                col.system_type_id = typ.system_type_id AND
                                col.user_type_id = typ.user_type_id
                where object_id = object_id(QUOTENAME(@numeTabel))
                and col.is_identity = 0
                and col.is_computed = 0
                and col.is_hidden = 0
                and col.is_rowguidcol = 0
                and col.generated_always_type = 0

        --select * from  @tmp_campuriTabel

        declare @testIdCamp int
        select @testIdCamp=IdCamp
        from @tmp_campuriTabel

        if @testIdCamp is null
          return null

        if  @tipStructuraOut = 'structuraJson'
```

```
begin
	declare @structuraJson nvarchar(max) =''
	select @structuraJson = @structuraJson + '
				' + QUOTENAME(NumeCamp) + ' ' + TipCamp + NrCaractere +
				 ' N''' + Strict + '''' + STRING_ESCAPE(NumeCamp, 'json') + '''''' +
				+ ','
	from @tmp_campuriTabel
	order by IdCamp
	SET @structuraJson = SUBSTRING(@structuraJson, 0, LEN(@structuraJson))
	--select @structuraJson as 'StructuraJSON'
	return @structuraJson
	--select  @structuraJson
end

if  @tipStructuraOut = 'structuraTabel'
begin
	declare @structuraTabel nvarchar(max)=''
	select @structuraTabel = @structuraTabel + QUOTENAME(NumeCamp) + ','
	from @tmp_campuriTabel
	order by IdCamp
	SET @structuraTabel = SUBSTRING(@structuraTabel, 0, LEN(@structuraTabel))
	--select @structuraTabel  as 'StructuraTabel'
   return @structuraTabel
end

if  @tipStructuraOut = 'structuraTabel_tmp'
begin
	declare @structuraTabel_tmp nvarchar(max)=''
	select @structuraTabel_tmp = @structuraTabel_tmp + '
```

```
                                        ' + QUOTENAME(NumeCamp) + ' ' + TipCamp + NrCaractere + ','
            from @tmp_campuriTabel
            order by IdCamp
            SET @structuraTabel_tmp = SUBSTRING(@structuraTabel_tmp, 0,
LEN(@structuraTabel_tmp))
            return @structuraTabel_tmp
        end

        if  @tipStructuraOut = 'structuraTabel_min'
        begin
            declare @structuraTabelMin nvarchar(max)=''
            select @structuraTabelMin = @structuraTabelMin + QUOTENAME(NumeCamp) + ','
            from @tmp_campuriTabel
            where Strict like 'strict%'
            order by IdCamp
            if @structuraTabelMin=''
               set @structuraTabelMin=null
            else
                    SET @structuraTabelMin = SUBSTRING(@structuraTabelMin, 0,
LEN(@structuraTabelMin))
          return @structuraTabelMin
        end

        -- else:
        return null

        end
        GO
        /****** Object:  Table [IMP].[IgnoredTables]    Script Date: 12/1/2023 4:10:56 PM ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [IMP].[IgnoredTables](
	[table_name] [nvarchar](30) NOT NULL,
	[details] [nvarchar](max) NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  View [dbo].[viewCris3_Test_ListaFK]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE view [dbo].[viewCris3_Test_ListaFK]
as
SELECT top 100 percent
  obj.name AS FK,
	tab1.object_id AS [tabelFK_id],
  tab1.name AS [tabelFK],
  col1.name AS [campFK],
  tab2.object_id AS [tabelPK_id],
	tab2.name AS [tabelPK],
  col2.name AS [campPK]
FROM [ENVDataLake].sys.foreign_key_columns fkc  -----
INNER JOIN [ENVDataLake].sys.objects obj
  ON obj.object_id = fkc.constraint_object_id
INNER JOIN [ENVDataLake].sys.tables tab1
```

```
   ON tab1.object_id = fkc.parent_object_id
INNER JOIN [ENVDataLake].sys.columns col1
   ON col1.column_id = parent_column_id AND col1.object_id = tab1.object_id

INNER JOIN [ENVDataLake].sys.tables tab2
   ON tab2.object_id = fkc.referenced_object_id
INNER JOIN [ENVDataLake].sys.columns col2
   ON col2.column_id = referenced_column_id AND col2.object_id = tab2.object_id

LEFT JOIN IMP.IgnoredTables it1 on tab1.name=it1.table_name
LEFT JOIN IMP.IgnoredTables it2 on tab2.name=it2.table_name

WHERE it1.table_name is null and it2.table_name is null

order by 3,6
GO
/****** Object:  View [dbo].[viewCris3_Test_ListaPK]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE view [dbo].[viewCris3_Test_ListaPK]
as
select distinct  t.object_id as IdTabel,  u.TABLE_NAME as TabelPK,u.COLUMN_NAME as CampPK
from ENVDataLake.INFORMATION_SCHEMA.TABLE_CONSTRAINTS c
inner join ENVDataLake.INFORMATION_SCHEMA.KEY_COLUMN_USAGE u
    on (c.CONSTRAINT_NAME=u.CONSTRAINT_NAME
        and c.CONSTRAINT_CATALOG=u.CONSTRAINT_CATALOG
```

```
            and c.CONSTRAINT_SCHEMA=u.CONSTRAINT_SCHEMA)
        inner join ENVDataLake.sys.tables  t on t.name = u.TABLE_NAME
        inner join sys.schemas s on t.schema_id=s.schema_id
        left join imp.IgnoredTables i on  t.name=i.table_name
        where i.table_name is null
            and c.CONSTRAINT_TYPE='PRIMARY KEY' and s.name='dbo'
        GO
        /****** Object:  View [dbo].[viewCris_DependenteTabele]    Script Date: 12/1/2023 4:10:56 PM
******/
        SET ANSI_NULLS ON
        GO
        SET QUOTED_IDENTIFIER ON
        GO

        CREATE view [dbo].[viewCris_DependenteTabele]
        as
        with relatii
        as (
        select t.object_id as tabelFK_id,  t.name as tabelFK, v.campFK, v.tabelPK_id, v.tabelPK
        from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
           left join [IMP].[IgnoredTables] it on t.name=it.table_name
                        left join [dbo].[viewCris3_Test_ListaFK] v on
                          t.object_id=v.tabelFK_id
        where s.name ='dbo' and it.table_name is null
        ----order by tabelFK, v.campFK
        ),
        ierarhii as
        (
        select tabelFK_id as IDParinteNivel, tabelFK as Parinte_nivel,
```

```
        tabelFK_id as IdCopil, tabelFK as Copil, 0 as NivelDependenta
    from  relatii
    union all
    select i.IDParinteNivel,i.Parinte_nivel,
                    r.tabelFK_id, r.tabelFK,NivelDependenta + 1
    from relatii r inner join ierarhii i on
        r.tabelPK_id=i.IdCopil
    ),
    niveluriIerarhii as
    (
    select i.IdCopil as IdTabel, i.Copil as NumeTabel,
        max(i.NivelDependenta) as NivelMaxDependenta
    from ierarhii i
    group by i.IdCopil,i.Copil
    )
    select n.IdTabel,n.NumeTabel,n.NivelMaxDependenta
    from niveluriIerarhii n
    GO
    /****** Object:  View [dbo].[viewCris_IerarhieCompletaTabele]    Script Date: 12/1/2023 4:10:56
PM ******/
    SET ANSI_NULLS ON
    GO
    SET QUOTED_IDENTIFIER ON
    GO

    CREATE view [dbo].[viewCris_IerarhieCompletaTabele]
    as
    with relatii
    as (
```

```
select t.object_id as tabelFK_id,  t.name as tabelFK, v.campFK, v.tabelPK_id, v.tabelPK
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
   left join [IMP].[IgnoredTables] it on t.name=it.table_name
                        left join [dbo].[viewCris3_Test_ListaFK] v on
                         t.object_id=v.tabelFK_id
where s.name ='dbo' and it.table_name is null
----order by tabelFK, v.campFK
)
---select * from relatii
,
ierarhii as
(
select tabelPK_id as IDParinte, tabelPK as tabelParinte,
     tabelFK_id as IDParinteNivel, tabelFK as Parinte_nivel,
     tabelFK_id as IdCopil, tabelFK as tabelCopil, campFK, 0 as NivelDependenta
from  relatii
union all
select tabelPK_id, tabelPK,
      i.IDParinteNivel,i.Parinte_nivel,
                    r.tabelFK_id, r.tabelFK,r.campFK,NivelDependenta + 1
from relatii r inner join ierarhii i on
      r.tabelPK_id=i.IdCopil
)
select * from ierarhii
where NivelDependenta>0
GO
/****** Object:  View [dbo].[viewCris3_Test_ListaPK_Import]    Script Date: 12/1/2023 4:10:56
PM ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE view [dbo].[viewCris3_Test_ListaPK_Import]
as
select distinct  t.object_id as IdTabel,  u.TABLE_NAME as TabelPK,u.COLUMN_NAME as CampPK_Import
from INFORMATION_SCHEMA.TABLE_CONSTRAINTS c
inner join INFORMATION_SCHEMA.KEY_COLUMN_USAGE u
    on (c.CONSTRAINT_NAME=u.CONSTRAINT_NAME
        and c.CONSTRAINT_CATALOG=u.CONSTRAINT_CATALOG
        and c.CONSTRAINT_SCHEMA=u.CONSTRAINT_SCHEMA)
inner join sys.tables  t on t.name = u.TABLE_NAME
inner join sys.schemas s on t.schema_id=s.schema_id
left join imp.IgnoredTables i on  t.name=i.table_name
where i.table_name is null
    and c.CONSTRAINT_TYPE='PRIMARY KEY' and s.name='dbo'
GO
/****** Object:  Table [dbo].[areas]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[areas](
        [branch_id] [bigint] NULL,
        [name] [nvarchar](191) NOT NULL,
        [obs] [nvarchar](191) NULL,
        [created_at] [datetime2](7) NULL,
        [updated_at] [datetime2](7) NULL,
```

```
	[id] [int] NOT NULL,
	[area_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_area_ID] PRIMARY KEY CLUSTERED
(
	[area_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[authorizations]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[authorizations](
	[car_id] [int] NOT NULL,
	[type] [nvarchar](191) NOT NULL,
	[start] [date] NOT NULL,
	[end] [date] NOT NULL,
	[document_number] [nvarchar](191) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[authorization_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_authorization_ID] PRIMARY KEY CLUSTERED
```

```
(
	[authorization_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[bin_collections]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[bin_collections](
	[location_id] [bigint] NULL,
	[tag_id] [bigint] NULL,
	[car_id] [bigint] NULL,
	[tag_code] [nvarchar](191) NOT NULL,
	[license_plate] [nvarchar](191) NOT NULL,
	[tag_type] [nvarchar](191) NOT NULL,
	[tag_volume] [nvarchar](191) NOT NULL,
	[collected] [smallint] NULL,
	[side] [smallint] NOT NULL,
	[load] [nvarchar](191) NULL,
	[car_index] [nvarchar](191) NOT NULL,
	[event] [smallint] NOT NULL,
	[lat] [decimal](10, 8) NULL,
	[lng] [decimal](11, 8) NULL,
	[datetime] [datetime2](7) NULL,
	[created_at] [datetime2](7) NULL,
```

```
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[bin_collection_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_bin_collection_ID] PRIMARY KEY CLUSTERED
(
	[bin_collection_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[bkJsonImport]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[bkJsonImport](
	[id] [bigint] IDENTITY(1,1) NOT NULL,
	[json_string] [nvarchar](max) NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [dbo].[branches]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[branches](
	[name] [nvarchar](191) NOT NULL,
```

```
	[lat] [decimal](10, 8) NULL,
	[lng] [decimal](11, 8) NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[branch_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_branche_ID] PRIMARY KEY CLUSTERED
(
	[branch_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[car_areas]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[car_areas](
	[car_id] [bigint] NULL,
	[area_id] [bigint] NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[car_area_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_area_ID] PRIMARY KEY CLUSTERED
```

```
(
	[car_area_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[car_brands]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[car_brands](
	[name] [nvarchar](191) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[car_brand_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_brand_ID] PRIMARY KEY CLUSTERED
(
	[car_brand_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[car_drivers]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[car_drivers](
	[car_id] [bigint] NULL,
	[driver_id] [bigint] NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[car_driver_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_driver_ID] PRIMARY KEY CLUSTERED
(
	[car_driver_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[car_models]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[car_models](
	[car_brand_id] [bigint] NULL,
	[name] [nvarchar](191) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
```

```
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[car_model_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_model_ID] PRIMARY KEY CLUSTERED
(
	[car_model_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[car_provenances]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[car_provenances](
	[name] [nvarchar](191) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[car_provenance_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_provenance_ID] PRIMARY KEY CLUSTERED
(
	[car_provenance_ID] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [dbo].[car_services]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[car_services](

        [car_id] [bigint] NULL,

        [start] [date] NOT NULL,

        [status] [smallint] NOT NULL,

        [name] [nvarchar](191) NULL,

        [obs] [nvarchar](191) NULL,

        [created_at] [datetime2](7) NULL,

        [updated_at] [datetime2](7) NULL,

        [id] [int] NOT NULL,

        [car_service_ID] [bigint] IDENTITY(1,1) NOT NULL,

        [import_ID] [bigint] NULL,

 CONSTRAINT [pk_car_service_ID] PRIMARY KEY CLUSTERED

(

        [car_service_ID] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [dbo].[car_types]    Script Date: 12/1/2023 4:10:56 PM ******/
```

```
SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[car_types](

	[name] [nvarchar](191) NOT NULL,

	[obs] [nvarchar](191) NULL,

	[created_at] [datetime2](7) NULL,

	[updated_at] [datetime2](7) NULL,

	[id] [int] NOT NULL,

	[car_type_ID] [bigint] IDENTITY(1,1) NOT NULL,

	[import_ID] [bigint] NULL,

 CONSTRAINT [pk_car_type_ID] PRIMARY KEY CLUSTERED

(

	[car_type_ID] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [dbo].[cars]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[cars](

	[car_index] [nvarchar](191) NULL,

	[license_plate] [nvarchar](191) NOT NULL,

	[type] [smallint] NOT NULL,
```

```
	[volume] [decimal](4, 2) NOT NULL,
	[average_consumption] [decimal](4, 2) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[vpn_status] [smallint] NOT NULL,
	[vpn_to_activate] [smallint] NOT NULL,
	[ip] [nvarchar](191) NULL,
	[reboot] [smallint] NOT NULL,
	[vin_number] [nvarchar](191) NULL,
	[compaction_degree] [nvarchar](191) NULL,
	[car_brand] [smallint] NULL,
	[car_model] [smallint] NULL,
	[car_type] [smallint] NULL,
	[car_status] [smallint] NULL,
	[provenance] [smallint] NULL,
	[payload] [numeric](10, 3) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[car_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_car_ID] PRIMARY KEY CLUSTERED
(
	[car_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[clients]    Script Date: 12/1/2023 4:10:56 PM ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[clients](
	[area_id] [bigint] NULL,
	[type] [smallint] NOT NULL,
	[name] [nvarchar](191) NOT NULL,
	[fiscal_code] [nvarchar](191) NOT NULL,
	[email] [nvarchar](191) NULL,
	[phone] [nvarchar](191) NULL,
	[address] [nvarchar](191) NULL,
	[contact_person] [nvarchar](191) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[client_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_client_ID] PRIMARY KEY CLUSTERED
(
	[client_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
 CONSTRAINT [clients$clients_fiscal_code_area_id_unique] UNIQUE NONCLUSTERED
(
	[fiscal_code] ASC,
	[area_id] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [dbo].[counties]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[counties](

	[name] [nvarchar](191) NOT NULL,

	[created_at] [datetime2](7) NULL,

	[updated_at] [datetime2](7) NULL,

	[id] [int] NOT NULL,

	[county_ID] [bigint] IDENTITY(1,1) NOT NULL,

	[import_ID] [bigint] NULL,

 CONSTRAINT [pk_countie_ID] PRIMARY KEY CLUSTERED

(

	[county_ID] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [dbo].[drivers]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO
```

```
CREATE TABLE [dbo].[drivers](
	[car_id] [bigint] NULL,
	[branch_id] [bigint] NULL,
	[name] [nvarchar](191) NOT NULL,
	[surname] [nvarchar](191) NOT NULL,
	[certificate] [nvarchar](191) NULL,
	[start] [date] NULL,
	[end] [date] NULL,
	[document_number] [nvarchar](191) NULL,
	[category] [nvarchar](max) NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[driver_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_driver_ID] PRIMARY KEY CLUSTERED
(
	[driver_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [dbo].[fuel_vouchers]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [dbo].[fuel_vouchers](
	[car_id] [bigint] NULL,
	[type] [smallint] NOT NULL,
	[document_number] [nvarchar](191) NOT NULL,
	[start] [date] NOT NULL,
	[quantity] [numeric](5, 2) NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[fuel_voucher_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_fuel_voucher_ID] PRIMARY KEY CLUSTERED
(
	[fuel_voucher_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[incidents]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[incidents](
	[car_id] [bigint] NULL,
	[tag_code] [nvarchar](191) NULL,
	[license_plate] [nvarchar](191) NULL,
```

```
	[collected] [smallint] NULL,
	[side] [tinyint] NOT NULL,
	[load] [nvarchar](191) NULL,
	[car_index] [nvarchar](191) NOT NULL,
	[event] [smallint] NOT NULL,
	[lat] [decimal](10, 8) NULL,
	[lng] [decimal](11, 8) NULL,
	[datetime] [datetime2](7) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[incident_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_incident_ID] PRIMARY KEY CLUSTERED
(
	[incident_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[localities]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[localities](
	[county_id] [bigint] NULL,
	[name] [nvarchar](191) NOT NULL,
```

```
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[locality_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_localitie_ID] PRIMARY KEY CLUSTERED
(
	[locality_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[locations]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[locations](
	[county_id] [bigint] NULL,
	[locality_id] [bigint] NULL,
	[client_id] [bigint] NULL,
	[people_nr] [smallint] NULL,
	[contract_number] [nvarchar](191) NULL,
	[contract_start_date] [date] NULL,
	[contract_end_date] [date] NULL,
	[street_type] [nvarchar](191) NULL,
	[street_name] [nvarchar](191) NULL,
	[street_number] [nvarchar](191) NULL,
```

```
	[building] [nvarchar](191) NULL,
	[entrance] [nvarchar](191) NULL,
	[floor] [nvarchar](191) NULL,
	[apartment] [nvarchar](191) NULL,
	[lat] [decimal](10, 8) NULL,
	[lng] [decimal](11, 8) NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[location_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_location_ID] PRIMARY KEY CLUSTERED
(
	[location_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
 CONSTRAINT [locations$locations_contract_number_unique] UNIQUE NONCLUSTERED
(
	[contract_number] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[road_maps]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
CREATE TABLE [dbo].[road_maps](
	[car_id] [bigint] NULL,
	[driver_id] [bigint] NULL,
	[area_id] [bigint] NULL,
	[departure_date] [date] NULL,
	[arrival_date] [date] NULL,
	[departure_time] [time](7) NULL,
	[arrival_time] [time](7) NULL,
	[departure_km] [numeric](10, 2) NOT NULL,
	[arrival_km] [numeric](10, 2) NULL,
	[departure_shift_id] [smallint] NULL,
	[arrival_shift_id] [smallint] NULL,
	[form_status] [smallint] NOT NULL,
	[obs] [nvarchar](max) NULL,
	[order_index] [int] NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[road_map_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_road_map_ID] PRIMARY KEY CLUSTERED
(
	[road_map_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[roles]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[roles](
	[name] [nvarchar](191) NOT NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[role_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_role_ID] PRIMARY KEY CLUSTERED
(
	[role_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[tags]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tags](
	[location_id] [bigint] NULL,
	[volume] [smallint] NOT NULL,
	[type] [smallint] NOT NULL,
```

```
	[car_type] [smallint] NOT NULL,
	[tag_code] [nvarchar](191) NOT NULL,
	[status] [smallint] NOT NULL,
	[obs] [nvarchar](191) NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[tag_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_tag_ID] PRIMARY KEY CLUSTERED
(
	[tag_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[tickets]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tickets](
	[car_id] [bigint] NULL,
	[waste_destination] [smallint] NOT NULL,
	[collection_code] [smallint] NOT NULL,
	[waste_source] [smallint] NOT NULL,
	[scale_number] [nvarchar](191) NOT NULL,
	[issue_date] [date] NOT NULL,
```

```
	[issue_time] [time](7) NOT NULL,
	[county_id] [bigint] NULL,
	[locality_id] [bigint] NULL,
	[weight] [numeric](10, 3) NOT NULL,
	[created_at] [datetime2](7) NULL,
	[updated_at] [datetime2](7) NULL,
	[id] [int] NOT NULL,
	[ticket_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_ticket_ID] PRIMARY KEY CLUSTERED
(
	[ticket_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[user_areas]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[user_areas](
	[user_id] [bigint] NULL,
	[area_id] [bigint] NULL,
	[user_area_ID] [bigint] IDENTITY(1,1) NOT NULL,
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_user_area_ID] PRIMARY KEY CLUSTERED
(
```

```
        [user_area_ID] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
    ) ON [PRIMARY]
    GO
    /****** Object:  Table [dbo].[users]    Script Date: 12/1/2023 4:10:56 PM ******/
    SET ANSI_NULLS ON
    GO
    SET QUOTED_IDENTIFIER ON
    GO
    CREATE TABLE [dbo].[users](
        [branch_id] [bigint] NULL,
        [role_id] [bigint] NULL,
        [name] [nvarchar](191) NOT NULL,
        [email] [nvarchar](191) NULL,
        [phone] [nvarchar](191) NULL,
        [email_verified_at] [datetime2](7) NULL,
        [password] [nvarchar](191) NOT NULL,
        [is_admin] [smallint] NOT NULL,
        [is_supervisor] [smallint] NOT NULL,
        [is_operator] [smallint] NOT NULL,
        [remember_token] [nvarchar](100) NULL,
        [obs] [nvarchar](191) NULL,
        [language] [nvarchar](191) NULL,
        [created_at] [datetime2](7) NULL,
        [updated_at] [datetime2](7) NULL,
        [id] [int] NOT NULL,
        [user_ID] [bigint] IDENTITY(1,1) NOT NULL,
```

```
	[import_ID] [bigint] NULL,
 CONSTRAINT [pk_user_ID] PRIMARY KEY CLUSTERED
(
	[user_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
 CONSTRAINT [users$users_email_unique] UNIQUE NONCLUSTERED
(
	[email] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [IMP].[Import]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [IMP].[Import](
	[id] [bigint] IDENTITY(1,1) NOT NULL,
	[institution_id] [bigint] NOT NULL,
	[import_time] [datetime] NOT NULL,
	[from_date] [datetime] NOT NULL,
	[to_date] [datetime] NOT NULL,
	[json_id] [bigint] NULL,
	[operation_type] [nvarchar](10) NULL,
	[import_IPAddress] [varchar](15) NULL,
	[ProcessingStartTime] [datetime] NULL,
```

```
	[ProcessingEndTime] [datetime] NULL,
	[Deleted] [bit] NULL,
	[DeletionDate] [datetime] NULL,
 CONSTRAINT [PK_Import] PRIMARY KEY CLUSTERED
(
	[id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [IMP].[Institution]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [IMP].[Institution](
	[id] [bigint] IDENTITY(1,1) NOT NULL,
	[InstitutionName] [nvarchar](100) NOT NULL,
	[Username] [nvarchar](100) NULL,
	[Password] [nvarchar](100) NULL,
 CONSTRAINT [PK_Institution] PRIMARY KEY CLUSTERED
(
	[id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [LOG].[JobProcessJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
```

```
SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [LOG].[JobProcessJSON](

	[Id] [int] IDENTITY(1,1) NOT NULL,

	[Mesaj] [nvarchar](500) NULL,

	[DataOraInceput] [datetime] NULL,

	[DataOraSfarsit] [datetime] NULL,

	[Status] [bit] NULL,

 CONSTRAINT [PK_JobProcessJSON] PRIMARY KEY CLUSTERED

(

	[Id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [LOG].[JSON_Import]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [LOG].[JSON_Import](

	[id] [bigint] IDENTITY(1,1) NOT NULL,

	[json_string] [nvarchar](max) NULL,

 CONSTRAINT [PK_JSON] PRIMARY KEY CLUSTERED

(

	[id] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO

/****** Object:  Table [LOG].[ValidationErrors]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [LOG].[ValidationErrors](

        [ErrorId] [bigint] IDENTITY(1,1) NOT NULL,

        [ShortName] [nvarchar](200) NULL,

        [Description] [nvarchar](max) NULL,

        [ErrDateTime] [datetime] NULL,

        [ValidationType] [nvarchar](100) NULL,

        [ImportId] [bigint] NULL,

 CONSTRAINT [PK_ValidationErrors] PRIMARY KEY CLUSTERED

(

        [ErrorId] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO

/****** Object:  Table [shared].[DestinatieDeseu]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO
```

```
CREATE TABLE [shared].[DestinatieDeseu](
	[Id] [tinyint] NOT NULL,
	[Name] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_DestinatieDeseu] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [shared].[SursaDeseu]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [shared].[SursaDeseu](
	[Id] [tinyint] NOT NULL,
	[Name] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_SursaDeseu] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [shared].[TipClient]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [shared].[TipClient](

	[Id] [tinyint] NOT NULL,

	[Code] [nvarchar](2) NULL,

	[Name] [nvarchar](50) NOT NULL,

 CONSTRAINT [PK_TipClient] PRIMARY KEY CLUSTERED

(

	[Id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object:  Table [shared].[TipEveniment]    Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [shared].[TipEveniment](

	[Id] [tinyint] NOT NULL,

	[Name] [nvarchar](50) NOT NULL,

 CONSTRAINT [PK_TipEveniment] PRIMARY KEY CLUSTERED

(

	[Id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO
```

```
/****** Object:  Table [shared].[TipMasina]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [shared].[TipMasina](
	[Id] [tinyint] NOT NULL,
	[Name] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_TipMasina] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [shared].[TipRecipient]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [shared].[TipRecipient](
	[Id] [tinyint] NOT NULL,
	[Name] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_TipRecipient] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
```

```
) ON [PRIMARY]
GO
/****** Object:  Table [shared].[VolumRecipient]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [shared].[VolumRecipient](
	[Id] [tinyint] NOT NULL,
	[Volume] [numeric](5, 2) NOT NULL,
	[Number] [tinyint] NOT NULL,
 CONSTRAINT [PK_VolumRecipient] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Temp].[TestDate]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Temp].[TestDate](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[sDataInceput] [nvarchar](100) NULL,
	[sDataSfarsit] [nvarchar](100) NULL,
	[DataInceput] [datetime] NULL,
```

```
	[DataSfarsit] [datetime] NULL,

	[DataInregistrarii] [datetime] NULL,

 CONSTRAINT [PK_TestDate] PRIMARY KEY CLUSTERED

(

	[Id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [IMP].[Import] ADD  CONSTRAINT [DF_Import_Deleted]  DEFAULT ((0)) FOR [Deleted]

GO

ALTER TABLE [dbo].[areas]  WITH CHECK ADD  CONSTRAINT [fk_Import_areas] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[areas] CHECK CONSTRAINT [fk_Import_areas]

GO

ALTER TABLE [dbo].[authorizations]  WITH CHECK ADD  CONSTRAINT [fk_Import_authorizations] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[authorizations] CHECK CONSTRAINT [fk_Import_authorizations]

GO

ALTER TABLE [dbo].[bin_collections]  WITH CHECK ADD  CONSTRAINT [fk_Import_bin_collections] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[bin_collections] CHECK CONSTRAINT [fk_Import_bin_collections]

GO
```

```
ALTER TABLE [dbo].[branches]  WITH CHECK ADD  CONSTRAINT [fk_Import_branches] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[branches] CHECK CONSTRAINT [fk_Import_branches]

GO

ALTER TABLE [dbo].[car_areas]  WITH CHECK ADD  CONSTRAINT [fk_Import_car_areas] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_areas] CHECK CONSTRAINT [fk_Import_car_areas]

GO

ALTER TABLE [dbo].[car_brands]  WITH CHECK ADD  CONSTRAINT [fk_Import_car_brands] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_brands] CHECK CONSTRAINT [fk_Import_car_brands]

GO

ALTER TABLE [dbo].[car_drivers]  WITH CHECK ADD  CONSTRAINT [fk_Import_car_drivers] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_drivers] CHECK CONSTRAINT [fk_Import_car_drivers]

GO

ALTER TABLE [dbo].[car_models]  WITH CHECK ADD  CONSTRAINT [fk_Import_car_models] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_models] CHECK CONSTRAINT [fk_Import_car_models]

GO
```

```
ALTER TABLE [dbo].[car_provenances] WITH CHECK ADD CONSTRAINT [fk_Import_car_provenances] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_provenances] CHECK CONSTRAINT [fk_Import_car_provenances]

GO

ALTER TABLE [dbo].[car_services] WITH CHECK ADD CONSTRAINT [fk_Import_car_services] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_services] CHECK CONSTRAINT [fk_Import_car_services]

GO

ALTER TABLE [dbo].[car_types] WITH CHECK ADD CONSTRAINT [fk_Import_car_types] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[car_types] CHECK CONSTRAINT [fk_Import_car_types]

GO

ALTER TABLE [dbo].[cars] WITH CHECK ADD CONSTRAINT [fk_Import_cars] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[cars] CHECK CONSTRAINT [fk_Import_cars]

GO

ALTER TABLE [dbo].[clients] WITH CHECK ADD CONSTRAINT [fk_Import_clients] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[clients] CHECK CONSTRAINT [fk_Import_clients]

GO
```

```
ALTER TABLE [dbo].[counties] WITH CHECK ADD CONSTRAINT [fk_Import_counties] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[counties] CHECK CONSTRAINT [fk_Import_counties]

GO

ALTER TABLE [dbo].[drivers] WITH CHECK ADD CONSTRAINT [fk_Import_drivers] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[drivers] CHECK CONSTRAINT [fk_Import_drivers]

GO

ALTER TABLE [dbo].[fuel_vouchers] WITH CHECK ADD CONSTRAINT [fk_Import_fuel_vouchers] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[fuel_vouchers] CHECK CONSTRAINT [fk_Import_fuel_vouchers]

GO

ALTER TABLE [dbo].[incidents] WITH CHECK ADD CONSTRAINT [fk_Import_incidents] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[incidents] CHECK CONSTRAINT [fk_Import_incidents]

GO

ALTER TABLE [dbo].[localities] WITH CHECK ADD CONSTRAINT [fk_Import_localities] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[localities] CHECK CONSTRAINT [fk_Import_localities]

GO
```

```
ALTER TABLE [dbo].[locations] WITH CHECK ADD CONSTRAINT [fk_Import_locations] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[locations] CHECK CONSTRAINT [fk_Import_locations]

GO

ALTER TABLE [dbo].[road_maps] WITH CHECK ADD CONSTRAINT [fk_Import_road_maps] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[road_maps] CHECK CONSTRAINT [fk_Import_road_maps]

GO

ALTER TABLE [dbo].[roles] WITH CHECK ADD CONSTRAINT [fk_Import_roles] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[roles] CHECK CONSTRAINT [fk_Import_roles]

GO

ALTER TABLE [dbo].[tags] WITH CHECK ADD CONSTRAINT [fk_Import_tags] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[tags] CHECK CONSTRAINT [fk_Import_tags]

GO

ALTER TABLE [dbo].[tickets] WITH CHECK ADD CONSTRAINT [fk_Import_tickets] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[tickets] CHECK CONSTRAINT [fk_Import_tickets]

GO
```

```
ALTER TABLE [dbo].[user_areas] WITH CHECK ADD CONSTRAINT [fk_Import_user_areas] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[user_areas] CHECK CONSTRAINT [fk_Import_user_areas]

GO

ALTER TABLE [dbo].[users] WITH CHECK ADD CONSTRAINT [fk_Import_users] FOREIGN KEY([import_ID])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [dbo].[users] CHECK CONSTRAINT [fk_Import_users]

GO

ALTER TABLE [IMP].[Import] WITH CHECK ADD CONSTRAINT [FK_Import_Institution] FOREIGN KEY([institution_id])

REFERENCES [IMP].[Institution] ([id])

GO

ALTER TABLE [IMP].[Import] CHECK CONSTRAINT [FK_Import_Institution]

GO

ALTER TABLE [IMP].[Import] WITH CHECK ADD CONSTRAINT [FK_Import_JSON] FOREIGN KEY([json_id])

REFERENCES [LOG].[JSON_Import] ([id])

GO

ALTER TABLE [IMP].[Import] CHECK CONSTRAINT [FK_Import_JSON]

GO

ALTER TABLE [LOG].[ValidationErrors] WITH CHECK ADD CONSTRAINT [FK_ValidationErrors_Import] FOREIGN KEY([ImportId])

REFERENCES [IMP].[Import] ([id])

GO

ALTER TABLE [LOG].[ValidationErrors] CHECK CONSTRAINT [FK_ValidationErrors_Import]

GO
```

```
/****** Object:  StoredProcedure [dbo].[spCat_AdaugCheiPrimareNoi]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE proc [dbo].[spCat_AdaugCheiPrimareNoi]
as
BEGIN
declare @SQL nvarchar(max)=''

--Select @SQL=@SQL +'ALTER TABLE ' + s.name + '.' + t.name + ' ADD ' + LEFT(t.name,LEN(t.name)-1) + '_ID bigint identity not null;'+ char(13)
--from sys.tables t
--inner join sys.schemas s on s.schema_id=t.schema_id
--where t.name<>'sysdiagrams'

Select @SQL=@SQL +'ALTER TABLE ' + s.name + '.' + t.name + ' ADD CONSTRAINT ' + 'pk_' + LEFT(t.name,LEN(t.name)-1) + '_ID PRIMARY KEY ' +
                                        '(' + LEFT(t.name,LEN(t.name)-1) + '_ID)' +
                                        ';' + char(13)
from sys.tables t
inner join sys.schemas s on s.schema_id=t.schema_id
where t.name<>'sysdiagrams'

Print  @SQL
exec sp_executesql @SQL
end
GO
```

```
/****** Object:  StoredProcedure [dbo].[spCat_CheieExternaCuImport]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE proc [dbo].[spCat_CheieExternaCuImport]
as
begin
declare @SQL nvarchar(max)=''

Select                  @SQL=@SQL +
                            'ALTER TABLE ' + s.name + '.'+t.name +
                            ' ADD import_ID bigint;' +char(13) +
                            ' ALTER TABLE ' + s.name + '.'+t.name +' ADD CONSTRAINT fk_Import_' + t.name +
                            ' FOREIGN KEY (import_ID) REFERENCES ' + 'IMP.Import(id);'+char(13)
FROM                    sys.tables t
INNER JOIN              sys.schemas s on s.schema_id=t.schema_id
where                   t.name not in ('sysdiagrams','Import')

print @SQL
exec (@SQL)
end
GO
/****** Object:  StoredProcedure [dbo].[spCat_RefacereCampIdFaraIdentity]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
create proc [dbo].[spCat_RefacereCampIdFaraIdentity]
as
begin
declare @SQL nvarchar(max)=''

Select @SQL=@SQL + 'ALTER TABLE ' + s.name + '.' + t.name + ' DROP COLUMN ' + c.name + ';' + char(13)
from sys.columns c
INNER JOIN sys.tables t on t.object_id=c.object_id
INNER JOIN sys.schemas s on s.schema_id=t.schema_id
where s.name='medias'
and c.is_identity=1

Select @SQL=@SQL + 'ALTER TABLE ' + s.name + '.' + t.name + ' ADD id int not null;' + char(13)
from sys.tables t
INNER JOIN sys.schemas s on s.schema_id=t.schema_id
where s.name='medias'

Print @SQL
exec sys.sp_executesql @SQL
end
GO
/****** Object: StoredProcedure [dbo].[spCat_StergCheiPrimare] Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
CREATE proc [dbo].[spCat_StergCheiPrimare]
as
begin
	DECLARE @sql NVARCHAR(MAX);
	SET @sql = N'';

	SELECT @sql = @sql + N'ALTER TABLE ' + QUOTENAME(s.name) + '.'+ QUOTENAME(OBJECT_NAME(t.[object_id])) + ' DROP CONSTRAINT ' + QUOTENAME(I.[name]) +';'
	FROM sys.indexes I
	INNER JOIN sys.tables t on t.object_id=I.object_id
	inner join sys.schemas s on s.schema_id=t.schema_id
	Where is_primary_key = 1 Or is_unique_constraint = 1
	 And I.type_desc = 'CLUSTERED';

	Select @sql=@sql + N'DROP INDEX ' + QUOTENAME(I.[name]) + ' ON ' + QUOTENAME(s.name)+ '.'+  QUOTENAME(OBJECT_NAME(t.[object_id])) +';'
	FROM sys.indexes I
	INNER JOIN sys.tables t on t.object_id=I.object_id
	inner join sys.schemas s on s.schema_id=t.schema_id
	Where is_primary_key = 0 And is_unique_constraint = 0
	 And I.type_desc = 'CLUSTERED';

	PRINT @sql;
	exec sys.sp_executesql @sql;
end
GO
/****** Object:  StoredProcedure [dbo].[spCat_StergeCheiExterne]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE proc [dbo].[spCat_StergeCheiExterne]
as
begin
declare @SQL nvarchar(max)=''

Select @SQL=@SQL +'ALTER TABLE ' + s.name + '.' + t.name + ' DROP CONSTRAINT ' + fk.name + ';' + char(13)
--Select fk.name, fkc.constraint_object_id, fkc.parent_object_id, cpa.name, cref.name, s.name, t.name
from sys.foreign_keys fk
INNER JOIN sys.foreign_key_columns fkc ON fkc.constraint_object_id = fk.object_id
  INNER JOIN sys.columns cpa ON fkc.parent_object_id = cpa.object_id AND fkc.parent_column_id = cpa.column_id
  INNER JOIN sys.columns cref ON fkc.referenced_object_id = cref.object_id AND fkc.referenced_column_id = cref.column_id
INNER JOIN sys.tables t on cpa.object_id=t.object_id
INNER JOIN sys.schemas s on s.schema_id=t.schema_id
order by t.name

PRINT @SQL
exec sp_executesql @SQL
end
GO
/****** Object:  StoredProcedure [dbo].[spCatVerificErori]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
create proc [dbo].[spCatVerificErori] @json nvarchar(max)
as
begin
declare @errors nvarchar(max)=''
exec [dbo].[spCris_#ValidareStructuraJSON] @json, @errors out
select @errors as SchemaErrors
end
GO
/****** Object:  StoredProcedure [dbo].[spCatVerificJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
create proc [dbo].[spCatVerificJSON] @json nvarchar(max)
as
declare @result int
Select @result= isjson(isnull(@json,''))
Select @result
GO
/****** Object:  StoredProcedure [dbo].[spCris_#ValidareStructuraJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [dbo].[spCris_#ValidareStructuraJSON] @pJSON nvarchar(max), @pEroriStructura nvarchar(max) out
```

```
as

BEGIN

set @pEroriStructura=N''

if isjson(isnull(@pJSON,''))=0
 begin
  set @pEroriStructura='JSON invalid'
  return
 end

declare @sql nvarchar(max) = ';'
select @sql=@sql + 'select ' + cast(t.object_id as varchar(10))+ ' as IdTabel, ''' +
                        + t.name + ''' as NumeTabel,
      json_query(''' + @pJSON + ''', ''$."'+ t.name + '"'') as DateJSON;'
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
where t.name<>'sysdiagrams' and s.name not in ('IMP', 'LOG')

----select @sql

declare @tRezultat table (IdTabel int,
                    NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                      StructuraJSON nvarchar(max),
                                                      StructuraTabel nvarchar(max),
                                                      StructuraTabel_tmp nvarchar(max)
                                                )
insert @tRezultat (IdTabel,NumeTabel,SirJSON)
```

```
	exec (@sql)


	----select * from @tRezultat


	update @tRezultat
	set StructuraJSON = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraJson'),
		StructuraTabel =[dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel'),
		StructuraTabel_tmp = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel_tmp')


	-----select * from @tRezultat


	declare c cursor
	for select NumeTabel, SirJSON, StructuraJSON, StructuraTabel, StructuraTabel_tmp from @tRezultat



	declare @idTabel int
	declare @numeTabel nvarchar(max)
	declare @sirJSON nvarchar(max)
	declare @structuraJSON nvarchar(max)
	declare @structuraTabel nvarchar(max)
	declare @structuraTabel_tmp nvarchar(max)



	open c
	fetch next from c into @numeTabel,@sirJSON,
		@structuraJSON,@structuraTabel,@structuraTabel_tmp
```

```
while @@FETCH_STATUS=0
begin
  begin try
        if isnull(@sirJSON,'')<>''
          begin
                                                 declare @sirSQL nvarchar(max) = N'
                                                declare @tmp table (' + @structuraTabel_tmp + ');
                                                INSERT  @tmp ( ' + @structuraTabel + ')
                                                SELECT ' + @structuraTabel + '
                                                FROM OPENJSON(''' + @sirJson + ''')
                                                        WITH (' + @structuraJson + ');'
                        --select @sirSQL
                                                exec (@sirSQL)
                                end
--------------------------------------------------


          end try
          begin catch
           set @pEroriStructura+=@numeTabel+';'
   end catch
   fetch next from c into @numeTabel,@sirJSON,
                @structuraJSON,@structuraTabel,@structuraTabel_tmp
end
close c
deallocate c

----select * from @tRezultat
```

```
	if @pEroriStructura=N''
	    set @pEroriStructura=null
	else
		set @pEroriStructura = SUBSTRING(@pEroriStructura,0,len(@pEroriStructura))

	END
	GO
	/****** Object:  StoredProcedure [dbo].[spCris_DeleteJSON]    Script Date: 12/1/2023 4:10:56
PM ******/
	SET ANSI_NULLS ON
	GO
	SET QUOTED_IDENTIFIER ON
	GO

	/****** Object:  StoredProcedure [dbo].[spCris_UpdateJSON]    Script Date: 02/07/2023 20:13:33
******/

	CREATE PROC [dbo].[spCris_DeleteJSON] @pImportID bigint,
				@pErrData nvarchar(max) out,
                                                                @pErrTransaction
nvarchar(max) out
	as

	BEGIN
	-----------------------------------------------------------------
	declare @start datetime = getdate()

	declare @json_string nvarchar(max)
	declare @institution_id bigint
```

```
select @json_string=l.[json_string],@institution_id=i.institution_id
from [LOG].[JSON_Import] l inner join IMP.Import i on l.id=i.json_id
where i.id = @pImportID
-----------------------------------------------------
if isjson(isnull(@json_string,''))=0  ---> ca verificare minimala, se presupune structura valida
 begin
  set @pErrData = 'JSON invalid'
  return
 end
-----------------------------------------------------

declare @tRezultat table (IdTabel int,
                    NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                    NumePK_Import sysname,
                                                            EroriMissingID nvarchar(max),
                                                                    EroriFK nvarchar(max)
                                                )
insert @tRezultat (IdTabel, NumeTabel, SirJSON, NumePK_Import)
select f.IdTabel, f.NumeTabel,f.SirJSON,v.CampPK_Import
from [dbo].[fCris_ExtrageDateJSON](@json_string) f
        inner join viewCris3_Test_ListaPK_Import v on f.IdTabel=v.IdTabel
-----------------------------------------------------------------
-----------------------------------------------------------------

delete from @tRezultat where SirJSON is null ---- pentru simpificare conditii ulterioare
----select * from @tRezultat
```

```
	declare @tmpPK table (IdTabel bigint, NumeTabel sysname, ValoarePK bigint,ValoarePK_Import bigint,ErrStergere nvarchar(30))

	declare @tmpPKValExistente table (IdTabel bigint, ValoarePK bigint, ValoarePK_import bigint)


	declare @idTabel int

	declare @numeTabel sysname

	declare @sirJSON nvarchar(max)

	declare @numePK_import sysname


	declare c cursor forward_only

	for

	select IdTabel, NumeTabel, SirJSON, NumePK_Import  from @tRezultat


	open c

	fetch next from c into @idTabel, @numeTabel,@sirJSON, @numePK_import


	while @@FETCH_STATUS=0

		begin

			insert @tmpPK (IdTabel,NumeTabel, ValoarePK)

		select distinct @idTabel, @numeTabel, id

				from OPENJSON(@sirJson)

				with (id bigint)

				where id is not null


		---declare @numePK_import sysname = (select CampPK_Import from viewCris3_Test_ListaPK_Import where TabelPK=@numeTabel)

				declare @sqlPK varchar(max) = 'select ' + cast(@idTabel as nvarchar(max)) + ', t.id, t.' +  @numePK_import

					+ ' FROM ' + @numeTabel + ' t inner join IMP.Import i on t.import_ID=i.id '
```

```
                                                                                    +       '
where i.institution_id=' + cast(@institution_id as nvarchar(max))
                                    ----------m.sus Select Distinct pt. teste!!! nu ar trebui sa existe duplicate in
BD
                                    ---------- deja am inserat de m.multe ori acelasi sir JSON

                                    insert @tmpPKValExistente (IdTabel, ValoarePK, ValoarePK_import)
                                exec (@sqlPK)

                            update @tmpPK
                                    set  ValoarePK_Import=e.ValoarePK_import   -- id generat pt BD curenta
                                    from @tmpPK t inner join @tmpPKValExistente e
                                                                        on          (t.IdTabel=e.IdTabel          and
t.ValoarePK=e.ValoarePK)
                                    where t.IdTabel=@idTabel

                                    update @tmpPK
                                    set ErrStergere = 'ID'   -- id lipsa
                                    where IdTabel=@idTabel and  ValoarePK_Import is null

                                    delete from @tmpPKValExistente

                                    fetch next from c into @idTabel, @numeTabel,@sirJSON, @numePK_import
                            end
                    close c
                    deallocate c

                    ----select * from @tmpPK
                    ------------------------------------------------------------------------
                    ------------------------------------------------------------------------
```

```
	declare @tmpDependenteTabele table (idTabel bigint, numeTabel sysname,
nivelMaxDependenta tinyint)

	insert @tmpDependenteTabele (idTabel, numeTabel, nivelMaxDependenta)
	select IdTabel,NumeTabel,NivelMaxDependenta
	from viewCris_DependenteTabele
	------------------------------------------------------------------------
	------------------------------------------------------------------------
	-----select * from @tmpDependenteTabele

	declare @sqlEroriStergere nvarchar(max)=N''
	declare @tmpStergereFK table (idTabel_parinte bigint, tabel_parinte sysname,
				idTabel_copil bigint, tabel_copil sysname)

	declare @tmpVerificaFK table (idTabel_parinte bigint, tabel_parinte sysname,
				idTabel_copil bigint, tabel_copil sysname,
							campFK sysname,
NivelDependenta tinyint)

	declare @tmpValoriBD_TabelCopil table (ValoarePK bigint, ValoareFK bigint)


	declare @nivelDependenta tinyint
	declare @maxGeneral tinyint

	select @maxGeneral=max(nivelMaxDependenta)
	from @tmpDependenteTabele

	declare cDependente cursor forward_only
```

```
for
select idTabel, numeTabel, nivelMaxDependenta
from @tmpDependenteTabele
order by 3 DESC



open cDependente
fetch next from cDependente into @idTabel, @numeTabel, @nivelDependenta
while @@FETCH_STATUS=0
        begin
            --- dependente noi, de verificat acum
                insert @tmpVerificaFK(tabel_parinte,
                                        tabel_copil, campFK)
                (select distinct [tabelParinte], [tabelCopil], i.[campFK]--
,min([NivelDependenta]) as NivelDependenta
                from [dbo].[viewCris_IerarhieCompletaTabele] i left join
@tmpStergereFK e
                    on (i.tabelParinte=e.tabel_parinte and i.tabelCopil=e.tabel_copil)
                where i.Parinte_Nivel  = @numeTabel and i.NivelDependenta>0
                    and e.tabel_parinte is null
                )
                ----- dependente deja prelucrate --> ca sa nu repete verificarile in functie
de ierahia curenta (pt tabelul parinte)
                insert @tmpStergereFK(tabel_parinte, tabel_copil)
                select tabel_parinte, tabel_copil
                from @tmpVerificaFK

                declare cVerificaFK cursor forward_only
                for
                select tabel_parinte, tabel_copil, campFK
```

```
                    from @tmpVerificaFK

                    declare @tabel_parinte sysname
                    declare @tabel_copil sysname
                    declare @campFK sysname

                    open cVerificaFK
          fetch next from cVerificaFK into @tabel_parinte, @tabel_copil, @campFK
          while @@FETCH_STATUS=0
               begin
                         declare @sqlTabelCopil nvarchar(max)='select c.id, c.' + @campFK
                                   + ' from ' + @tabel_copil
                                   + ' c inner join imp.import i on i.id=c.import_ID'
                                                                        +      '
where i.institution_id=' + convert(nvarchar(max),@institution_id)

            insert @tmpValoriBD_TabelCopil (ValoarePK, ValoareFK)
                              exec (@sqlTabelCopil)

                              ;with ValoriPKDeSters as (
                                select p.ValoarePK
                                from @tmpPK p
                                where p.NumeTabel=@tabel_parinte
                              ),
                              ValoriFKDeSters as (
                                select p.ValoarePK
                                from @tmpPK p
                                where p.NumeTabel=@tabel_copil
                              ),
```

```
                                        eroriFK as (
                                        select p.*
                                        from ValoriPKDeSters p
                                          inner join @tmpValoriBD_TabelCopil cBD on
p.ValoarePK=cBD.ValoareFK
                                          left join ValoriFKDeSters c on cBD.ValoarePK=c.ValoarePK
                    where c.ValoarePK is null
                                        )
                                        update @tmpPK
                                        set ErrStergere = 'FK'
                                        from @tmpPK p inner join eroriFK e on p.ValoarePK=e.ValoarePK
                                        where p.NumeTabel=@tabel_parinte

                                        delete from @tmpValoriBD_TabelCopil  --- pt urmatoarea iteratie

                                          fetch next from cVerificaFK into @tabel_parinte, @tabel_copil,
@campFK
                                    end
                                    close cVerificaFK
                                    deallocate cVerificaFK

                                    delete from @tmpVerificaFK --- pt urmatoarea iteratie
                     fetch next from cDependente into @idTabel, @numeTabel, @nivelDependenta
                   end
          close cDependente
          deallocate cDependente

          -----select * from @tmpPK
          ---------------------------------------------------------------------------
```

```
	--------------------------------------------------------------------------

	--------------------------------------------------------------------------

	declare @errStergere varchar(max)=''
	select @errStergere=p.ErrStergere
	from @tmpPK p
	where p.ErrStergere is not null

	-----nu conteaza ordinea de stergere, daca s-a stabilit ca e ok (relatiile nu sunt in BD curenta)
	declare cDelete cursor forward_only
	for
	select IdTabel,NumeTabel, NumePK_Import from @tRezultat

	declare @sqlDelete nvarchar(max)=N''
	-------------------------------------------------
	open cDelete
	fetch next from cDelete into @idTabel, @numeTabel, @numePK_import
	while @@FETCH_STATUS=0
		begin
			declare @listaErrMissingID varchar(max)=''
			declare @listaErrFK varchar(max)=''
			declare @listaID_Delete varchar(max)=''

			if @errStergere=''
				select @listaID_Delete+= convert(nvarchar(max), p.ValoarePK_Import) +
','
				from @tmpPK p
				where p.IdTabel=@idTabel
	else
```

```
                select @listaErrMissingID+= case when p.ErrStergere='ID' then
convert(nvarchar(max), p.ValoarePK) + ',' else '' end,
                @listaErrFK+= case when p.ErrStergere='FK' then convert(nvarchar(max),
p.ValoarePK) + ',' else '' end
                from @tmpPK p
                where p.IdTabel=@idTabel

            ---------- se mai notifica valori ID inexistente?
            if @listaErrMissingID<>''
                update @tRezultat
                set                                                    EroriMissingID
=SUBSTRING(@listaErrMissingID,0,len(@listaErrMissingID))
                where IdTabel=@idTabel
            ------------------------------------------------------------------------
        if @listaErrFK<>''
                update @tRezultat
                set EroriFK =SUBSTRING(@listaErrFK,0,len(@listaErrFK))
                where IdTabel=@idTabel
    ------------------------------------------------------------------------
        if @listaID_Delete<>''
                set @sqlDelete += 'Delete from ' + @numeTabel
                                + ' Where ' + @numePK_import + ' in (' +
SUBSTRING(@listaID_Delete,0,len(@listaID_Delete)) + ');'
            ------------------------------------------------------------------------
        fetch next from cDelete into @idTabel, @numeTabel, @numePK_import
            end
    close cDelete
    deallocate cDelete
    ----select * from @tRezultat
    ------------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------------
if @sqlDelete<>N''
	begin
		BEGIN TRANSACTION;
		-------select @sqlDelete

		begin try
			update [IMP].[Import]
			set [ProcessingStartTime]=@start
			where id=@pImportID

			exec (@sqlDelete)

			update [IMP].[Import]
			set [ProcessingEndTime]=getdate()
			where id=@pImportID
		end try

		begin catch
			set @pErrTransaction = 'Error number: ' +
convert(varchar(10),ERROR_NUMBER())
				+ ' Severity: ' +
convert(varchar(10),ERROR_SEVERITY())
				+ ' Description: ' +
ERROR_MESSAGE()
			if @@TRANCOUNT>0
				rollback;
		end catch

	if @@TRANCOUNT>0
```

```
                commit;
            end
        else  -------- erori
          begin
                set @pErrData = N''
                select @pErrData += t.NumeTabel + '(' + t.EroriFK
                                + isnull('; missing_id:' + t.EroriMissingID, '')
                                                                + ') '
                                                        from @tRezultat t
                                                        where  (t.EroriFK is not null)

                ----- select @pErrData , @pErrTransaction
             ----- select * from @tRezultat
                end

        END

        GO
        /****** Object:  StoredProcedure [dbo].[spCris_InsertJSON]    Script Date: 12/1/2023 4:10:56 PM
******/
        SET ANSI_NULLS ON
        GO
        SET QUOTED_IDENTIFIER ON
        GO
        CREATE PROC [dbo].[spCris_InsertJSON] @pImportID bigint,
                                @pErrTransaction nvarchar(max) out
        as

        BEGIN
```

```
----in prealabil verificare --> sir JSON valid ca structura!!!
declare @json_string nvarchar(max)
select @json_string=l.[json_string]
from [LOG].[JSON_Import] l inner join IMP.Import i on l.id=i.json_id
where i.id = @pImportID

------------------------------------------------------
------------------------------------------------------
--- comentariu 25.iulie ---------------------------------
/*
declare @sql nvarchar(max) = ';'
select @sql=@sql + 'select ' + cast(t.object_id as varchar(10))+ ' as IdTabel, ''' +
                        + t.name + ''' as NumeTabel,
      json_query(''' + @json_string + ''', ''$."'+ t.name + '"'') as DateJSON;'
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
where s.name = 'dbo' and t.name not in (select [table_name] from [IMP].[IgnoredTables])

----select @sql

declare @tRezultat table (IdTabel int,
                    NumeTabel nvarchar(100),
                                                        SirJSON nvarchar(Max),
                                                          StructuraJSON nvarchar(max),
                                                          StructuraTabel nvarchar(max)
                                                    )
insert @tRezultat (IdTabel,NumeTabel,SirJSON)
exec (@sql)
*/
```

```
-------------------------------------------------------------------------
--- cod nou 25.iulie --------------------------------
declare @tRezultat table (IdTabel int,
                          NumeTabel nvarchar(100),
                                                              SirJSON nvarchar(Max),
                                                                StructuraJSON nvarchar(max),
                                                                StructuraTabel nvarchar(max)
                                                            )
insert @tRezultat (IdTabel, NumeTabel, SirJSON)
select IdTabel, NumeTabel,SirJSON
from [dbo].[fCris_ExtrageDateJSON](@json_string)
--- sfarsit cod nou-------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------



delete from @tRezultat where SirJSON is null
----select * from @tRezultat

update @tRezultat
set StructuraJSON = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraJson'),
          StructuraTabel =[dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel')


-----select * from @tRezultat

declare c cursor
for select NumeTabel, SirJSON, StructuraJSON, StructuraTabel  from @tRezultat
```

```
declare @idTabel int
declare @numeTabel nvarchar(max)
declare @sirJSON nvarchar(max)
declare @structuraJSON nvarchar(max)
declare @structuraTabel nvarchar(max)

declare @start datetime = getdate()

open c

BEGIN TRANSACTION;

begin try
   update [IMP].[Import]
        set [ProcessingStartTime]=@start
        where id=@pImportID

        fetch next from c into @numeTabel,@sirJSON,
                                                @structuraJSON,@structuraTabel

        while @@FETCH_STATUS=0
                begin
                        set
@structuraTabel=replace(replace(replace(@structuraTabel,',[import_ID],',','),',[import_ID]',''),'[import_ID],','')
                        declare @sirSQL nvarchar(max) = N'INSERT ' + @numeTabel + '( ' + @structuraTabel + ',[import_ID])'
                                +'SELECT ' + @structuraTabel + ',' + cast(@pImportID as varchar(max))
                                +' FROM OPENJSON(''' + @sirJson + ''')'
```

```
                                        +' WITH (' + @structuraJson + ');'
            ---select @sirSQL
                        exec (@sirSQL)

        fetch next from c into @numeTabel,@sirJSON,
              @structuraJSON,@structuraTabel
    end

        update [IMP].[Import]
        set [ProcessingEndTime]=getdate()
        where id=@pImportID
end try

begin catch
        set @pErrTransaction = 'Error number: ' + convert(varchar(10),ERROR_NUMBER())
                    + 'Severity: ' + convert(varchar(10),ERROR_SEVERITY())
                                                + 'Description: ' + ERROR_MESSAGE()
        --select ERROR_NUMBER() AS errNumber,ERROR_SEVERITY() AS errSeverity,
        --      ERROR_MESSAGE() AS errMsg
  if @@TRANCOUNT>0
              rollback;
end catch

if @@TRANCOUNT>0
  commit;

close c
deallocate c
```

```
END
GO
/****** Object:  StoredProcedure [dbo].[spCris_PrelucreazaJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [dbo].[spCris_PrelucreazaJSON]
AS
BEGIN


declare @tmpErr_Institutii table (InstitutionID bigint)


;with ultimulImportOK
as
(
select institution_id, max(id) as IDMaxImport
from [IMP].[Import]
where ProcessingStartTime is not null
group by institution_id
)
insert @tmpErr_Institutii(InstitutionID)
select distinct i.institution_id -----> pt teste + i.id
from IMP.Import i inner join log.ValidationErrors e on i.id=e.ImportId
		left join ultimulImportOK u on i.institution_id=u.institution_id
where i.Deleted=0 and i.id>isnull(IDMaxImport,0)
```

```
declare c_import cursor forward_only
for
select i.id as import_id, i.institution_id, lower(trim(i.operation_type))
from [IMP].[Import] i left join [LOG].[ValidationErrors] e on i.id=e.ImportId
where --i.id >=99 and
            e.ErrorId is null and i.ProcessingStartTime is null
      and i.Deleted=0
order by i.id

declare @import_id bigint
declare @institution_id bigint
declare @operationType nvarchar(20)
declare @err_string nvarchar(max)
declare @errType nvarchar(30)



open c_import
fetch next from c_import into @import_id, @institution_id, @operationType

while @@FETCH_STATUS=0
          begin
            set @err_string=N''
            set @errType=N''

            declare @idInstErr bigint
            select @idInstErr=InstitutionID
            from @tmpErr_Institutii
            where InstitutionID = @institution_id
```

```
if @idInstErr is null
        begin
    begin try
        if @operationType = 'delete'
            begin
                declare @errData nvarchar(max)
                declare @errTransaction nvarchar(max)
                exec [dbo].[spCris_DeleteJSON] @import_id, @errData out, @errTransaction out

                ---------- ar trebui sa fie exclusive:
                if @errData<>N''
                  begin
                    set @errType= 'Data'
                    set @err_string = @errData
                  end
                if @errTransaction<>N''
                  begin
                    set @errType= 'Transaction'
                    set @err_string = @errTransaction
                  end
                ------------------------------------------------------------
            end
        if @operationType = 'add' or @operationType = 'edit'
            begin
                ---select @import_id, @err_string
```

```
                                                    exec        [dbo].[spCris_ValideazaSchemaJSON]
@import_id, @err_string out
                                                    ---select @import_id, @err_string

                                                    if @err_string<>N''
                                                            set @errType = 'Schema'
                                                    else
                                                             begin
                                                                    exec
[dbo].[spCris_ValideazaDateJSON] @import_id, @operationType, @err_string out
                                                                    if @err_string<>N''
                                                                            set  @errType  =
'Data'
                                                                    else
                                                                            begin
                                                                                    if
@operationType = 'add'

        exec [dbo].[spCris_InsertJSON] @import_id, @err_string out
                                                                                    if
@operationType = 'edit'

        exec [dbo].[spCris_UpdateJSON] @import_id, @err_string out
                                                                                    if
@err_string<>N''

        set @errType = 'Transaction'
                                                                            end
                                                            end
                                                end
                end try
                                begin catch
```

```
                                    set @errType='Unknown'
                                    set @err_string='Importul nu poate fi procesat'
                                end catch
                        end
                else
                        begin
                                set @errType='Batch'
                                set @err_string='Importul nu poate fi procesat - au fost identificate erori intr-un import anterior.'
                                end

                if  @err_string <> N''
                   begin
                                insert [LOG].[ValidationErrors]([Description], [ErrDateTime], [ValidationType], [ImportId])
                                select @err_string,getdate(),@errType,@import_id

                                if @errType <> 'Batch'
                                        insert @tmpErr_Institutii
                                        values (@institution_id)
                end

                fetch next from c_import into @import_id, @institution_id, @operationType
                   end
        close c_import
        deallocate c_import


        END
```

```
GO

/****** Object: StoredProcedure [dbo].[spCris_StergereIntegralaImport] Script Date: 12/1/2023 4:10:56 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE proc [dbo].[spCris_StergereIntegralaImport]  @pImportID bigint

as

---- doar pentru teste -> anuleaza efecte import (doar operatii ADD)

------ atentie la logica datelor, daca exista importuri/operatii ulterioare

begin


declare @tipOperatie nvarchar(max)

select @tipOperatie=operation_type

from IMP.Import

where id=@pImportID


if isnull(@tipOperatie,'') not in ('add', 'edit')

   begin

        select 'Import care nu corespunde unei operatii de tip Add/Edit'

        return

      end


declare @tmpDependenteTabele table (idTabel bigint, numeTabel sysname, nivelMaxDependenta tinyint)


insert @tmpDependenteTabele (idTabel, numeTabel, nivelMaxDependenta)

select IdTabel,NumeTabel,NivelMaxDependenta
```

```
from viewCris_DependenteTabele

declare cDependente cursor forward_only
for
select idTabel, numeTabel, nivelMaxDependenta
from @tmpDependenteTabele
order by 3 DESC

declare @idTabel bigint
declare @numeTabel sysname
declare @nivelMaxDependenta tinyint
declare @sqlDelete nvarchar(max)=N''

open cDependente
fetch next from cDependente into @idTabel, @numeTabel, @nivelMaxDependenta

while @@FETCH_STATUS=0
        begin
          set @sqlDelete += 'Delete from ' + @numeTabel + ' where import_ID = ' +
convert(nvarchar(max),@pImportID) + ';'
      fetch next from cDependente into @idTabel, @numeTabel, @nivelMaxDependenta
        end
close cDependente
deallocate cDependente

if len(@sqlDelete)>0
        begin
                BEGIN TRANSACTION;
```

```
                begin try
                        exec (@sqlDelete)

                        update [IMP].[Import]
                        set [ProcessingStartTime]=null, [ProcessingEndTime]=null
                        where id=@pImportID

                        select 'stergere ok - import ' + convert(nvarchar(max),@pImportID)
                end try

                begin catch
                        select 'Error number: ' + convert(varchar(10),ERROR_NUMBER())
                                + 'Severity: ' + convert(varchar(10),ERROR_SEVERITY())
                                + 'Description: ' + ERROR_MESSAGE()
                        if @@TRANCOUNT>0
                                rollback;
                end catch

                if @@TRANCOUNT>0
                        commit;
          end

        end
        GO
        /****** Object:  StoredProcedure [dbo].[spCris_UpdateJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
        SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [dbo].[spCris_UpdateJSON] @pImportID bigint,
                                @pErrTransaction nvarchar(max) out
as

BEGIN

----in prealabil verificare --> sir JSON valid ca structura!!!
declare @json_string nvarchar(max)
declare @institution_id bigint
select @json_string=l.[json_string],@institution_id=i.institution_id
from [LOG].[JSON_Import] l inner join IMP.Import i on l.id=i.json_id
where i.id = @pImportID

-----------------------------------------------------
-----------------------------------------------------
--- comentariu 25.iulie ---------------------------------
/*
declare @sql nvarchar(max) = ';'
select @sql=@sql + 'select ' + cast(t.object_id as varchar(10))+ ' as IdTabel, ''' +
                        + t.name + ''' as NumeTabel,
      json_query(''' + @json_string + ''', ''$."'+ t.name + '"'') as DateJSON;'
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
where s.name = 'dbo' and t.name not in (select [table_name] from [IMP].[IgnoredTables])

----select @sql
```

```
declare @tRezultat table (IdTabel int,
                          NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                      StructuraJSON nvarchar(max),
                                                      StructuraTabel nvarchar(max)
                                                   )
insert @tRezultat (IdTabel,NumeTabel,SirJSON)
exec (@sql)
*/
-------------------------------------------------------------------------
--- cod nou 25.iulie --------------------------------
declare @tRezultat table (IdTabel int,
                          NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                      StructuraJSON nvarchar(max),
                                                      StructuraTabel nvarchar(max)
                                                   )
insert @tRezultat (IdTabel, NumeTabel, SirJSON)
select IdTabel, NumeTabel,SirJSON
from [dbo].[fCris_ExtrageDateJSON](@json_string)
--- sfarsit cod nou-----------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------


delete from @tRezultat where SirJSON is null
----select * from @tRezultat

update @tRezultat
```

```
set StructuraJSON = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraJson'),
        StructuraTabel =[dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel')



declare c cursor
for select NumeTabel, SirJSON, StructuraJSON, StructuraTabel  from @tRezultat



declare @idTabel int
declare @numeTabel nvarchar(max)
declare @sirJSON nvarchar(max)
declare @structuraJSON nvarchar(max)
declare @structuraTabel nvarchar(max)
declare @tmpCampuriTabel table(NumeCamp nvarchar(30))
declare @start datetime = getdate()

open c

BEGIN TRANSACTION;

begin try
   update [IMP].[Import]
        set [ProcessingStartTime]=@start
        where id = @pImportID

        fetch next from c into @numeTabel,@sirJSON, @structuraJSON,@structuraTabel

        while @@FETCH_STATUS=0
                begin
```

```
                delete @tmpCampuriTabel

                    insert @tmpCampuriTabel
                    select value as NumeCamp
                    from string_split(@structuraTabel,',')
                    delete @tmpCampuriTabel where  NumeCamp in ('[id]','[import_ID]')

                    declare @sirCampuri nvarchar(max)=N''
                    select @sirCampuri += t.NumeCamp + '=' + 't1.' + t.NumeCamp + ','
                    from @tmpCampuriTabel t

                    if len(@sirCampuri)>0
                        begin
                            set  @sirCampuri+='import_id='  +  cast(@pImportID  as
varchar(max))
                            declare @sirSQL nvarchar(max) = N';with t1 as ('
                                +     'SELECT     '     +
@structuraTabel
                                + ' FROM OPENJSON(''' +
@sirJson + ''')'
                                +    '    WITH    ('    +
@structuraJson + '))'
                                +    '    UPDATE    '    +
@numeTabel
                                + ' SET ' + @sirCampuri
                                +    '    FROM    '    +
@numeTabel
                                    + ' t2 inner join t1
on t2.id=t1.id inner join IMP.Import i on t2.import_ID=i.id'
                                +         '         WHERE
i.institution_id=' + cast(@institution_id as varchar(max))
                            exec (@sirSQL)
```

```
                                    end

        fetch next from c into @numeTabel,@sirJSON, @structuraJSON,@structuraTabel
    end

        update [IMP].[Import]
        set [ProcessingEndTime]=getdate()
        where id = @pImportID
end try

begin catch
        set @pErrTransaction = 'Error number: ' + convert(varchar(10),ERROR_NUMBER())
                        + 'Severity: ' + convert(varchar(10),ERROR_SEVERITY())
                                                + 'Description: ' + ERROR_MESSAGE()
        --select ERROR_NUMBER() AS errNumber,ERROR_SEVERITY() AS errSeverity,
        --      ERROR_MESSAGE() AS errMsg
  if @@TRANCOUNT>0
                rollback;
end catch

if @@TRANCOUNT>0
  commit;

close c
deallocate c

END
GO
```

```
/****** Object:  StoredProcedure [dbo].[spCris_ValideazaDateJSON]    Script Date: 12/1/2023 4:10:56 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [dbo].[spCris_ValideazaDateJSON] @pImportID bigint,

@pOperationType nvarchar(20),
                                    @pErrData nvarchar(max) out
as
BEGIN


declare @json_string nvarchar(max)
declare @institution_id bigint
select @json_string=l.[json_string],@institution_id=i.institution_id
from [LOG].[JSON_Import] l inner join IMP.Import i on l.id=i.json_id
where i.id = @pImportID
-----------------------------------------------------
if isjson(isnull(@json_string,''))=0  ---> ca verificare minimala, se presupune structura valida
 begin
   set @pErrData = 'JSON invalid'
   return
 end
-----------------------------------------------------

-----------------------------------------------------
--- comentariu 25.iulie ---------------------------------
/*
declare @sql nvarchar(max) = ';'
```

```
select @sql=@sql + 'select ' + cast(t.object_id as varchar(10))+ ' as IdTabel, ''' +
                        + t.name + ''' as NumeTabel,
      json_query(''' + @json_string + ''', ''$.'''+ t.name + ''''') as DateJSON;'
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
where s.name = 'dbo' and t.name not in (select [table_name] from [IMP].[IgnoredTables])

--exec (@sql)
--select @sql

declare @tRezultat table (IdTabel int,
                    NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                      StructuraJSON nvarchar(max),
                                                      StructuraTabel nvarchar(max),
                                                      StructuraTabel_tmp nvarchar(max),
                                                      StructuraTabel_min nvarchar(max),
                                                            EroriPK nvarchar(max),
                                                            EroriFK nvarchar(max),
                                                            EroriNull nvarchar(max)
                                                )
insert @tRezultat (IdTabel,NumeTabel,SirJSON)
exec (@sql)
*/
-------------------------------------------------------------------------
--- cod nou 25.iulie --------------------------------
declare @tRezultat table (IdTabel int,
                    NumeTabel nvarchar(100),
                                                    SirJSON nvarchar(Max),
                                                      StructuraJSON nvarchar(max),
```

```
                                                    StructuraTabel nvarchar(max),
                                                    StructuraTabel_tmp nvarchar(max),
                                                    StructuraTabel_min nvarchar(max),
                                                        EroriPK nvarchar(max),
                                                        EroriFK nvarchar(max),
                                                        EroriNull nvarchar(max)
                                                )
        insert @tRezultat (IdTabel, NumeTabel, SirJSON)
        select IdTabel, NumeTabel,SirJSON
        from [dbo].[fCris_ExtrageDateJSON](@json_string)
        --- sfarsit cod nou-----------------------------------
        ------------------------------------------------------------------
        ------------------------------------------------------------------


        ----------delete from @tRezultat where SirJSON is null  --- pentru a simplifica conditiile ulterioare
        ----select * from @tRezultat
        ------------------------------------------------------------------------
        ------------------------------------------------------------------------

        update @tRezultat
        set StructuraJSON = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraJson'),
           StructuraTabel_min  = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel_min'),
                StructuraTabel_tmp                                                              =
[dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel_tmp')

        ---select * from @tRezultat

        declare @tmpPK table (IdTabel bigint, NumeTabel sysname, ValoarePK bigint, NrAparitiiValPK int)
```

```
	declare @tmpFK table (IdTabel bigint, NumeTabel sysname, CampFK sysname, ValoareFK bigint)
	declare @tmpPKValExistente table (IdTabel bigint, NumeTabel sysname, ValoarePK bigint)
	declare @tmpNotNull table (campNotNull nvarchar(max))
	declare @tmpNull table(campNull nvarchar(max))

	declare c cursor forward_only
	for
	select IdTabel, NumeTabel, SirJSON, StructuraJSON, StructuraTabel_min, StructuraTabel_tmp ----
, StructuraTabel---
	from @tRezultat

	----declare @valid int
	declare @idTabel int
	declare @numeTabel nvarchar(max)
	declare @sirJSON nvarchar(max)
	declare @structuraJSON nvarchar(max)
	declare @structuraTabel nvarchar(max)
	declare @structuraTabel_tmp nvarchar(max)
	declare @structuraTabel_min nvarchar(max)

	open c
	fetch next from c into @idTabel, @numeTabel,@sirJSON,
		@structuraJSON,@structuraTabel_min, @structuraTabel_tmp --,@structuraTabel

	while @@FETCH_STATUS=0
		begin

			;with tmpJSON as (
				select id as ValoarePK, count(id) as NrAparitiiValPK
```

```
from OPENJSON(@sirJson)
with (id bigint)
where id is not null  --> Null e tratat separat
group by id
)
insert @tmpPK (IdTabel,NumeTabel, ValoarePK, NrAparitiiValPK)
select @idTabel, @numeTabel, ValoarePK, NrAparitiiValPK
from tmpJSON


--------- declare @sqlPK varchar(max) = 'select distinct ' + cast(@idTabel as nvarchar(max)) + ','''' + @numeTabel + '''', id FROM ' + @numeTabel + ';'
--------- 10 iulie - modificare pt includere institutie!!!
declare @sqlPK varchar(max) = 'select distinct ' + cast(@idTabel as nvarchar(max)) + ','''' + @numeTabel
                                + ''''', t.id FROM ' + @numeTabel + ' t inner join IMP.Import i on t.import_ID=i.id '
                                                                                                + ' where i.institution_id=' + cast(@institution_id as nvarchar(max))
----------m.sus Select Distinct pt. teste!!! nu ar trebui sa existe duplicate in BD
---------- deja am inserat de m.multe ori acelasi sir JSON

insert @tmpPKValExistente (IdTabel,NumeTabel, ValoarePK)
exec (@sqlPK)



declare @sqlFK varchar(max)=N''
select @sqlFK+='SELECT ' + cast(v.tabelFK_id as nvarchar(max)) + ','''' +
                    +  @numeTabel + ''''','''+v.campFK + ''''','
                    + cast(v.campFK  as nvarchar(max)) +
```

```
                                        ' FROM OPENJSON(''' + @sirJSON + ''') ' +
                                        ' WITH (' + QUOTENAME(v.campFK)  + ' bigint);'
                                        from [dbo].[viewCris3_Test_ListaFK] v
                                                        where v.tabelFK_id=@idTabel
                                if (len(@sqlFK)>0 )
                                begin
                                        ----select @sqlFK
                                        insert @tmpFK(IdTabel, NumeTabel, CampFK, ValoareFK)
                                        exec (@sqlFK)
                                end
        ------------------------------------------------------------------------------------------
        ------------------------------------------------------------------------------------------

        ------------- validare notNull:
                                delete from @tmpNotNull
                                delete from @tmpNull

                                if isnull(@structuraTabel_min,'')<>'' ---and @numeTabel in
('clients','localities','areas')
                                begin
                                        insert @tmpNotNull(campNotNull)
                                        select replace(replace(value,'[',''),']','')
                                        from string_split(@structuraTabel_min,',')
                                        ----select * from @tmpNotNull

                                        declare @sqlNull nvarchar(max) = N'
                                                declare @tmp table (' + @structuraTabel_tmp + ');
                                                INSERT  @tmp ( ' + @structuraTabel_min + ')
                                                SELECT ' + @structuraTabel_min + '
```

```
                                        FROM OPENJSON(''' + @sirJson + ''')
                                            WITH (' + @structuraJson + ');'
                                    select @sqlNull += ' SELECT DISTINCT case when ' + t.campNotNull + ' Is Null then ''' + t.campNotNull + ''' else ''''end' +
                                                                                    ' FROM @tmp ' +
                                                                                    ' WHERE ' + t.campNotNull + ' Is Null;'
                                    from  @tmpNotNull t


                                    insert @tmpNull
                                    exec (@sqlNull)

                                    declare @errNull nvarchar(max)=N''
                                    select @errNull += campNull + ','
                                    from @tmpNull

                                    if len(@errNull)>0
                                    begin
                                        update @tRezultat
                                        set EroriNull =SUBSTRING(@errNull,0,len(@errNull))
                                        where IdTabel=@idTabel
                                    end

                                end

                fetch next from c into @idTabel, @numeTabel,@sirJSON,
            @structuraJSON,@structuraTabel_min, @structuraTabel_tmp --,@structuraTabel
            end
```

```
close c
deallocate c

---select * from @tRezultat
---select * from @tmpSirFK
-------------------------------------------------
----select * from @tmpPK
--select * from @tmpFK
-------------------------------------------------------------------------
-------------------------------------------------------------------------
-------------------------------------------------------------------------
-------------------------------------------------------------------------

---------- validare PK:
declare @tmpTabelValidarePK table (IdTabelPK bigint, ValoarePK nvarchar(50))
if @pOperationType='add'
  begin
    insert @tmpTabelValidarePK (IdTabelPK, ValoarePK)
          select p.IdTabel, p.ValoarePK
          from @tmpPK p
          where p.NrAparitiiValPK > 1

          insert @tmpTabelValidarePK (IdTabelPK, ValoarePK)
          select p.IdTabel, p.ValoarePK
          from @tmpPK p inner join @tmpPKValExistente e on
                                        (p.IdTabel=e.IdTabel and p.ValoarePK=e.ValoarePK)
    where p.NrAparitiiValPK = 1
  end
```

```
if @pOperationType='edit'
  begin
        insert @tmpTabelValidarePK (IdTabelPK, ValoarePK)
        select p.IdTabel, p.ValoarePK
        from @tmpPK p inner join @tmpPKValExistente e on
                                    (p.IdTabel=e.IdTabel and p.ValoarePK=e.ValoarePK)
    where p.NrAparitiiValPK > 1

        insert @tmpTabelValidarePK (IdTabelPK, ValoarePK)
        select p.IdTabel, p.ValoarePK
        from @tmpPK p left join @tmpPKValExistente e on
                                    (p.IdTabel=e.IdTabel and p.ValoarePK=e.ValoarePK)
        where e.IdTabel is null
  end
-------------------------------------------------------------

declare c_PK cursor forward_only
for
select distinct IdTabelPK--, p.NumeTabel
from @tmpTabelValidarePK

open c_PK
fetch next from c_PK into @idTabel---, @numeTabel
while @@FETCH_STATUS = 0
        begin
                declare @listaEroriPK nvarchar(max)=''
                select @listaEroriPK+=convert(nvarchar(max), t.ValoarePK) + ','
                from @tmpTabelValidarePK t
                where t.IdTabelPK=@idTabel
```

```
                    if len(@listaEroriPK)>0
                        begin
                            update @tRezultat
                            set EroriPK=SUBSTRING(@listaEroriPK,0,len(@listaEroriPK))
                            where IdTabel=@idTabel
                        end
                    fetch next from c_PK into @idTabel---, @numeTabel
                end
        close c_PK
        deallocate c_PK

        ---------select * from @tRezultat
        ------------------------------------------------------------------------
        ------------------------------------------------------------------------

        ---------------------- validare FK:
        declare @tmpEroriFK table (IdTabel bigint, NumeTabel sysname, CampFK sysname, ValoareFK
bigint);
        declare @tmpPK_All table (IdTabel bigint, NumeTabel sysname, ValoarePK bigint)

        insert @tmpPK_All (IdTabel, NumeTabel, ValoarePK)     ------- valabil pt insert + update
        select IdTabel, NumeTabel,ValoarePK
        from @tmpPKValExistente

        if @pOperationType = 'add'   ------- doar pt. insert
          insert @tmpPK_All (IdTabel, NumeTabel, ValoarePK)
          select p.IdTabel, p.NumeTabel, p.ValoarePK
```

```
        from @tmpPK p left join @tmpPKValExistente e on (p.IdTabel=e.IdTabel and
p.ValoarePK=e.ValoarePK)
        where e.IdTabel is null

    insert @tmpEroriFK (IdTabel, NumeTabel, CampFK, ValoareFK)
    select distinct f.IdTabel,f.NumeTabel,f.CampFK,f.ValoareFK
    from viewCris3_Test_ListaFK v
        inner join @tmpFK f
            on (v.tabelFK_id=f.IdTabel and v.campFK=f.CampFK)
            left join @tmpPK_All p on (v.tabelPK_id=p.IdTabel and f.ValoareFK=p.ValoarePK)
    where p.ValoarePK is null

    declare @campFK nvarchar(max)
    declare @valoareFK bigint

    declare c_FK cursor forward_only
    for
    select distinct IdTabel, NumeTabel,CampFK
    from @tmpEroriFK

    open c_FK
    fetch next from c_FK into @idTabel, @numeTabel,@campFK
    while @@FETCH_STATUS = 0
      begin
        declare @listaFKValEronate nvarchar(max)=N''
            select @listaFKValEronate+=convert(nvarchar(max), t.ValoareFK )  +','
            from @tmpEroriFK t
            where t.IdTabel=@idTabel and t.CampFK=@campFK
```

```
                if len(@listaFKValEronate)>0
                begin
                        update @tRezultat
                        set EroriFK = isnull(EroriFK,'') + @campFK + '(' +
SUBSTRING(@listaFKValEronate,0,len(@listaFKValEronate)) + ')'
                        where IdTabel=@idTabel

        end
        fetch next from c_FK into @idTabel, @numeTabel,@campFK
    end
close c_FK
deallocate c_FK

---------------------------------------------------------------------------
---------------------------------------------------------------------------
-----select * from @tRezultat


/*
set @pErrData=(select t.NumeTabel As ErrSource,
      t.EroriPK As PKErr, t.EroriFK AS FKErr, t.EroriNull As NullErr
from @tRezultat t
where  (t.EroriPK is not null)  or (t.EroriFK is not null)
                        or (t.EroriNull is not null)
for json auto---, include_null_values
)
*/

set @pErrData = N''
```

```
	select @pErrData += replace(t.NumeTabel + '(' + isnull('#PK:'+t.EroriPK,'')
                                                + isnull('; #FK:'+t.EroriFK,'')
                                                + isnull('; #Null:'+t.EroriNull,'')
                                                + ') ','(; ','(')
	from @tRezultat t
	where (t.EroriPK is not null) or (t.EroriFK is not null)
		or (t.EroriNull is not null)



	set @pErrData = isnull(@pErrData,'')


	----select @pErrData


	END
	GO
	/****** Object:  StoredProcedure [dbo].[spCris_ValideazaSchemaJSON]    Script Date: 12/1/2023
4:10:56 PM ******/
	SET ANSI_NULLS ON
	GO
	SET QUOTED_IDENTIFIER ON
	GO


	CREATE PROC [dbo].[spCris_ValideazaSchemaJSON] @pImportID bigint,
                                @pErrSchema nvarchar(max) out
	as


	BEGIN


	set @pErrSchema=N''
```

```
declare @json_string nvarchar(max)
select @json_string=l.[json_string]
from [LOG].[JSON_Import] l inner join IMP.Import i on l.id=i.json_id
where i.id = @pImportID
-----------------------------------------------------
if isjson(isnull(@json_string,''))=0  ---> ca verificare minimala, se presupune structura valida
 begin
  set @pErrSchema = 'JSON invalid'
  return
 end
-----------------------------------------------------
-----------------------------------------------------
--- comentariu 25.iulie --------------------------------
/*
declare @sql nvarchar(max) = ';'
select @sql=@sql + 'select ' + cast(t.object_id as varchar(10))+ ' as IdTabel, ''' +
                        + t.name + ''' as NumeTabel,
      json_query(''' + @json_string + ''', ''$."'+ t.name + '"'') as DateJSON;'
from sys.tables t inner join sys.schemas s on t.schema_id=s.schema_id
where s.name = 'dbo' and t.name not in (select [table_name] from [IMP].[IgnoredTables])

----select @sql

declare @tRezultat table (IdTabel int,
                        NumeTabel nvarchar(100),
                                                        SirJSON nvarchar(Max),
                                                          StructuraJSON nvarchar(max),
                                                          StructuraTabel nvarchar(max),
```

```
                                                    StructuraTabel_tmp nvarchar(max),
                                                      ErrStructura nvarchar(max)
                                                )
insert @tRezultat (IdTabel,NumeTabel,SirJSON)
exec (@sql)
*/
-----------------------------------------------------------------------
--- cod nou 25.iulie ---------------------------------
declare @tRezultat table (IdTabel int,
                          NumeTabel nvarchar(100),
                                                   SirJSON nvarchar(Max),
                                                    StructuraJSON nvarchar(max),
                                                    StructuraTabel nvarchar(max),
                                                    StructuraTabel_tmp nvarchar(max),
                                                      ErrStructura nvarchar(max)
                                                )
insert @tRezultat (IdTabel, NumeTabel, SirJSON)
select IdTabel, NumeTabel,SirJSON
from [dbo].[fCris_ExtrageDateJSON](@json_string)
--- sfarsit cod nou-----------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------


delete from @tRezultat where SirJSON is null
----select * from @tRezultat
-----------------------------------------------------------------------
-----------------------------------------------------------------------
```

```
	update @tRezultat
	set StructuraJSON = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraJson'),
		StructuraTabel =[dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel'),
		StructuraTabel_tmp = [dbo].[fCris3_Test_DateValidare](NumeTabel,'structuraTabel_tmp')
	-----select * from @tRezultat

	declare c cursor forward_only
	for
	select IdTabel, NumeTabel, SirJSON, StructuraJSON, StructuraTabel, StructuraTabel_tmp from @tRezultat

	declare @idTabel int
	declare @numeTabel nvarchar(max)
	declare @sirJSON nvarchar(max)
	declare @structuraJSON nvarchar(max)
	declare @structuraTabel nvarchar(max)
	declare @structuraTabel_tmp nvarchar(max)

	open c
	fetch next from c into @idTabel, @numeTabel,@sirJSON,
		@structuraJSON,@structuraTabel,@structuraTabel_tmp

	while @@FETCH_STATUS=0
	begin
		declare @sirSQL nvarchar(max) = N'
			declare @tmp table (' + @structuraTabel_tmp + ');
			INSERT  @tmp ( ' + @structuraTabel + ')
			SELECT ' + @structuraTabel + '
```

```
                                        FROM OPENJSON(''' + @sirJson + ''')
                                        WITH (' + @structuraJson + ');'
                --select @sirSQL

                begin try
                        exec (@sirSQL)
            end try

                begin catch
                        declare @errTabel nvarchar(max)=N''
                        exec  [dbo].[spCris_ValideazaSchemaTabel]  @numeTabel,@sirJSON,  @errTabel
out

                        /*
                        if isnull(@errTabel,'')<>''
                           set @pErrSchema += @numeTabel + '(' + @errTabel + ');'
                                */

                        update @tRezultat
                        set ErrStructura =  @errTabel
                        where IdTabel=@idTabel


                end catch
                fetch next from c into @idTabel, @numeTabel,@sirJSON,
                      @structuraJSON,@structuraTabel,@structuraTabel_tmp
        end

        close c
```

```
	deallocate c

	/*
	set @pErrSchema = (select t.NumeTabel As ErrTable, t.ErrStructura as ErrColumns
						from @tRezultat t
						where  t.ErrStructura is not null
						for json auto
					)
	*/


	select @pErrSchema += t.NumeTabel + '(' + t.ErrStructura  + ');'
	from @tRezultat t
	where  t.ErrStructura is not null


	if len(@pErrSchema)>0
		set @pErrSchema = SUBSTRING(@pErrSchema ,0,len(@pErrSchema))

	---select @pErrSchema

	END
	GO
	/****** Object:  StoredProcedure [dbo].[spCris_ValideazaSchemaTabel]    Script Date: 12/1/2023
4:10:56 PM ******/
	SET ANSI_NULLS ON
	GO
	SET QUOTED_IDENTIFIER ON
	GO
```

```
CREATE proc [dbo].[spCris_ValideazaSchemaTabel] @pNumeTabel sysname,
                        @pSirJSON nvarchar(max),
                        @pErrSchema nvarchar(max) out
as
BEGIN

set @pErrSchema=N''
declare @sqlStructuraTabel nvarchar(max)= 'select NumeCamp,NumeCamp_StructuraTabel,NumeCamp_StructuraJSON from dbo.fCris_#StructuraTabel(''' + @pNumeTabel + ''')'

declare @tmpStructuraTabel table (
                        NumeCamp nvarchar(100),
                        NumeCamp_StructuraTabel nvarchar(100),
                        NumeCamp_StructuraJSON nvarchar(100)
                )
insert into @tmpStructuraTabel
exec (@sqlStructuraTabel)

declare cTabel cursor forward_only
for
select NumeCamp, NumeCamp_StructuraTabel, NumeCamp_StructuraJSON
from @tmpStructuraTabel

declare @numeCamp nvarchar(100)
declare @numeCamp_StructuraTabel nvarchar(100)
declare @numeCamp_StructuraJSON nvarchar(200)

open cTabel
```

```
	fetch next from cTabel into @numeCamp,@numeCamp_StructuraTabel, @numeCamp_StructuraJSON


	while @@FETCH_STATUS=0
	begin
			declare @sirSQLTabel nvarchar(max) = N'
					declare @tmp table (' + @numeCamp_StructuraTabel + ');
					INSERT  @tmp ( ' + @numeCamp + ')
					SELECT ' + @numeCamp + '
					FROM OPENJSON(''' + @pSirJson + ''')
					WITH (' + @numeCamp_StructuraJSON + ');'
			--select @sirSQL
			begin try
				exec (@sirSQLTabel)
			end try
			begin catch
				---select @numeTabel, @numeCamp,@sirSQLTabel
				set @pErrSchema+=@numeCamp+','
				--select @errTabel
			end catch
			fetch next from cTabel into @numeCamp,@numeCamp_StructuraTabel,@numeCamp_StructuraJSON
	end
	close cTabel
	deallocate cTabel

	if len(@pErrSchema)>0
		set @pErrSchema = SUBSTRING(@pErrSchema ,0,len(@pErrSchema))
```

```
END

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.areas' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'areas'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.authorizations' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'authorizations'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.bin_collections' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'bin_collections'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.branches' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'branches'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_areas' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_areas'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_brands' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_brands'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_drivers' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_drivers'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_models' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_models'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_provenances' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_provenances'
```

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_services' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_services'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.car_types' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'car_types'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.cars' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'cars'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.clients' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'clients'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.counties' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'counties'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.drivers' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'drivers'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.fuel_vouchers' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'fuel_vouchers'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.incidents' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'incidents'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.localities' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'localities'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.locations' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'locations'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.road_maps' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'road_maps'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.roles' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'roles'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.tags' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'tags'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.tickets' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'tickets'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.user_areas' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'user_areas'

GO

EXEC sys.sp_addextendedproperty @name=N'MS_SSMA_SOURCE', @value=N'medias.users' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'users'

GO