Raport 4
Sistemul informatic integrat

CUPRINS

1 Introducere

Proiectul are ca scop realizarea unui model de laborator pentru un sistem de tip CPSS (Cyber-Physical-Social System) pentru dezvoltare sustenabilă în domeniul furnizării de apă în mediul urban prin implicarea activă a comunității. În implementarea acestuia intervin mai multe scenarii determinate de factori ce țin de infrastructura inteligentă precum și de interacțiunea cu cetățeanul.

Dezvoltarea unor sisteme pilot care să identifice consumurile de apă din interiorul unei gospodarii/apartament și monitorizarea mai frecventă în mai multe puncte de consum asigură suportul pentru identificarea mult mai rapidă a unor incidente în rețeaua de apă și permite predicția cu o precizie mai mare a consumurilor de apă.

Un management optimal al rețelei de furnizare a apei presupune, pe lângă partea de monitorizare și o parte de analiză și interpretare automată a datelor colectate în vederea detectării unor comportamente anormale (detecție de anomalii) și predicția consumurilor viitoare.

Eficiența implementării unui sistem de management a unei rețele de apă depinde în egală măsură de factorii decizionali, dar și de contribuția și implicarea directă a consumatorilor. În acest scop proiectul prezintă un sistem colaborativ de colectare a informațiilor bazat pe două concepte: crowdsensing (colectare de date prin contribuția utilizatorilor) și Serious Gaming (introducerea de recompense pentru contribuțiile individuale, asemănător cu tehnicile utilizate în jocuri). În acest mod utilizatorii devin factori activi în optimizarea consumurilor de apă și în detectarea în timp util a incidentelor din rețea.

Arhitectura sistemului evidențiază componentele funcționale și intercondiționările dintre ele. Proiectarea detaliată a componentelor sistemului și a interacțiunilor dintre acestea au fost concepute sub forma unor servicii accesibile prin interfețe standardizate pentru asigurarea scalabilității și extensibilității sistemului. Detaliile de implementare a componentelor prezintă sistemele hardware și metodele software utilizate.

2 Arhitectura generală a sistemului integrat

Arhitectură generală a sistemului este prezentată în Figura 1.

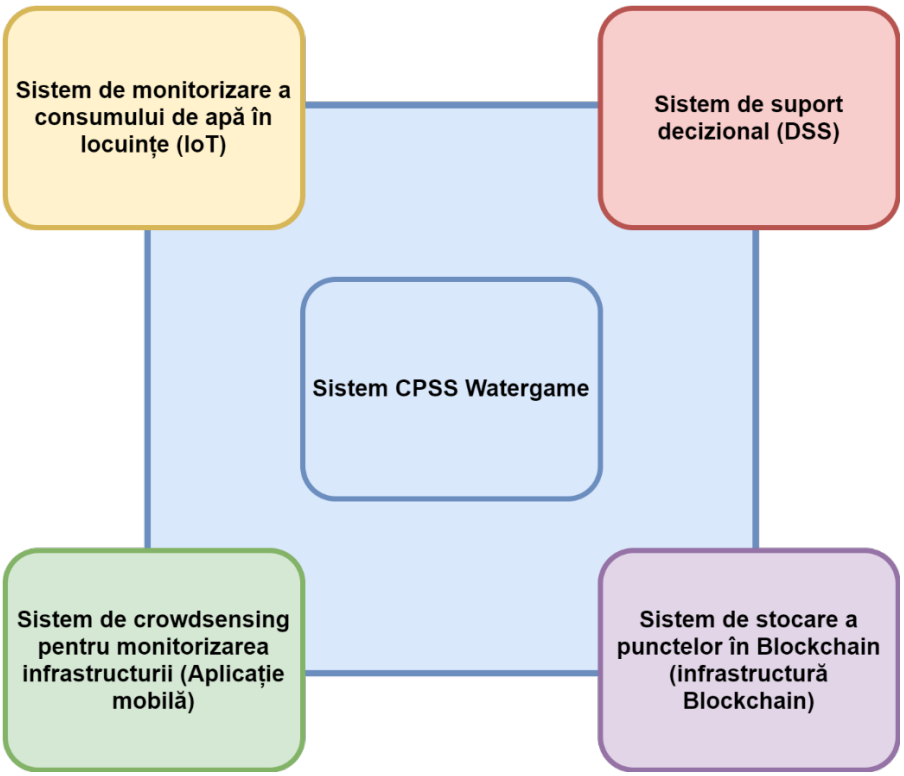


Figura 1 Arhitectura sistemului CPSS

Sistemul este conceput ca un ansamblu de subsisteme independente, care pot interacționa în contextul unei arhitecturi decuplate. Fiind un sistem complex de tip CPSS, poate fi văzut ca un meta-sistem alcătuit din mai multe componente fundamentale:

- sistemul de monitorizare a consumului de apă în locuințe prin intermediul unei arhitecturi de tip IoT;
- sistemul de suport decizional (DSS) ce oferă funcții inteligente de procesare a datelor;
- sistemul de crowdsensing pentru monitorizarea infrastructurii, prin care este implicat factorul social;
- sistemul de stocare a punctelor în Blockchain.

Integrarea acestor componente este posibilă la nivel de cazuri de utilizare, prin sistemul de puncte ce pot fi obținute din raportări sau prin eficientizarea consumului de apă în gospodărie. Sistemul de suport decizional utilizează datele colectate de la senzori pentru a oferi o imagine de ansamblu pentru furnizori și recomandări personalizate pentru consumatori. Sistemul de crowdsensing are la bază o aplicație mobilă prin care se pot raporta incidente în furnizarea apei sau în contextul infrastructurii de apă la nivel urban.

Sunt prezentate în continuare aceste sub-sisteme din perspectiva arhitecturii și a tehnologiilor folosite, fiecare sub-sistem fiind reprezentat printr-o arhitectură și un set de tehnologii specifice.

2.1 Arhitectura componentei de monitorizare a consumului (WMS)

Sistemul de monitorizare a consumului de apă poate fi descris prin următoarele scenarii generale:

- Instalarea și configurarea senzorilor la rețeaua de apă a cetățeanului;
- Colectarea și trimiterea datelor către server în timp real;
- Oferirea unei imagini de ansamblu asupra consumului de apă al cetățenilor, cu scopul de a realiza recomandări personalizate.

Pentru definirea cazurilor de utilizare (a consumului de apă în locuințe) au fost considerate următoarele componente și interacțiuni:

- WMS App - aplicație monitorizare consum;
- IoT Network - rețea de senzori;
- Data Service - serviciu colectare a datelor de consum;
- Firebase Authentication - autentificare utilizatori;
- Firebase Storage – serviciu de stocare a datelor de la utilizatori.

Există mai multe tipuri de utilizatori considerați în sistemul de monitorizare a consumului de apă: consumator (Customer) și furnizor (Supplier), precum și modulele de achiziție a datelor de consum (IoT Module) și tehnicianul (Engineer). Consumatorul și furnizorul interacționează cu sistemul de monitorizare prin intermediul aplicației.

Astfel, aceștia se pot autentifica având roluri diferite și funcționalități specifice. Consumatorul poate să vizualizeze datele de la senzorii asociați, în timp ce furnizorul poate vizualiza datele de la toți senzorii instalați în locuințe. În plus, furnizorul poate realiza asocierea dintre senzori / module de achiziție și consumatori.

Funcțiile de autentificare și raportare (trimitere și vizualizare rapoarte) sunt procesate de serviciul Firebase. Datele de la senzori sunt colectate prin intermediul modulelor IoT care se conectează prin MQTT (Message Queuing Telemetry Transport) la serviciul de date.

Serviciul de date permite monitorizarea consumului de apă în timp real precum și colectarea acestora de la modulele de achiziție. Configurarea modulelor se realizează de către tehnician și include stabilirea conexiunii la rețeaua locală de WiFi și configurarea debitmetrelor atașate.

Arhitectura de colectare și procesare a datelor

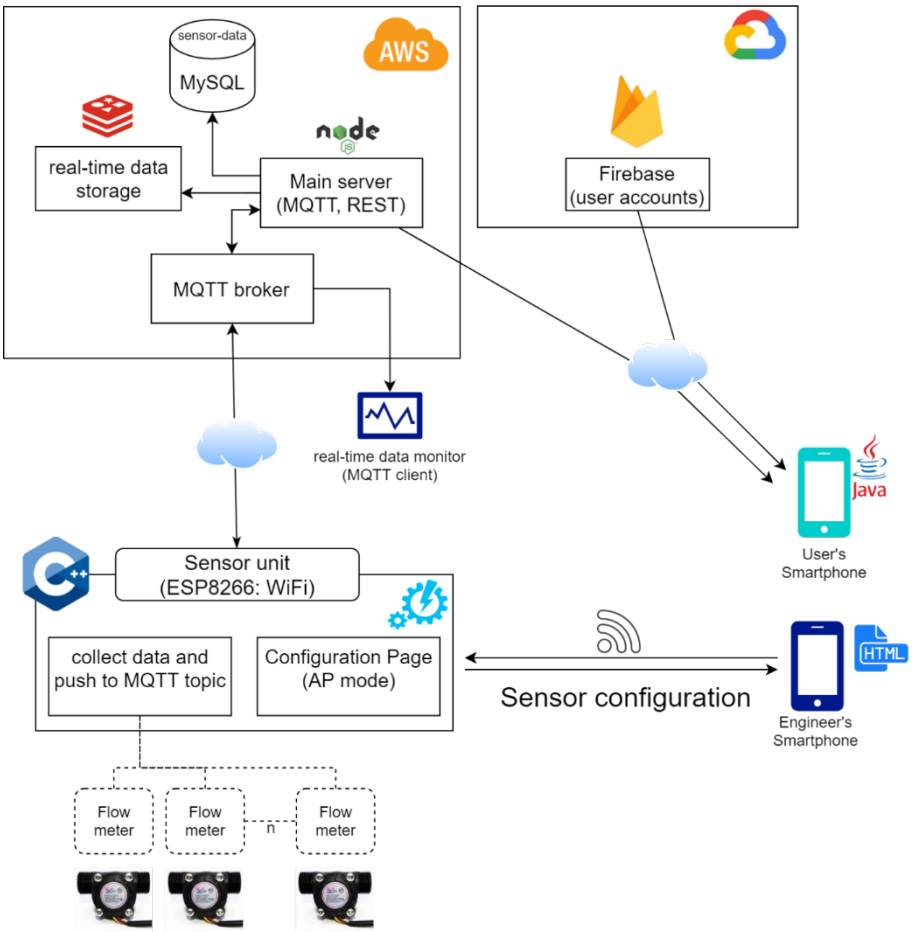


Figura 2 Arhitectura sistemului de monitorizare a consumului de apă în locuințe (UPB.C)

Varianta UPB pentru colectarea eficientă a datelor legate de consumul de apă (**UPB.C**) descrisă în Figura 2 presupune că rețeaua de senzori este integrată într-o arhitectură bazată pe servicii Cloud.

Astfel, achiziția datelor se realizează cu ajutorul unei plăci de dezvoltare NodeMCU bazată pe microcontrolerul ESP8266 care dispune de comunicare WiFi, pini GPIO (General-purpose input / output) și poate fi programat în mediul Arduino (Parihar, 2019). O astfel de placă de dezvoltare poate fi conectată la mai multe debitmetre, monitorizând astfel debitul de apă prin mai multe conducte. Modulul de achiziție astfel proiectat, este pre-programat și dispune de o interfață de configurare ce permite conectarea la rețeaua locală și atașarea unui număr variabil de debitmetre. Interfața este expusă printr-un server găzduit direct pe microcontrolerul ESP8266 configurat în același timp în modul stație și client WiFi.

Colectarea datelor se realizează printr-un broker de mesaje MQTT, ce permite o arhitectură decuplată printr-un protocol standard de comunicație, adaptat pentru domeniul IoT (Mishra & Kertesz, 2020). MQTT este un protocol de comunicație de tip publish-subscribe ce suportă conexiune bi-direcțională peste TCP / IP, fiind ideal pentru schimbul de mesaje între dispozitive IoT (IBM, 2021). Dispozitivele IoT se pot astfel conecta la broker-ul de mesaje ce are rol de rutare a

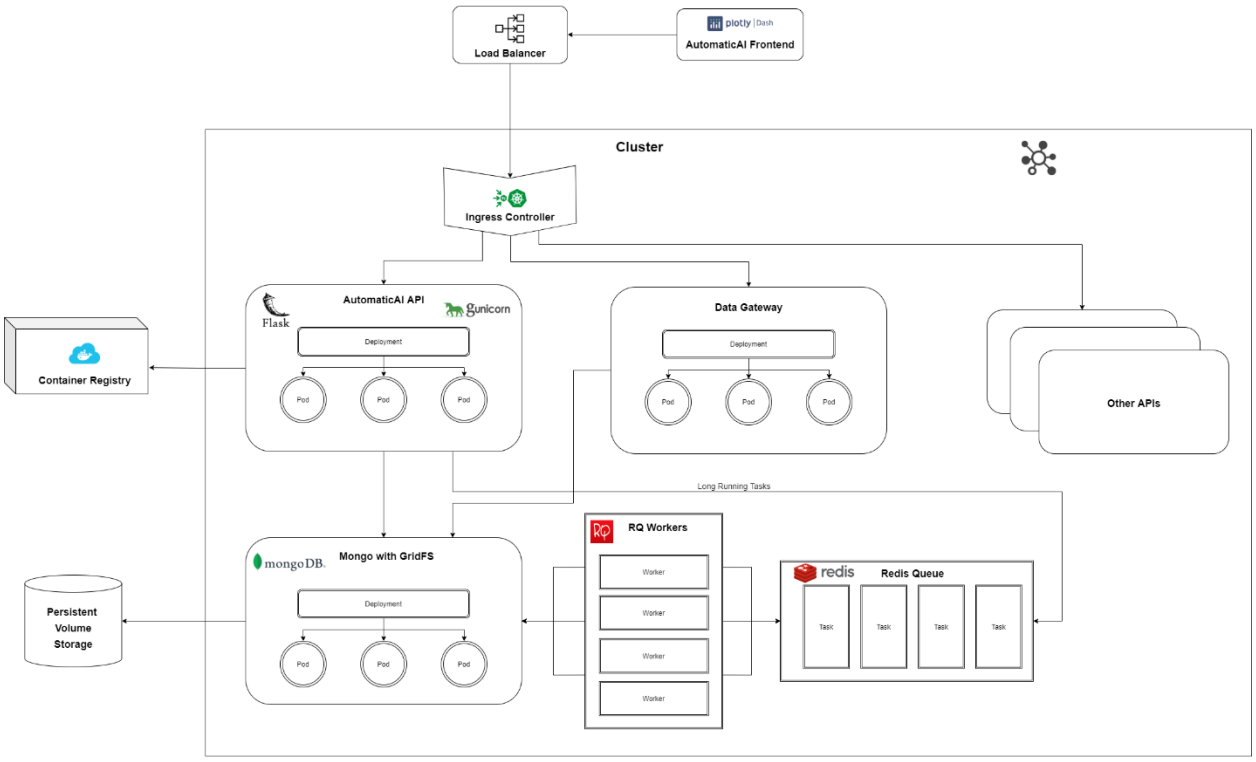
mesajelor primite de la clienți, și pot publica mesaje pe un topic sau se pot abona la un topic pentru a primi mesaje.

Server-ul aplicației se conectează la broker-ul MQTT pentru a recepționa datele trimise de senzori și realizează interfața cu baza de date MySQL[1] (Bell, 2016). Datele colectate sunt salvate periodic în baza de date, conform cu intervalul configurat la nivel de dispozitiv IoT. Există două topice pentru transmiterea datelor: date de consum instantaneu (monitorizare în timp real, cu frecvență ridicată), date de volum (înregistrate în baza de date, cu o frecvență mai scăzută). Datele pot fi accesate în timp real folosind un client MQTT (e.g. aplicație desktop, module IoT), conectarea fiind permisă pe bază de credențiale de acces la nivel de broker.

Pentru disponibilitatea datelor pentru vizualizare în timp real, este adăugat un modul REDIS[2] pentru stocarea temporară a ultimelor măsurători colectate pe topicul de consum instantaneu (Chen et al., 2016).

Datele legate de consum și starea senzorilor sunt prezentate către client cu ajutorul unei aplicații mobile (Android). Autentificarea și gestionarea conturilor pentru utilizatori se realizează prin serviciul Firebase[3], iar încărcarea datelor de consum se realizează prin cereri HTTP de la server-ul principal.

Sistemul de procesare și analiză a datelor



[1] https://www.mysql.com/

[2] https://redis.io/

[3] https://firebase.google.com/

Figura 3 Arhitectura detaliată a platformei extinse AutomaticAI

După cum se poate vedea în Figura 3, există un site web implementat în Dash Plotly care comunică cu aplicația bazată pe microservicii printr-un load balancer. Folosind acest load balancer, se decuplează clientul de clustere și se expune un IP public prin care serviciile din cluster pot fi apelate de clienți. Deoarece request-urile diferite pot ajunge la diferite servicii pe diferite noduri sau mașini virtuale, a fost folosit un controler de intrare (ingress controller) NGINX, care funcționează ca un proxy invers ascunzând diferitele mașini virtuale de la client și, de asemenea, rutează cererile pe baza sub-domeniului furnizat în adresa URL a request-ului.

În Figura 3, unul dintre cele mai importante microservicii este AutomaticAI API. Acest serviciu conține logica pentru configurarea unui pipeline pentru rezolvarea problemelor de clasificare, regresie sau de detectare a anomaliilor. Pipeline-ul poate fi configurat pe baza unui json de configurare. Acest serviciu conține, de asemenea, algoritmul PSO-SA, care poate selecta automat un tip de algoritm AI și poate regla hiper-parametrii acestuia doar pe baza datelor de intrare, fără intervenție umană.

Selectarea, reglarea și antrenarea unui algoritm AI este un proces consumator de timp și de lungă durată. Pentru a preveni blocarea API-ului, se rulează toate procesele de lungă durată în paralel și asincron. Pentru a obține o paralelizare adevărată și o procesare asincronă, a fost aplicat modelul producător-consumator, în care se folosește o coadă Redis pentru a programa sarcinile care trebuie procesate offline de către procesele de lucru numite workers.

Pentru stocarea fișierelor de date de intrare și a modelelor antrenate se folosește MongoDB cu GridFS. GridFS este utilizat pentru stocarea și preluarea fișierelor mari. Aceste fișiere sunt împărțite în bucăți și fiecare bucată este stocată ca document separat. De asemenea, se stochează metadate despre bucăți, astfel fișierul original putând fi reconstruit oricând.

Serviciul numit AutomaticAI API prezintă mai multe module, fiecare având responsabilități diferite. Modulul principal este numit App, care face legătura între diferitele componente și module ale aplicației și definește rutele prin care request-urile pot ajunge la componenta care poate să le trateze.

Aplicația conține mai multe controllere:

1) UsersAPI – conține endpointuri de CRUD pentru utilizatori (adică metode de GET, POST, PUT și DELETE);
2) DatasetsAPI – conține endpointuri de CRUD pentru seturi de date;
3) AIModelAPI – endpoint pentru a citi modelul antrenat și salvat în baza de date;
4) TrainAPI – endpoint pentru declanșarea antrenării modelului selectat;
5) JobsAPI – pentru procesele de lungă durată se generează un job care se rulează în spate. Acest modul este folosit pentru a vizualiza job-urile și statusurile lor (e util pentru depanare);
6) ModelSelectionAPI – conține un endpoint care declanșează rularea algoritmului de selecție și optimizare automată a modelelor AI, numit PSO-SA;
7) PipelineAPI- modulul cel mai complex, folosit pentru a crea un pipeline configurabil, așa cum este descris în secțiunea următoare;
8) LstmAPI – conține un endpoint pentru crearea și configurarea unui model LSTM;
9) AutoArimaAPI - conține un endpoint pentru crearea și configurarea algoritmului Auto-Arima;

10) EvaluateAPI – conține diferite funcții pentru evaluarea algoritmilor de clasificare (acurateţe, precizie, scor f1, matrice de confuzie etc.) și de regresie (MAE, RMSE etc.);
11) TimeSeriesEvaluationAPI – conține diferite funcții pentru evaluarea algoritmilor folosiți pentru analiza, procesarea seriilor de timp și de predicție a valorilor următoare.

Sistemul de detecție a anomaliilor și de predicție

Serviciile de detecție a anomaliilor și de predicție (UTCN.A1 și UTCN.A2) sunt oferite prin intermediul platformei numite AutomaticAI (Czako, Sebestyen & Hangan, 2021). Dezvoltarea acestei platforme a fost realizată în cadrul unui proiect anterior și cuprinde toate metodele și algoritmii necesari pentru a configura și regla un algoritm de detectare a anomaliilor de la zero. Platforma conține algoritmi de preprocesare, cum ar fi diferite tipuri de transformare a datelor sau metode de scalare a datelor, algoritmi de selectare a caracteristicilor și de reducere a dimensionalității, cum ar fi PCA, LDA și alții, algoritmi de clasificare și regresie și, de asemenea, o mare varietate de metode de vizualizare a datelor și a rezultatelor. Arhitectura de nivel înalt al acestei platforme se regăsește în Figura 4.

Pentru procesarea și analiza datelor de consum de apă și pentru detecția anomaliilor în calitatea apei s-a extins platforma existentă prin adăugarea unei funcționalități extrem de utile legate de posibilitatea de a crea și configura diferite pipeline-uri de procesare și analiză a datelor de intrare. Folosind această opțiune a fost creată soluția pentru detecția anomaliilor în calitatea apei, construind un pipeline care conține mai muți pași de preprocesare și un model de AI pentru clasificare.

Pe lângă acest pipeline dinamic, pentru a avea funcționalități de analiză și procesare a seriilor de timp, au fost adăugate module noi pentru vizualizarea datelor de tip serii de timp, pentru configurarea modelelor de predicție, vizualizarea rezultatelor și evaluarea modelelor antrenate. Pentru predicție a fost adăugat algoritmul Auto-ARIMA (Yermal & Balasubramanian, 2017) și posibilitatea de a configura diferite arhitecturi de rețele neuronale recurente de tip LSTM (Lu et al., 2020), (Xingjian et al., 2015).

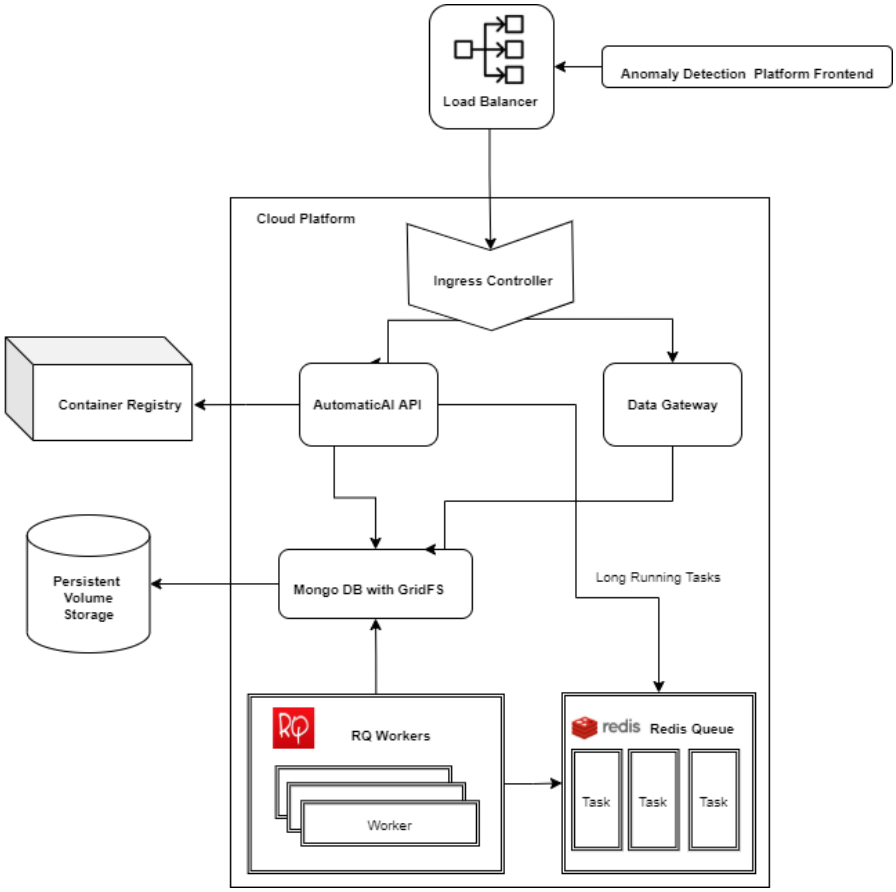


Figura 4 Arhitectura de nivel înalt a platformei AutomaticAI

Toate aceste module și funcționalități noi au fost adăugate în serviciul AutomaticAI și a fost creată și o parte de interfață cu utilizatorul (UI) pentru folosirea cât mai ușoară a acestora.

Sistemul de notificări și alerte

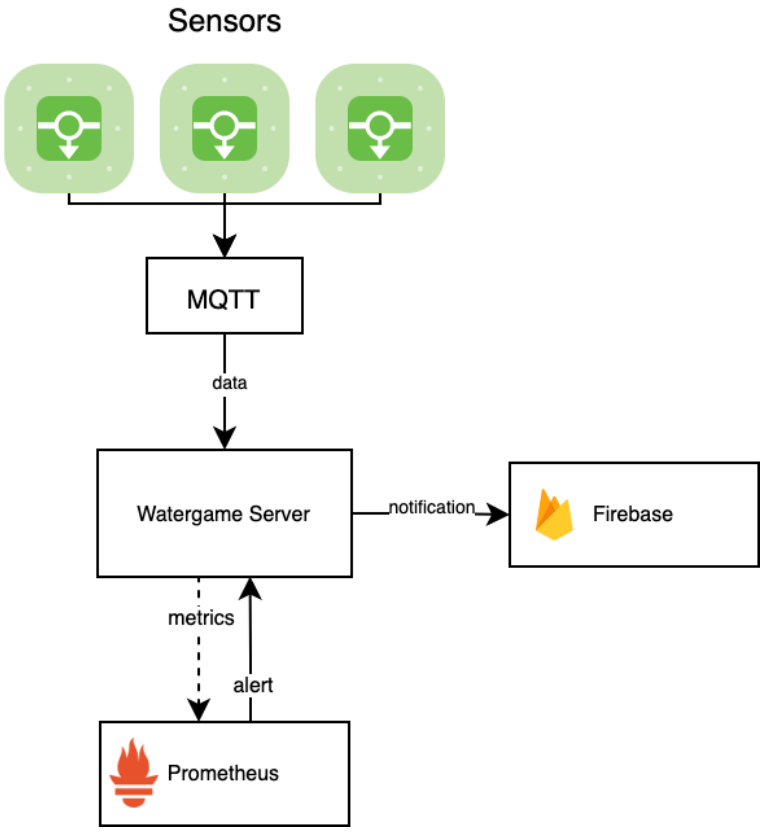


Figura 5 Sistemul de notificări

Sistemul de notificări, reprezentat în este integrat în server-ul aplicației de colectare a datelor de la senzori și permite notificarea utilizatorilor aplicației WaterMonitoringSystem cu privire la starea de funcționare a senzorilor.

Prometheus îndeplinește funcția de monitorizare si alertare, folosindu-se de metricile puse la dispoziție de server. Alertele sunt tratate de server salvând notificări corespunzătoare in baza de date Firebase.

2.2 Arhitectura componentei de suport decizional (DSS)

Sistemul de suport decizional presupune analiza datelor și apoi utilizarea arborilor și a sistemelor de decizie pentru a oferi recomandări în vederea optimizării consumului și a costurilor implicate, precum și educarea utilizatorilor. De asemenea, acest sistem este responsabil de realizarea de diverse vizualizări a consumatorilor pentru a ușura luarea de decizii.

Pipeline-ul de procesare folosit în scenariile UPB.A1 - UPB.A3 (Figura 6) a fost implementat în Python folosind pachetul scikit-learn (Pedregosa et al., 2011; scikit-learn developers (BSD License)., n.d.) pentru metodele de clustering (e.g. K-Means și OPTICS), în timp ce pre-procesarea implică

pachetul statsmodels[4] pentru analiza seriilor temporale (adică descompunerea sezonieră) și *numpy* pentru normalizarea datelor.

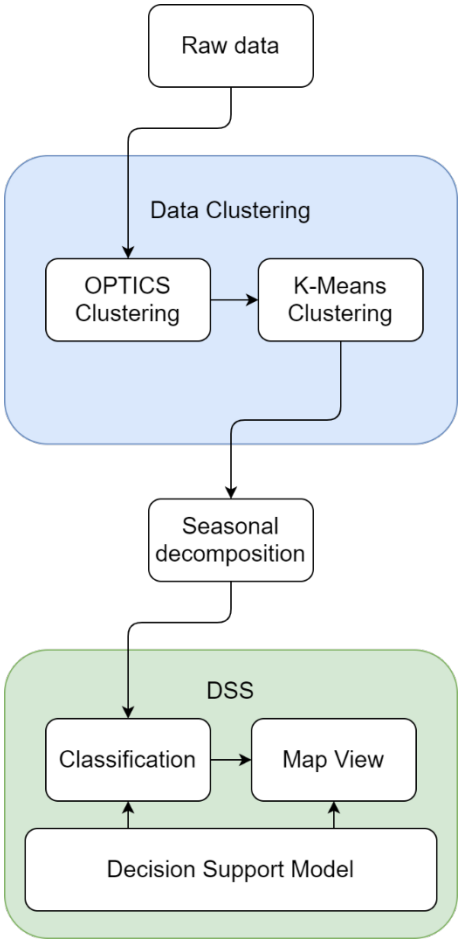


Figura 6 Proiectarea componentei de suport decizional folosită în scenariile UPB.A1- UPB.A3

2.2.1 Clustering pe bază de date de consum folosind K-Means

Metoda de clustering K-Means a fost validată în prealabil pe setul de date Helix (Chatzigeorgakidis, 2018) pentru a împărți consumatorii în grupuri similare în funcție de consumul de resurse de apă, în timp ce diferite metode de pre-procesare au fost utilizate pentru a crește nivelul de detaliu în ceea ce privește comportamentele identificate ale acestora. Pipeline-ul de clusterizare este prezentat în Figura 7.

[4] https://www.statsmodels.org/stable/index.html

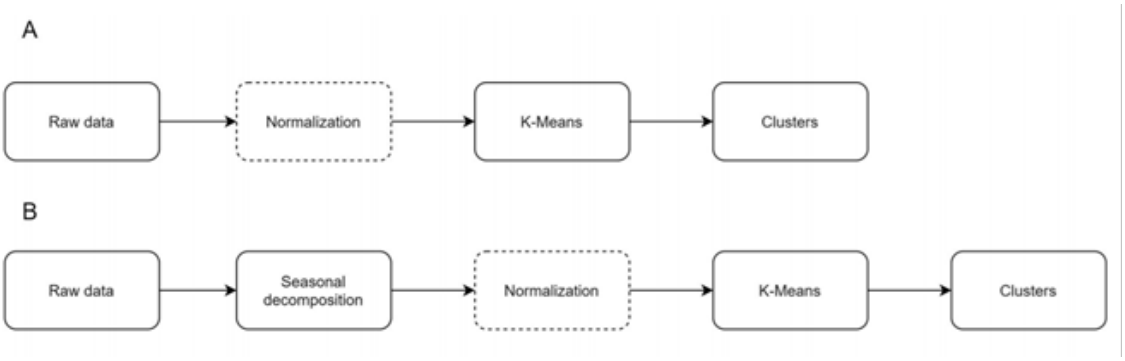


Figura 7 Pipeline clusterizare

Prin analiza de tip clustering, se poate oferi o imagine de ansamblu asupra consumului de apă în contextul realizării un DSS (Decision Support System) pentru un management durabil al resurselor de apă în viitor.

Un prim tip de informație ce poate fi generat este modul în care se grupează consumatorii de apă în funcție de consumul lor zilnic. Astfel, pentru a împărți setul de date în grupuri cu consum similar de apă, am folosit metoda de grupare K-Means (Syakur M. et al., 2018). Mai mult, pentru a determina numărul optim de clustere, au fost folosite două metode:

- Prima metodă utilizată este metoda Elbow deoarece rezultatele sunt promițătoare pentru multiple aplicații în diverse domenii (Poyrazoglu G., 2021). Această analiză se bazează pe WCSS (Within-Cluster Sum of Squares), unde se calculează suma distanței pătrate dintre fiecare membru al clusterului și centroidul clusterului (Nair A., 2022).
- A doua metodă utilizează analiza siluetei clusterelor (Rousseeuw P., 1987), care evaluează separarea dintre clustere și permite atât inspecția vizuală, cât și evaluarea numerică folosind scorul siluetei.

Prin urmare, cele două metode utilizate reprezintă un suport pentru luarea deciziilor privind alegerea numărului optim de clustere. Datele au fost procesate folosind Python, pachetul scikit-learn (Pedregosa F., 2011) și NumPy (Harris C., 2020).

2.2.2 Metode de clasificare

Această subsecțiune prezintă principalele metode experimentale concepute pentru determinarea sursei unui eveniment de consum de apă, care se caracterizează prin durată (măsurată în minute) și volumul de apă consumat (măsurat în litri).

Au fost folosiți patru algoritmi de clasificare, doi cu o abordare de învățare automată: Decision Tree și Random Forest, și doi cu deep learning: Dense Neural Network și Recurrent Neural Network.

Arborele decizional

Învățarea automată este o ramură importantă a inteligenței artificiale. Este împărțită în diferite clase de algoritmi. În ceea ce privește modelele de clasificare, acestea pot lua forma unei structuri de decizie printr-un clasificator de tip arbore de decizie.

Pașii pentru construirea unui arbore de decizie sunt următorii:

- Setul de date este împărțit în subseturi mai mici;
- Se asociază un nod atributului care are cea mai bună diviziune;
- Pentru fiecare nod, se adaugă atâtea ramuri câte valori distincte are atributul.

Pentru ca nodul să devină o frunză, majoritatea observațiilor dintr-un subset trebuie să aibă aceeași clasă. Seturile de date liniare folosesc adesea arbori de decizie deoarece sunt eficienți și rapizi. Prin urmare, am ales să folosim arbori de decizie și în studiul nostru.

Modelul Random Forest

Pentru a obține o mai bună acuratețe a rezultatelor, s-a ales și testarea algoritmul Random Forest pe setul de date. Acest algoritm își propune să combine mai mulți arbori de decizie pentru a obține un rezultat mai precis. Având în vedere că cei doi algoritmi de învățare automată prezentați (Decision Tree și Random Forest) folosesc date categorice, s-a folosit o reprezentare întreagă pentru maparea valorilor țintă. Modelele de învățare profundă, pe de altă parte, necesită codificare one-hot, folosind o funcție de pierdere adecvată pentru clasificarea multi-clasă.

Rețeaua neuronală densă

Un eșantion de intrare pentru analiza propusă este reprezentat de un vector unidimensional având două caracteristici (volum și durată). Având în vedere această proprietate, cea mai comună soluție de deep learning pentru problema actuală de clasificare este o arhitectură Dense Neural Network. Blocul de bază al unui astfel de model constă dintr-un strat liniar, urmat de o funcție de activare. Termenul „dens” folosit în numele modelului provine din proprietatea matematică a unui strat liniar de mapare a fiecărui element de intrare la toate elementele de ieșire printr-o combinație liniară. Această proprietate este în opoziție cu alte funcții parametrice aplicate în mod obișnuit în învățarea profundă care funcționează pe porțiuni ale intrării (de exemplu, convoluția). Pentru soluția de față, modelul dens este descris de trei blocuri intermediare de acest fel (cu ReLU ca funcție de activare) și un strat liniar final asociat cu o funcție softmax pentru îndeplinirea obiectivului de clasificare. Deoarece datele sunt extrase din relativ puține surse, după fiecare bloc dens intermediar a fost inserat și un modul de regularizare a abandonului pentru a preveni supra-adaptarea (overfitting).

Rețeaua neuronală recurentă (RNN)

Al doilea model neuronal care a fost utilizat se bazează pe un bloc LSTM (Long Short-Term Memory) (Hochreiter S. et al., 1997) ca strat de bază. În această configurație, există trei porți: intrare, ieșire și uitare. Modelul primește intrarea curentă (xt), starea ascunsă anterioară ($ht-1$) și starea anterioară a memoriei ($ct-1$) la fiecare pas de timp. Ieșirile constau din starea ascunsă curentă (ht) și starea curentă a memoriei (ct).

Expresiile pentru fiecare poartă (adică intrarea it, ieșirea ot și uitarea ft) și, precum și cele pentru starea ascunsă curentă (ht) și starea de memorie curentă (ct) sunt următoarele:

$$
\begin{aligned}
i_t &= \sigma_s(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_s(W_o x_t + U_o h_{t-1} + b_t) \\
f_t &= \sigma_s(W_f x_t + U_f h_{t-1} + b_f) \\
\tilde{c}_t &= \sigma_h(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t * \tilde{c}_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

unde:

- x_t este termenul de intrare la pasul de timp t;
- h_t și h_{t-1} sunt următoarea stare ascunsă, respectiv starea anterioară ascunsă;
- $\tilde{c}_t$ este vectorul de activare a intrării celulei;
- c_t este starea curentă a memoriei, iar c_{t-1} este starea anterioară a memoriei;
- W_i, W_o și W_f sunt ponderile pentru starea curentă de intrare a fiecărei porți, în timp ce U_i, U_o și U_f sunt ponderile pentru starea anterioară ascunsă;
- b_i, b_o, b_f și b_c sunt vectorii de polarizare;
- σ_s este funcția de activare a sigmoidului;
- σ_h este funcția de activare a tangentei hiperbolice;
- $\odot$ operator este produsul Hadamard.

2.3 Arhitectura componentei de Crowdsensing (CS)

Arhitectura componentei de crowdsensing (Figura 8) are la bază servicii Cloud și o aplicație web interactivă, adaptată formatului mobil. Spre deosebire de celelalte componente, factorul uman este central în funcționarea aplicației pentru raportarea incidentelor pe hartă. Aplicația permite plasarea de indicatori pe hartă în mod interactiv, pe bază de geolocație, cu elemente de joc proiectate pentru stimularea participării.

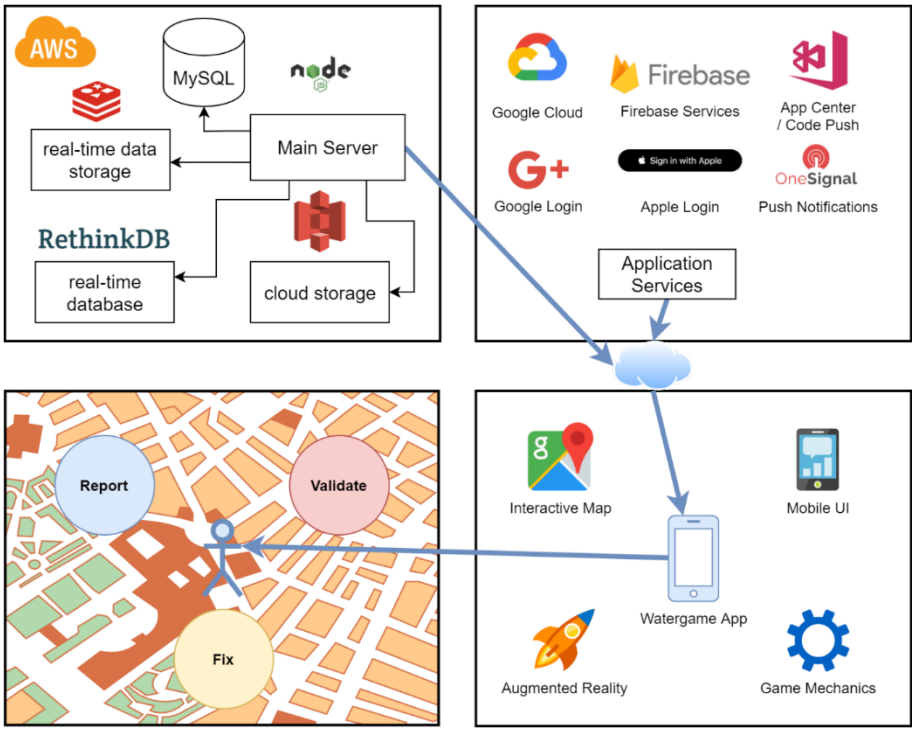


Figura 8 Arhitectura sistemului de raportare a incidentelor pe bază de crowdsensing

Infrastructura Cloud este găzduită pe AWS (Amazon Web Services) reprezentată printr-un server de aplicație care gestionează cererile HTTP de la aplicația mobilă precum și logica de business pentru accesarea bazei de date și a sistemelor de stocare a datelor temporare. Sistemul operează cu o bază de date MySQL ce asigură persistența datelor și o structură bine definită pentru operabilitate. Pentru scalabilitate, sistemul incorporează și o memorie temporară de tip REDIS, în timp ce datele multimedia sunt stocate într-un serviciu de stocare cloud (Amazon S3).

Arhitectura jocului este reprezentată de cele 3 scenarii specifice fiecărui tip de utilizator (Finder, Fixer, Validator) și are la bază harta interactivă configurată pentru a integra elementele de joc propuse.

2.4 Arhitectura componentei Blockchain (B)

Arhitectura componentei Blockchain are la bază o rețea de test Ethereum și o aplicație web funcțională în condiții de laborator. Interacțiunea cu rețeaua blockchain se realizează prin intermediul aplicației sau prin intermediul unui wallet conectat direct (e.g. MetaMask (MetaMask, 2021)). Pentru a sincroniza wallet-urile cu conturile de utilizator din sistem, sistemul va fi conectat indirect (prin server-ul de aplicație, folosind cereri HTTP) la baza de date MySQL de pe AWS.

Rețeaua de test este pusă la dispoziție de platforma Ethereum și presupune o serie de noduri conectate între ele în mod peer-to-peer, care oferă posibilitatea introducerii datelor în blockchain, validării tranzacțiilor și interogării datelor (Neto, 2020).

Aplicația web prin intermediul căreia se poate tranzacționa token-ul (WGT – Water Game Token) este instalată folosind NPM Lite Server[5]. Soluția include diverse validări pentru operațiunile ce se doresc a fi efectuate (donare de token, transfer token) prin intermediul JavaScript și web3.js. De asemenea, prin intermediul Web3.js se realizează conexiunea la blockchain și interacțiunea cu Smart Contracts (Ethereum Revision a57dd3c5, 2016).

Arhitectura soluției oferă posibilitatea tranzacționării token-ului generat și din afara soluției, prin intermediul alternativelor precum: exchange-uri sau platforme de tranzacționare de tokeni, dacă se dorește trecerea monedei într-un mediu de producție.

Metamask si Web3.js

Pentru facilitarea interacțiunii utilizatorului cu soluția blockchain s-a folosit Web3.js[6] împreună cu Metamask (MetaMask, 2021) (vezi Figura 9). Metamask va fi folosit ca un gateway și oferă posibilitatea comunicării cu aplicații de tip blockchain, dar poate fi folosit și ca wallet. Metamask oferă posibilitatea de administra mai multe conturi / chei de conturi, de a tranzacționa criptomonede într-un mod securizat și a accesa aplicații descentralizate prin intermediul unei extensii ce poate fi instalată la nivelul unui browser web sau prin intermediul unei aplicații mobile.

Web3.js reprezintă o colecție de biblioteci JavaScript prin intermediul cărora se facilitează interacțiunea cu un nod din rețeaua blockchain.

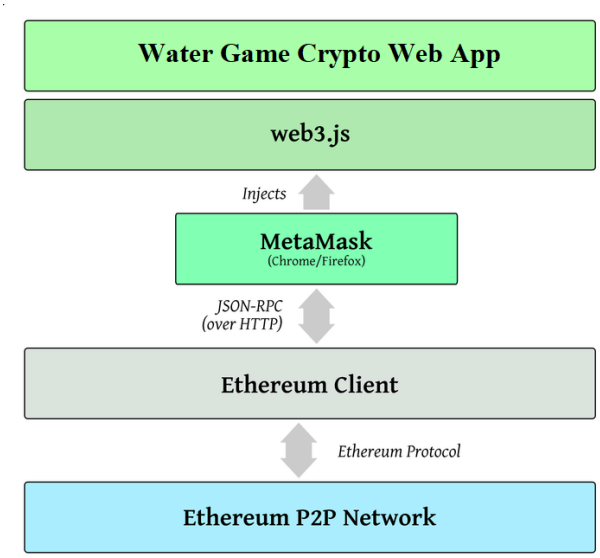


Figura 9 Proiectarea componentei blockchain

3 Integrarea sistemului

Integrarea sistemului este ilustrată prin diagrama de activități din Figura 10. Componentele sistemului au fost dezvoltate independent și interacționează în contextul unor activități bine definite.

[5] https://www.npmjs.com/package/lite-server

[6] https://web3js.readthedocs.io/en/v1.5.2/

Astfel, sistemul WMS – monitorizare a consumului de apă, se integrează cu sistemul DSS – suport decizional, la nivel de date, iar sistemul DSS se integrează cu sistemul Blockchain tot la nivel de date. Sistemul CS – raportare urbană, se integrează cu sistemul Blockchain la nivel de aplicație, cu privire la recompensarea interacțiunilor. Integrarea dintre DSS și CS este indirectă, prin intermediul monedei virtuale emise la nivelul sistemului Blockchain, neexistând altfel interacțiuni directe între cele două componente.

Figura 10 mai evidențiază corespondența dintre componentele sistemului informatic integrat și aplicațiile dezvoltate.

Astfel, aplicația WaterMonitoringSystem – WMS App (UPB) realizează interfața cu utilizatorul, i.e., consumatorul din rețeaua de distribuție a apei, și interacționează cu sistemul de monitorizare (WMS) și de suport decizional (DSS). Aplicația WaterMonitoringSystem preia datele de la sistemul de monitorizare a consumului de apă, și implementează vizualizarea pe hartă a senzorilor instalați în locuințe, precum și consumul de apă în timp real.

Aplicația AutomaticAI – WMS App (UTCN) realizează interfața cu operatorul de date și interacționează cu sistemul de suport decizional (DSS), cu privire la introducerea, procesarea, și validarea datelor. Sistemul DSS presupune un set de algoritmi de procesare a datelor, care sunt apelabili din interfață și implementează scenariile propuse pentru extragerea informațiilor relevante în contextul metodelor de suport decizional.

Aplicația Watergame – Watergame Crowdsensing App (UPB), realizează interfața cu utilizatorul care poate raporta incidente în contextul infrastructurii de apă la nivel urban, și interacționează cu sistemul Blockchain la nivel de aplicație. Integrarea presupune accesarea contului din rețeaua blockchain și realizarea de operațiuni cu moneda virtuală WGT (Watergame Token).

Aplicația Watergame Token App (UPB) prezintă o interfață de control pentru gestionarea contului utilizatorului în rețeaua blockchain, cu funcționalități specifice.

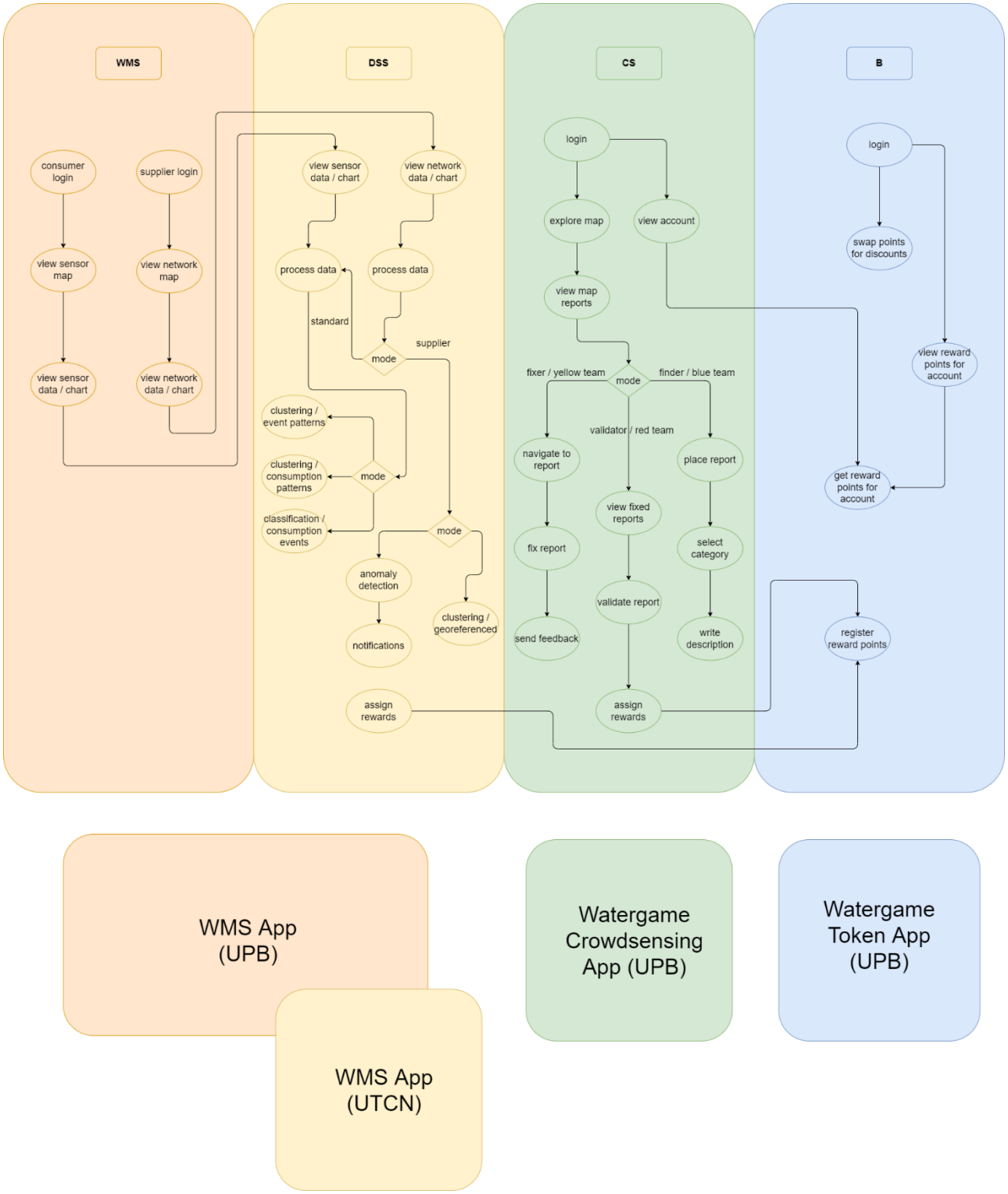


Figura 10 Diagrama de activități a sistemului integrat

3.1 Integrarea componentelor de monitorizare și suport decizional

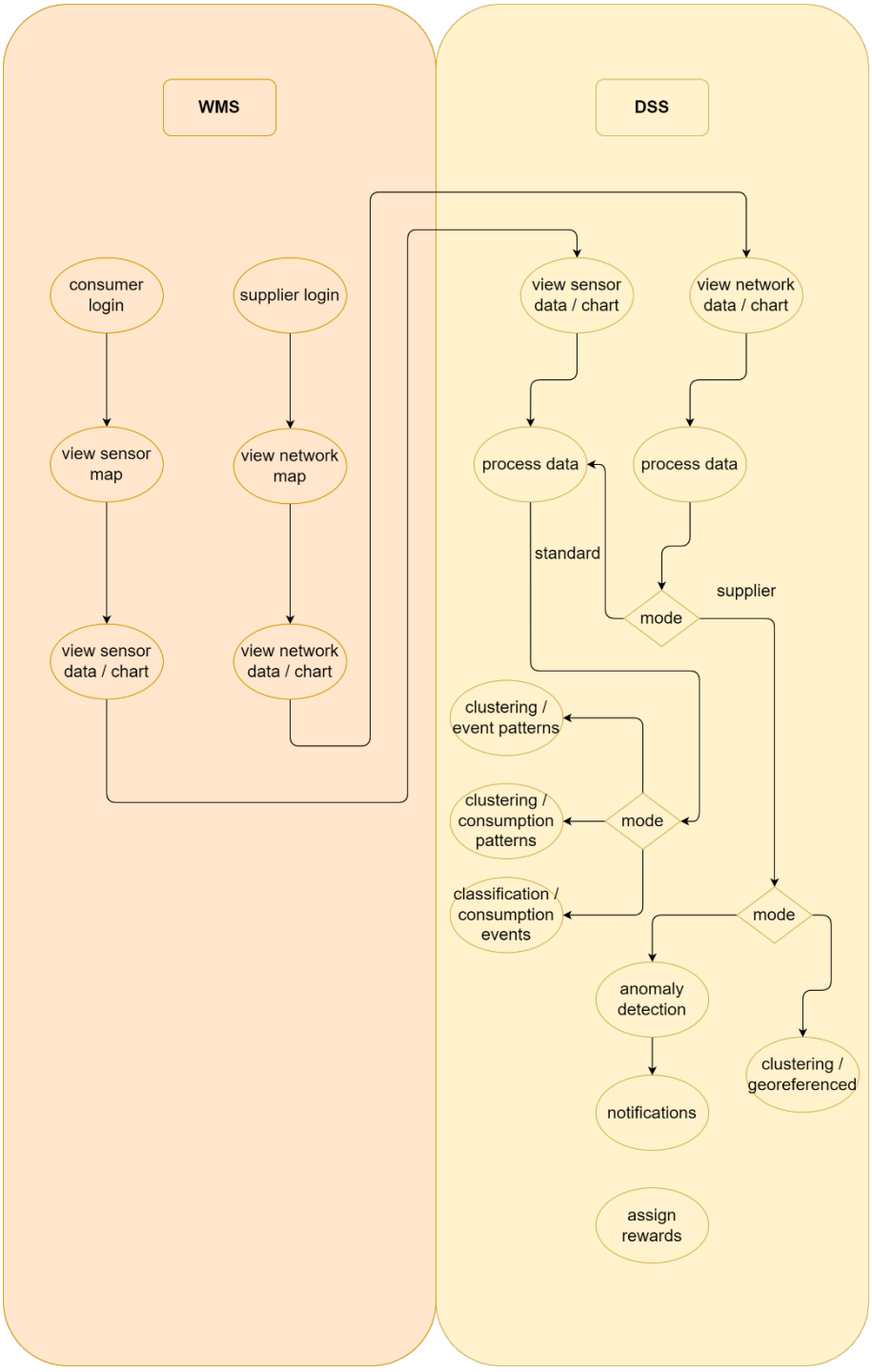


Figura 11 Diagrama de activități a sistemului integrat pentru monitorizare și suport decizional

Integrarea componentelor WMS și DSS este ilustrată în diagrama de activități din Figura 11. În contextul sistemului de monitorizare a consumului de apă (WMS), sunt disponibile metode de accesare și vizualizare a datelor preluate de la senzorii instalați în locuințe: accesare date în timp real sau stocate, vizualizare pe hartă a senzorilor instalați, vizualizare sub formă de grafice. Sistemul

de suport decizional (DSS) este alcătuit dintr-un set de algoritmi de procesare a datelor pentru implementarea diferitelor scenarii propuse: gruparea și clasificarea datelor, detecția anomaliilor, și notificări la nivel de sistem.

3.1.1 Metoda de colectare a datelor și detecție a anomaliilor

Anomaliile care pot apărea în infrastructurile de apă variază de la probleme de calitate a apei la probleme de infrastructură, cum ar fi funcționarea defectuoasă a contoarelor, senzorilor sau scurgerile de apă, până la perturbarea tiparelor de consum ale utilizatorilor. Aceste anomalii pot fi detectate prin analiza datelor colectate de la contoare inteligente, de la senzori care măsoară diferiți parametri legați de calitatea apei (cum ar fi valoarea pH, conductivitate, turbiditate etc.) și de la alte tipuri de senzori instalați de-a lungul rețelei de distribuție a apei.

Sistemul de colectare, analiză și raportare a anomaliilor este structurat pe trei straturi: Monitoring, Edge și Cloud.

La baza arhitecturii se află nivelul de Monitorizare, care include punctele de colectare a datelor (contoare inteligente) instalate la locația consumatorului și măsoară consumul instant de apă. Contoarele inteligente au fost implementate cu debitmetre de apă YF-S201 (optimusdigital.ro, 2022) atașate la plăcile NodeMCU (Nodemcu.readthedocs.io, 2022) echipate cu microcipuri wifi ESP8266 (espressif.com, 2022). Mai multe astfel de puncte de colectare a datelor pot fi instalate într-o singură locație. Datele neprocesate sunt trimise periodic, în pachete, către dispozitivele Edge.

Nivelul Edge cuprinde dispozitive cu capacitate redusă de stocare și procesare, care aplică algoritmi de preprocesare asupra datelor primite de la contoarele inteligente. Datele preprocesate sunt apoi trimise către serviciul de colectare a datelor din Cloud pentru stocare pe termen lung și procesare offline. La Edge, datele sunt stocate într-o fereastră de timp configurată în funcție de capacitatea de stocare a dispozitivului utilizat. Pentru experimentele noastre inițiale, am folosit un dispozitiv Nvidia Jetson Nano (developer.nvidia.com, 2022). Acest dispozitiv are un CPU de tip Quad-core ARM A57 cu o frecvență de 1.43 GHz, cu o memorie de 4GB LPDDR4, spațiu de stocare folosind un microSD de 64GB și un GPU de tip Maxwell 128-core.

La nivel de Cloud, există un serviciu de colectare a datelor și un serviciu avansat de procesare a datelor (offline). Serviciul de colectare a datelor primește date preprocesate de la dispozitivele Edge și le stochează într-o bază de date standard OGC SensorThings (fraunhoferiosb.github.io, 2022). Serviciul de prelucrare a datelor poate aplica proceduri complexe de preprocesare și analiză, folosind algoritmi de învățare automată asupra datelor stocate de la serviciul de colectare a datelor pentru consumatori sau pe seturi de date din alte surse. Serviciul de procesare a datelor are atașată o interfață web care permite utilizatorului să încarce seturi de date, să configureze și să ruleze mai mulți algoritmi de procesare a datelor și să vizualizeze rezultatele analizei datelor. Serviciul de analiză a datelor poate suporta sarcini periodice de analiză a datelor care emit notificări de anomalii către sisteme externe, cum ar fi alte dispozitive sau aplicații mobile.

Acces la date externe

Pentru a putea analiza datele colectate de terți în cadrul sistemului nostru, am proiectat o componentă de acces la date care este capabilă să citească date din servicii de date externe sau din baze de date. Datele pot fi trimise către sistemul nostru în mai multe moduri:

1. Încărcarea fișierelor de date - acceptăm mai multe extensii de fișiere: .csv, .xslx, .json, .xml, .npy, .h5, .mat, .arff;
2. Furnizarea unui șir de conexiune (connection string) - utilizatorul poate furniza un șir de conexiune și numele tabelului care conține datele, astfel sistemul nostru poate fi conectat direct la un sistem extern;
3. Prin solicitări HTTP - este disponibil un API care poate fi folosit pentru a încărca date în sistemul nostru;
4. Comunicare MQTT pentru date în timp real.

Această componentă comunică și cu serviciul nostru de colectare a datelor, care este responsabil de colectarea datelor de la diferiți senzori și dispozitive de Edge conectate.

Detecția anomaliilor

Pentru a găsi un algoritm de prognoză a consumului de apă cu performanță bună, am efectuat mai multe experimente pe setul de date numit Helix (Chatzigeorgakidis, 2018). Setul de date conține serii de timp privind consumul de apă pentru o perioadă de doi ani și pentru un total de 822 de gospodării. Fiecare element din setul de date constă dintr-un identificator anonim a unei case, latitudinea și longitudinea geografică a locației sale și o serie de timp medie de 24 x 7 = 168 de valori pentru fiecare oră din săptămână. Seria de timp medie este produsă prin medierea datelor orare de-a lungul întregii perioade.

Pentru fiecare serie de timp din setul de date am antrenat și testat mai mulți algoritmi:

1. Moving Average (Nakano, Takahashi, & Takahashi, 2021)
2. Auto-regression (Zhou & Li, 2019)
3. Exponential Smoothing (Raiyn & Toledo, 2014)
4. Auto-ARIMA (Tim, 2015)
5. XGBRegressor (Zhagparov et al., 2021)
6. LSTM (Yu et al., 2019)
7. Facebook's Prophet (Toni et al., 2021)

3.1.2 Aplicația mobilă WaterMonitoringSystem

Aplicația mobilă WaterMonitoringSystem (WMS) disponibilă pentru Android permite monitorizarea consumului de apă din perspectiva consumatorilor sau a furnizorilor, oferind o interfață interactivă pentru sistemul de monitorizare a consumului (Figura 12). Datele de consum sunt preluate de la server-ul aplicației, iar autentificarea se realizează prin Firebase pentru o securitate sporită și decuplarea serviciilor de date (datele colectate de la senzori sunt gestionate separat de cele referitoare la conturile utilizatorilor).

Aplicația mobilă implementează cazurile de utilizare identificate în etapa de proiectare a arhitecturii sistemului, precum: vizualizarea senzorilor pe hartă, vizualizarea graficelor, cont furnizori / consumatori, raportare probleme.

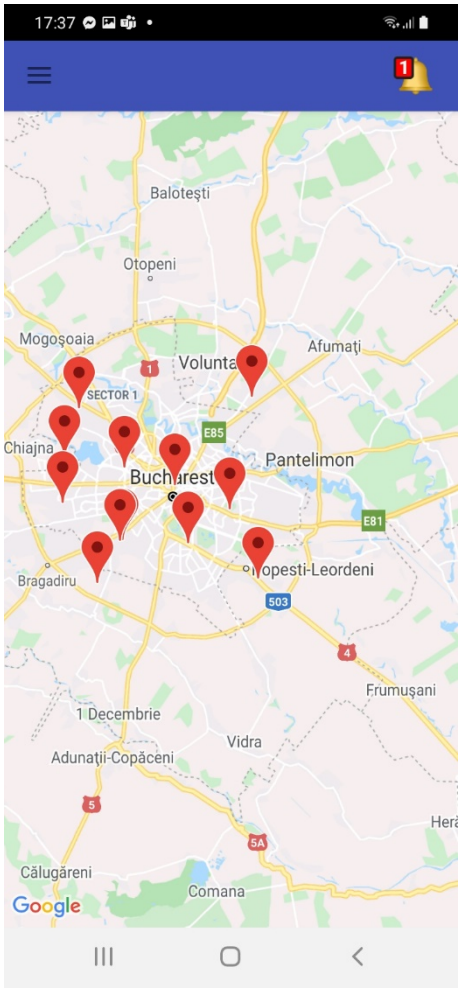


Figura 12 Aplicația WaterMonitoringSystem (mod furnizor). Vizualizarea pe hartă a modulelor instalate.

Implementarea aplicației este realizată în Android Studio, folosind limbajul Java. Pentru conectarea la serviciul Firebase, se configurează un proiect în Firebase Console și o aplicație mobilă (Khawas & Shah, 2018). Platforma generează un fișier de configurare care este adăugat în proiectul Android, preluat de SDK-ul Firebase pentru conectare la baza de date.

3.2 Integrarea componentelor de raportare urbană și Blockchain

În contextul aplicației Watergame, factorul uman este principalul motor pentru raportarea incidentelor în infrastructura de apă. Aplicația permite plasarea de indicatori pe hartă în mod interactiv, pe bază de geo-localizare și elemente de joc pentru stimularea participării.

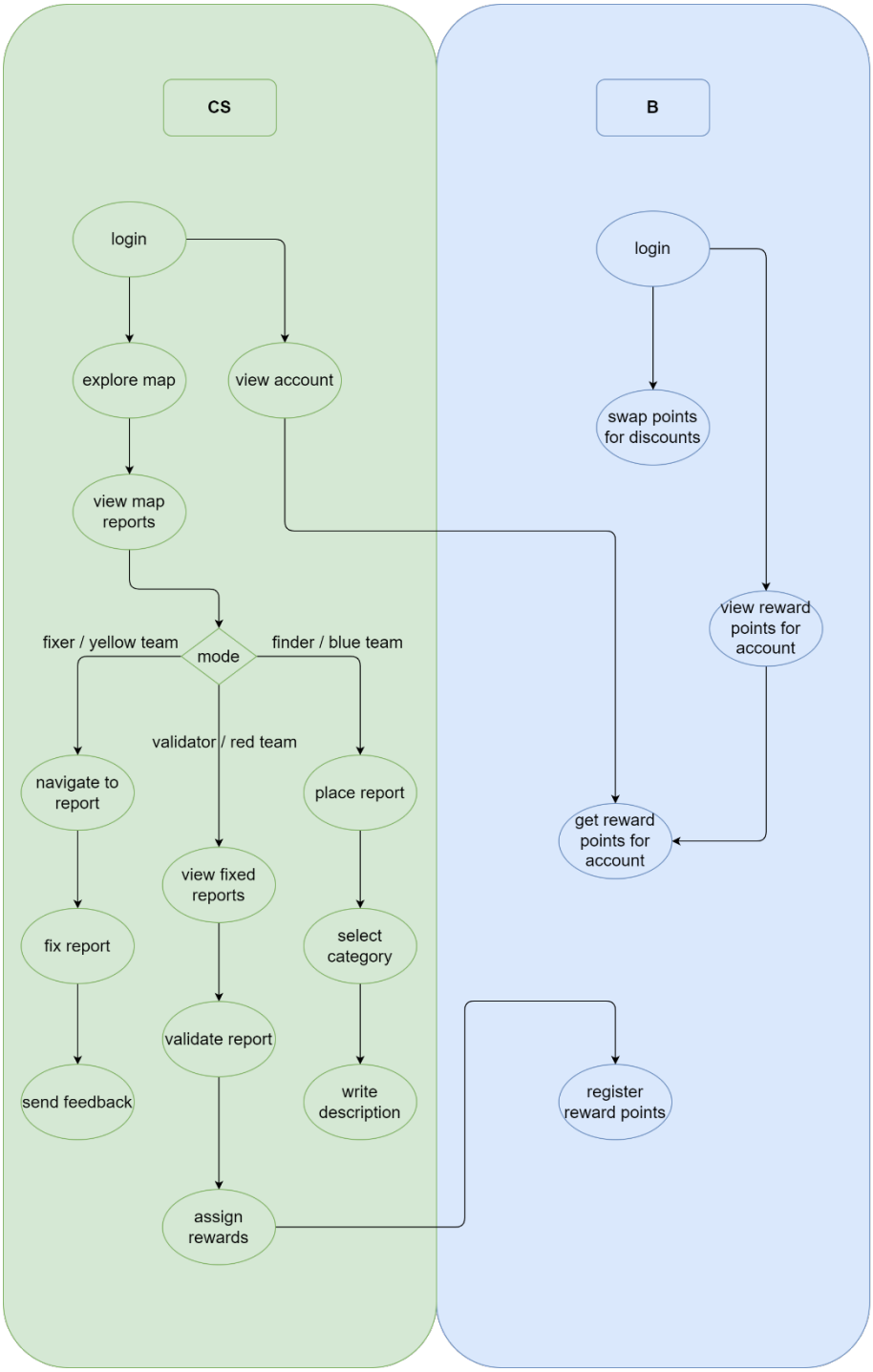


Figura 13 Diagrama de activități a sistemului integrat pentru raportare și recompensare

Integrarea componentelor CS și B este ilustrată în diagrama de activități din Figura 13. În contextul sistemului de raportare a incidentelor în rețeaua de distribuție a apei la nivel urban (CS), sunt disponibile metode interactive cu elemente de gamificare: explorare și vizualizare raportări pe hartă, plasare și editare a raportărilor pe hartă, validare raportări în mod interactiv, vizualizare nivel de reputație, magazin virtual cu recompense. Sistemul de suport Blockhain (B) constituie mecanismul de validare a recompenselor, prin intermediul monedei virtuale WGT (Watergame Token), urmărind

scenariile de valorificare a punctelor: vizualizarea tranzacțiilor și a balanței contului în rețeaua blockchain, preschimbarea WGT în oferte avantajoase sau discount-uri în furnizarea serviciilor în rețeaua de distribuție a apei, înregistrarea punctelor asociate WGT în contextul recompensării utilizatorilor.

3.2.1 Integrarea aplicației de raportare urbană

Arhitectura componentei de raportare urbană (Figura 14) are la bază o infrastructură de tip Cloud pentru suportul serviciilor de date. Infrastructura Cloud este găzduită pe AWS (Amazon Web Services) reprezentată printr-un server de aplicație care gestionează cererile HTTP precum și logica de acces la baza de date și la sistemul de stocare a datelor multimedia. Sistemul operează cu o bază de date MySQL ce asigură persistența datelor și o structură bine definită pentru operabilitate. Pentru scalabilitate, sistemul incorporează și o memorie temporară de tip REDIS. Datele multimedia sunt stocate într-un serviciu de stocare cloud (Amazon S3).

Interfața este realizată sub formă de aplicație web interactivă, cu extensie pentru dispozitive mobile. Aplicația web hibridă se ocupă de motorul de joc, interfața cu utilizatorul și interacțiunile cu serviciile externe. Principalele caracteristici gravitează în jurul componentei Map, care se bazează pe biblioteca Google Maps. Aplicația a fost proiectată folosind un framework hibrid care permite o flexibilitate sporită și suport cross-platform. Mai multe componente native sunt utilizate pentru a valorifica capabilitățile dispozitivelor mobile, de exemplu, geo-localizarea, notificările push, vibrațiile și camera. Implementarea funcționalităților presupune integrări cu servicii externe precum: Firebase și Google Maps. Arhitectura jocului este reprezentată de cele 3 scenarii specifice fiecărui tip de utilizator (Finder, Fixer, Validator) și are la bază harta interactivă configurată pentru a integra elementele de joc propuse.

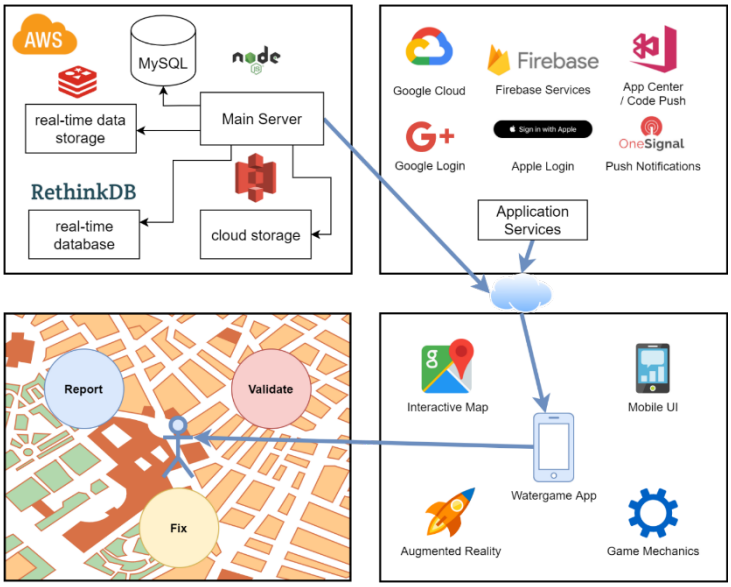


Figura 14 Arhitectura sistemului de raportare a incidentelor pe bază de crowdsensing

Back end-ul aplicației este construit pe Node.js, folosind framework-ul Nest.js[7] și conține logica de aplicație pe partea de server. Acesta are la bază o arhitectură pe 3 niveluri ce implică o modularitate nativă și mentenabilitate crescută prin separarea controlerelor (rutelor) de stratul de servicii (Business Logic) și de stratul de acces la date (Object Data Manager) (NestJS, 2022).

Nivelul de stocare a datelor este compus dintr-o bază de date relațională MySQL, și un depozit de date REDIS în memorie. Subsistemul REDIS[8] a fost adăugat pentru a îmbunătăți performanța și scalabilitatea sistemului atunci când se lucrează cu cantități mari de date și se gestionează mai multe solicitări. Sistemele de stocare în memorie au fost utilizate pe scară largă pentru aplicații cu randament ridicat pentru memorarea în cache a bazelor de date și mesageria în timp real, asigurând o bună scalabilitate și fiabilitate a datelor pentru majoritatea scenariilor de utilizare a sistemelor cloud (Chen, 2016).

3.2.2 Integrarea componentei Blockchain

Pentru realizarea integrării cu componenta Blockchain, acest modul a fost supus unor transformări pentru facilitarea procesului de interconectare cu celelalte sisteme.

Un prim pas a constat in modificarea contractelor inteligente specifice blockchain, acestea fiind împărțite după cum urmează:

- WaterGameToken – contract ce administrează token în sine cu toate funcționalitățile specifice. Pentru implementarea acestei funcționalități, în scopul optimizării și eficientizării codului existent s-a introdus biblioteca openzeppelin[9], bibliotecă gratuită specifică limbajului Solidity, ce conferă o securitate mai ridicată și mentenanță în timp real din partea comunității de contribuitori.
- WaterGameTokenSale – contract ce administrează zona de ICO (oferta inițială de vânzare a token-ului) și de achiziționare token. Această funcționalitate permite alocarea în cadrul contractului a unei cantități de token specificată la momentul deploymentului în rețea. În scopul oferirii de control asupra acestui proces, au fost implementate metode care permit: pornirea campaniei de vânzare, oprirea temporară a campaniei de vânzare și închiderea campaniei, prin returnarea cantității de monedă rămasă nevândută în contul administratorului soluției.
- WaterGameSpender – contract prin intermediul căruia se va permite cumpărarea unui produs prin criptomonedă. Această funcționalitate a fost dezvoltată pentru a putea fi apelată direct în soluția de Crowdsensing. Pentru îmbunătățirea zonei de administrare a fost implementată și funcționalitatea de restricționare a cheltuirii tokenilor, funcționalitate activabilă la cerere.
- WaterGameRewarder – contract prin intermediul căruia se va permite oferirea de recompense către un utilizator în momentul în care acesta efectuează o acțiune în sistemele integrate ca, de exemplu încărcarea unui eveniment în soluția de Crowdsensing. Pentru

[7] https://nestjs.com/

[8] https://redis.io/

[9] https://github.com/OpenZeppelin

îmbunătățirea zonei de administrare a fost implementată și funcționalitatea de restricționare a primirii recompenselor, funcționalitate activabilă la cerere.

Următoarea etapă pentru facilitarea integrării a fost reprezentată de generarea fișierelor JSON caracteristice contractelor inteligente și modificarea zonei de front-end a aplicației deja existente astfel încât metodele caracteristice să fie ușor integrabile în orice modul front-end al unuia dintre celelalte module ale sistemului.

În final, a rezultat o aplicație de sine stătătoare ce se conectează la rețeaua de test Ethereum (Goerli) și comunică direct cu infrastructura blockchain prin intermediul web3.js și Metamask, fără necesitatea unei zone de back-end. Datorită modularizării acestei aplicații, componentele acesteia, au fost ușor integrabile în aplicația de Crowdsensing.

În scopul integrării componentei și a utilizării ei la nivel public, contractele inteligente au fost înregistrate în rețeaua publica Goerli. Informațiile despre aceste contracte pot fi vizualizate pe site-ul oficial etherscan[10].

Tranzacțiile prin intermediul cărora au fost activate funcționalitățile descrise mai sus sunt următoarele:

- Activare rewarding: „*0xf74ecbfd97245f5cd6b6f4aefc3576435568fca7fc094250c9fbafeb5536914f*”
- Activare donații: „*0x5fe71e65934528866ed5c672aa807edfaf259dad506c7ae03d6c97439cea1089*”
- Activare utilizare token: „*0xcdafaa7fb9801e61f4efb51097681671071dd9fa1f3ef16620046a6d07e4f200*”
- Lichiditate contract sale: „*0x355b4861e4fae8630478752e95b6712fb64fc8039d209f422e8eb2027f263f09*”
- Cumparare token ICO: „*0x11c3b04b20420232bb4da693049e6cdeeb926598ecee8d70ddc562e338d188d0*”

În ceea ce privește zona de front-end a aplicației, aceasta a fost modularizată astfel încât fiecare funcționalitate să aibă o metodă independentă, ușor integrabilă în cadrul oricărei alte soluții front-end.

- *getReward: function()*
- *spendOnItem: function()*
- *donateToken: function()*
- *buyToken: function()*

4 Validarea sistemului și demonstrarea funcționalităților

În această etapă, a fost validată funcționarea sistemului integrat la nivel de laborator. Sunt prezentate instrumentele, metodele, și interacțiunile la nivel de sistem, precum și rezultatele experimentale obținute în contextul scenariilor definite în etapa de proiectare.

4.1 Verificarea funcționării sistemului integrat

În această activitate, au fost realizate integrările propuse, iar sistemul a fost validat în ansamblu, pe baza scenariilor experimentale.

[10] https://goerli.etherscan.io/

4.1.1 Monitorizare în rețeaua de senzori

Analiza datelor

Pentru a analiza datele primite de la senzori, am utilizat platforma de analiză de date și am antrenat mai multe modele ARIMA pentru predicția consumului de apă. Figura 15 prezintă un exemplu în care am antrenat un model ARIMA pe datele primite de la senzorul 89776. După antrenarea modelului, în partea de jos putem vedea metrica de validare RMSE (în cazul acesta e 2.231 ceea ce e o valoare foarte bună), precum și valorile reale (prezentate pe diagramă folosind culoarea albastră) și valorile prezise de model (marcate cu roșu). Așa cum se vede și în figură, am reușit să creăm un model foarte precis, linia roșie (valori prezise) se suprapune cu linia albastră (valori reale) aproape perfect.

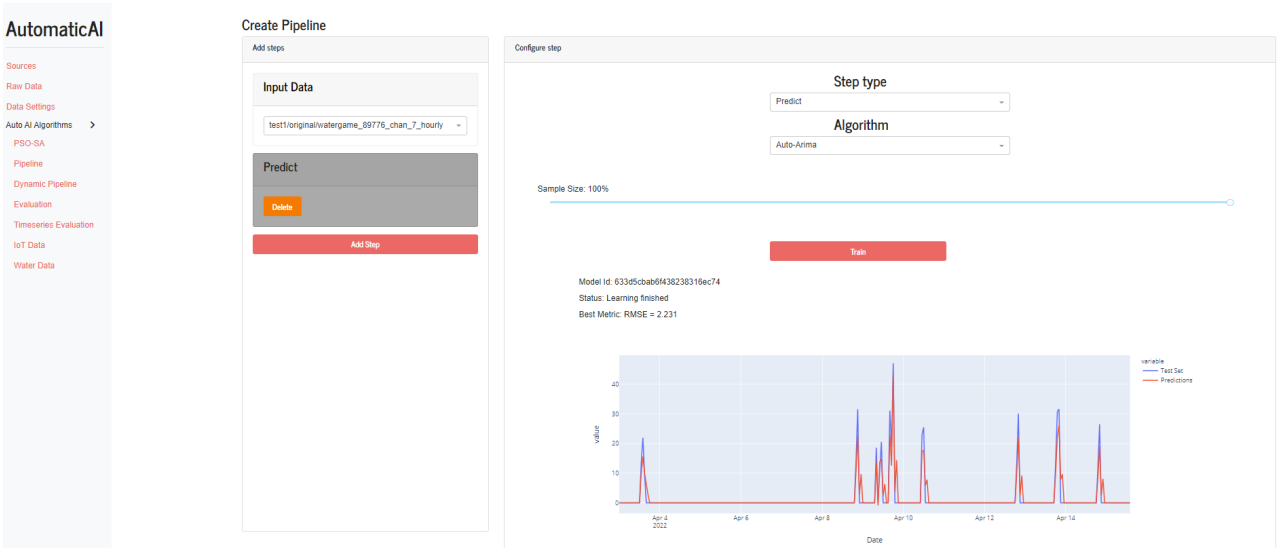


Figura 15 Antrenarea modelului ARIMA pe date citite de la senzori

Pentru fiecare senzor am antrenat modele separate de tip ARIMA, astfel am reușit să maximizăm precizia și să minimizăm impactul unui senzor stricat. Figura 16 prezintă câteva rezultate obținute folosind date reale culese de la senzori.

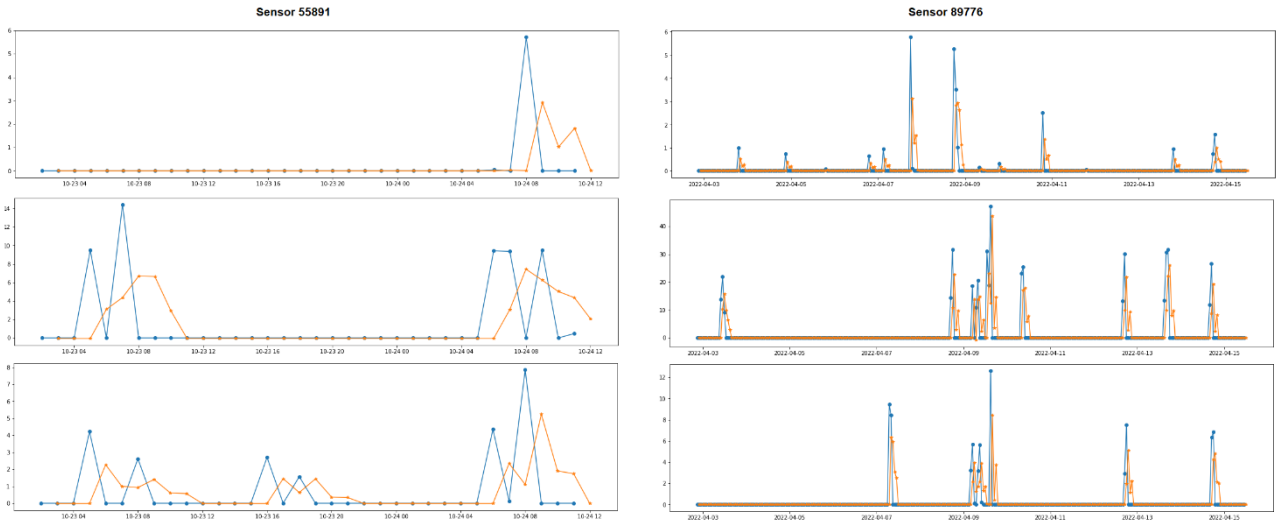


Figura 16 Rezultatele modelelor ARIMA antrenate pe date primite de la senzori

Așa cum se vede în Figura 16, în cazul senzorului 89776 avem un model mult mai precis, ceea ce se datorează faptului că în acest caz am avut mai multe date de antrenare.

Monitorizarea defecțiunilor

Pentru monitorizarea defecțiunilor, Prometheus oferă o interfață grafică pentru vizualizarea metricilor existente și a configurației actuale (Figura 17). Metricile sunt citite de la server la fiecare 5 secunde.

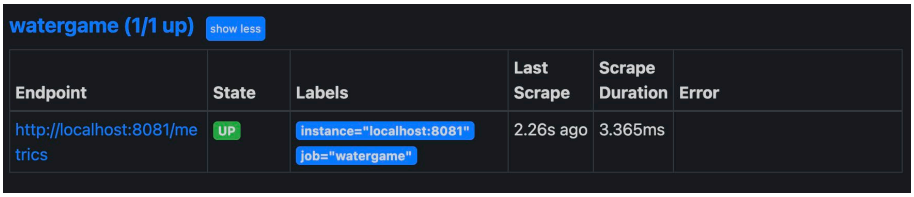


Figura 17 Interfața Prometheus

Metrica poate fi vizualizata sub formă de tabel, evidențiind-se seriile de timp împreună cu ultima valoare înregistrată, așa cum se poate observa în Figura 18.

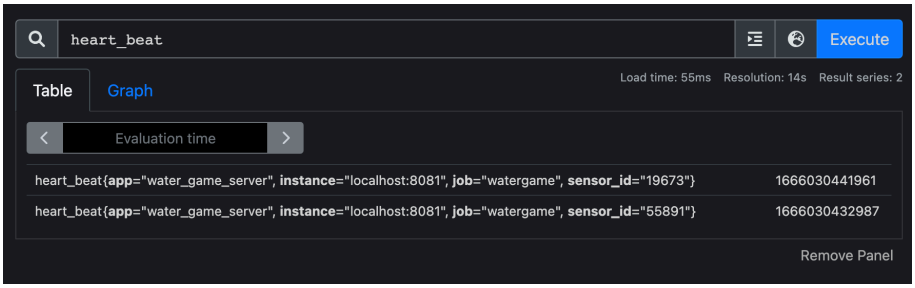


Figura 18 Vizualizare metrici în format tabelar

O altă vizualizare, prezentată în Figura 19, este aceea sub formă de grafic, putând fi observată evoluția în timp a valorii seriilor de timp. Este evidențiată corelarea dintre momentul înregistrării (axa X) și momentul primirii mesajului de către server (axa Y):

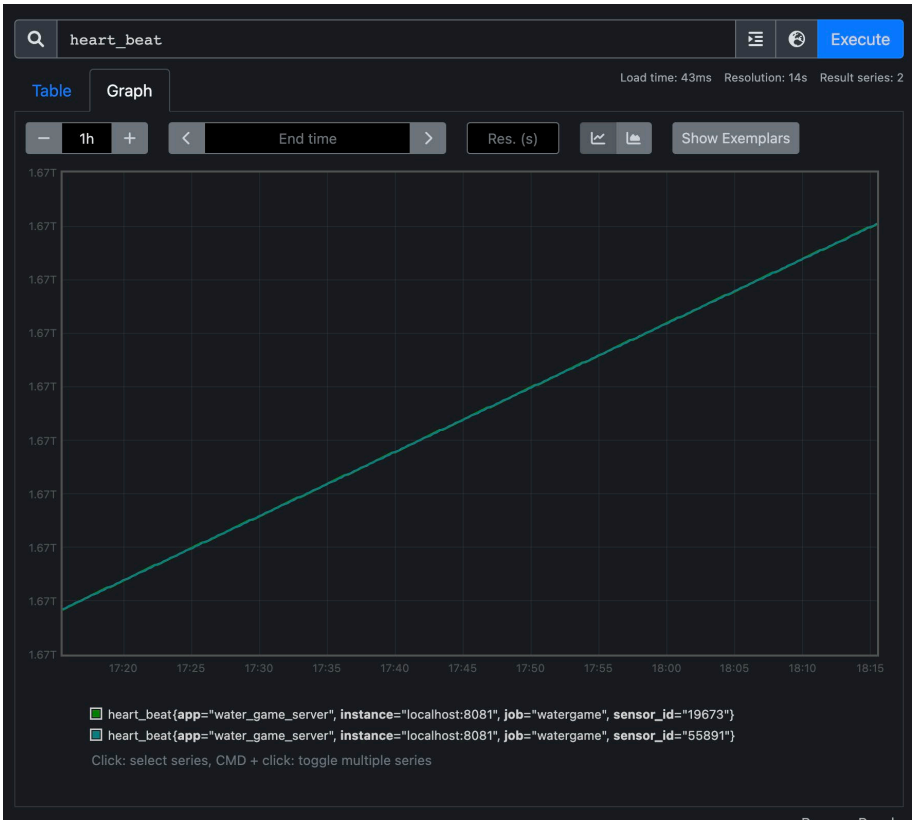


Figura 19 Vizualizare metrici în format grafic

Metrica ***heart_beat*** a fost agregata sub numele de ***hb:time_since_last_seconds*** care reprezintă timpul în secunde de la ultimul mesaj heartbeat. Agregarea are loc prin evaluarea la fiecare minut a expresiei: (time() * 1000 - last_over_time(heart_beat[1y])) / 1000

În graficul metricii agregate, prezentat în Figura 20, observăm menținerea unui comportament constant in condiții de funcționare normala a sistemului (senzorii trimit date).

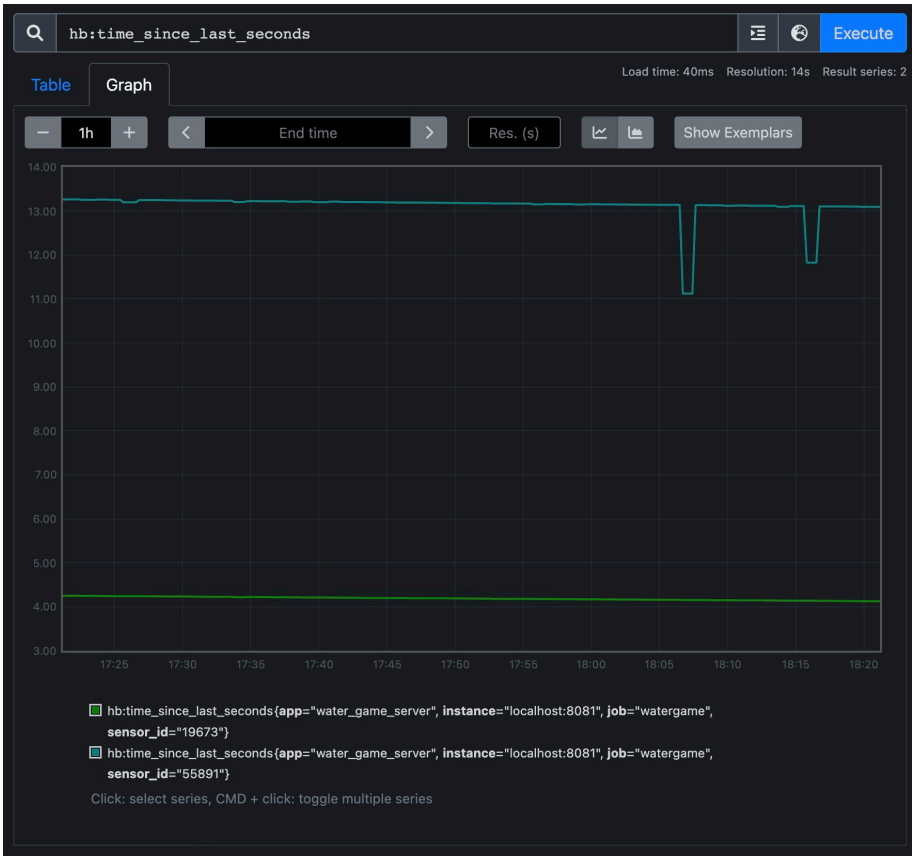


Figura 20 Vizualizare metrici agregate

Se observă o monotonie descrescătoare datorată de diferențele dintre ceasurile senzorilor care publică mesaje regulat si ceasul Prometheus care calculează agregatul periodic.

Alerta este configurată pentru a urmări valoarea metricii agregate (Figura 21).

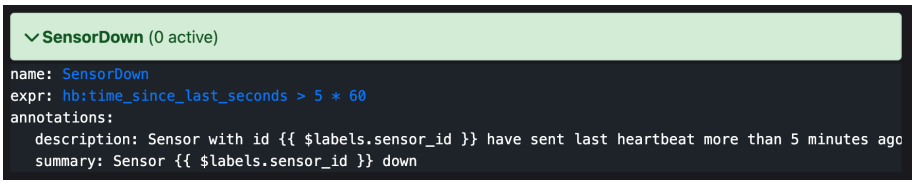


Figura 21 Configurarea alertei

Declanșarea unei alerte

In situația in care un senzor se oprește din a trimite mesaje de heartbeat, timestamp-ul ultimului mesaj va rămâne constant. Acest comportament se regăsește in captura de ecran din Figura 22.

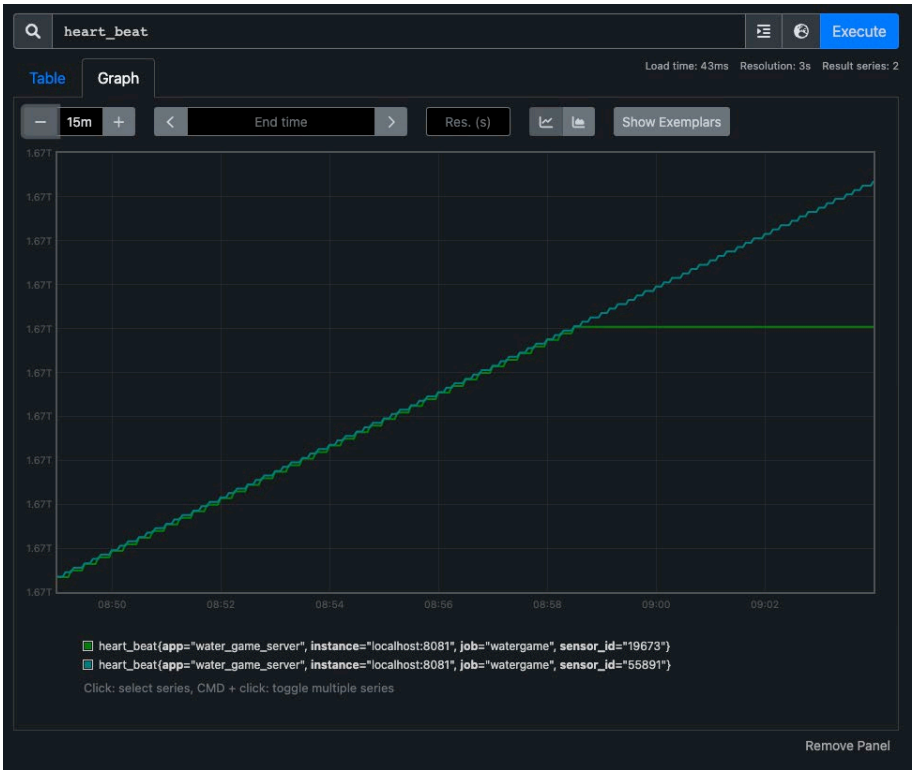


Figura 22 Declanșarea defecțiunii

Se observă că pentru senzorul cu id-ul 19673 ultimul mesaj a fost primit la 09:58:23 GMT+0.

În Figura 23, metrica agregată care reprezintă timpul de la ultimul mesaj va crește.

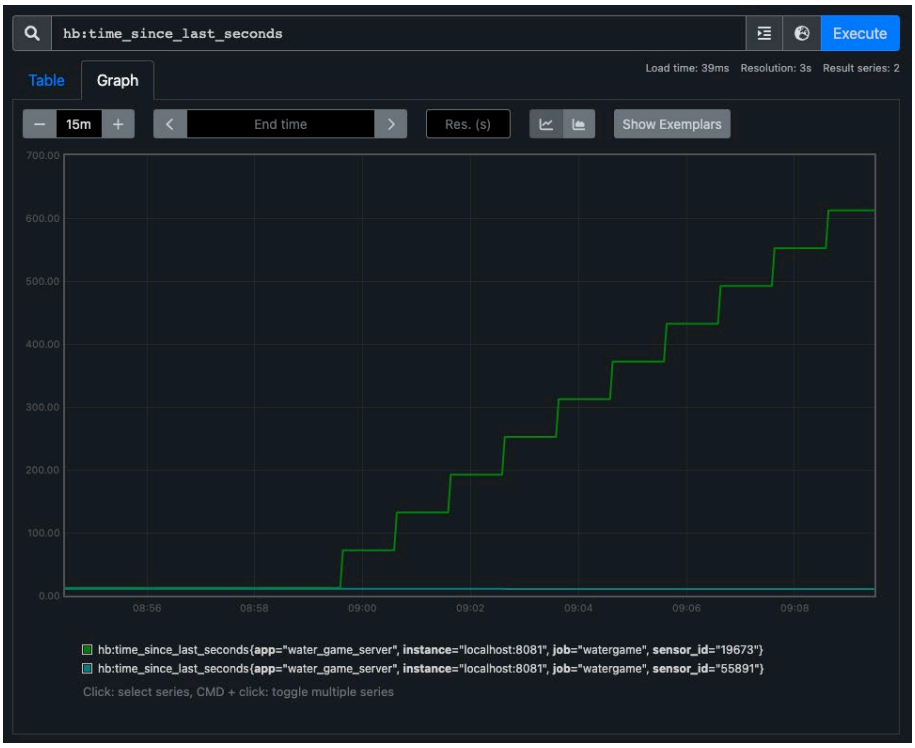


Figura 23 Detecția defecțiunilor prin metrica agregată

La 5 minute de la ultimul mesaj se declanșează automat alerta (Figura 24).

Figura 24 Declanșarea alertei

Alerta este trimisă către serverul aplicației care generează o notificare către furnizor, ce poate fi citită din aplicația mobilă (Figura 25).

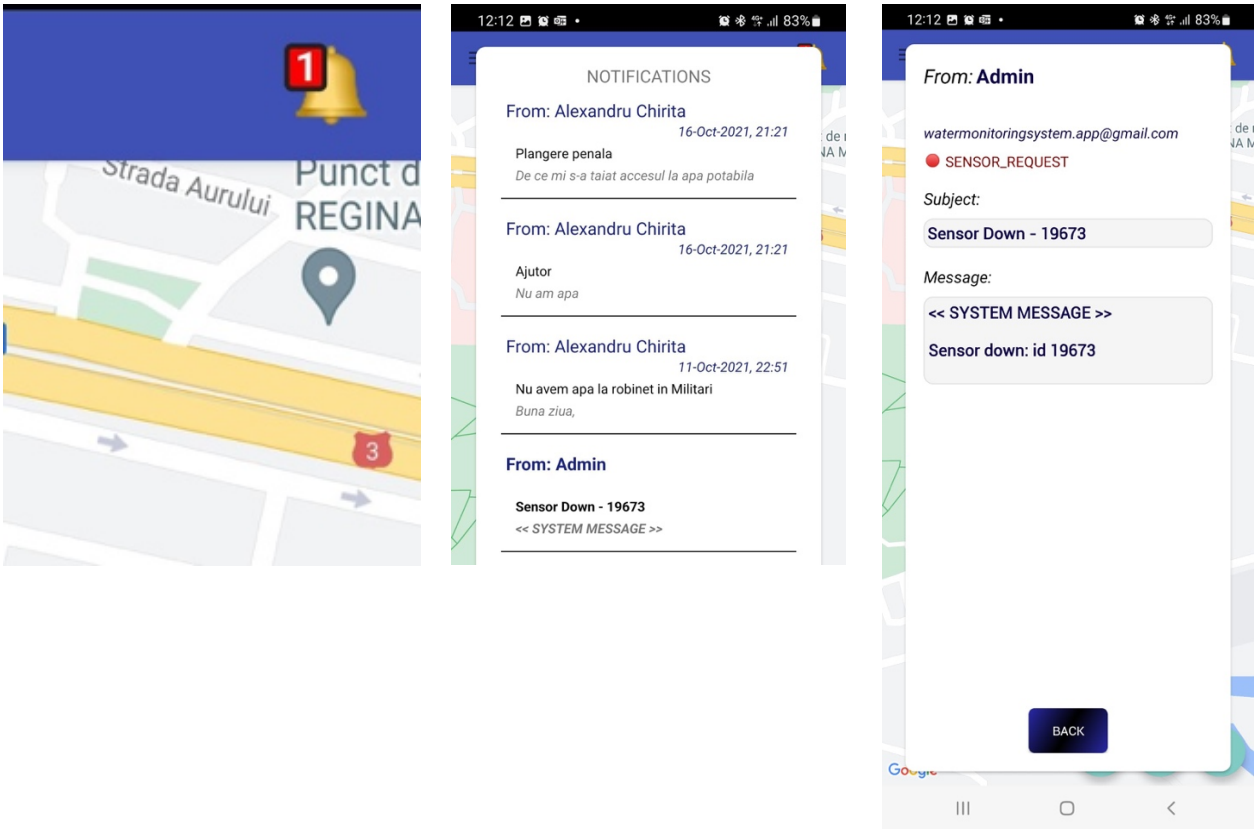


Figura 25 Notificare în aplicația WaterMonitoringSystem

Atunci când senzorul revine cu mesaje la server, se actualizează automat metrica agregată și status-ul alertei, așa cum este evidențiat în Figura 26 și Figura 27.

Figura 26 Actualizare metrică agregată

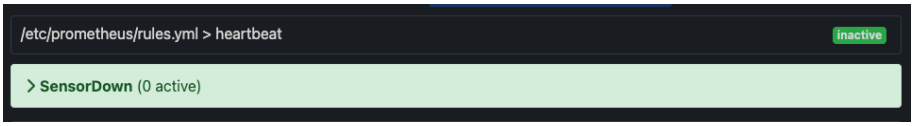


Figura 27 Resetarea alertei

Pentru a limita numărul de notificări, rezolvarea unei alerte nu generează o notificare, deoarece informația legata de funcționalitatea senzorului poate fi extrasa din interfața grafica a Prometheus.

4.1.2 Vizualizare și suport decizional în rețeaua de senzori

Metodele de procesare a datelor în contextul scenariilor sunt implementate ca script-uri Python și integrate în aplicația de procesare a datelor. Rularea scenariilor de grupare a modelelor de consum este evidențiată prin diagrama de activități din Figura 28:

- Gruparea modelelor de consum se realizează prin scripturile extract_combined.py (extrage datele din fișiere CSV asociate fiecărui consumator într-o reprezentare matriceală), și show_extracted.py (rulează algoritmul de grupare și efectuează reprezentarea vizuală a modelelor de consum ca serii de timp)
- Gruparea evenimentelor de consum are la bază reprezentarea combinată a datelor, obținută prin rularea script-ului extract_combined.py. Prin script-ul extract_events.py, se parcurg seriile de timp asociate fiecărui consumator și sunt extrase evenimentele de consum reprezentate prin durată și volum. Script-ul process_events.py încarcă evenimentele extrase, rulează algoritmul de grupare în funcție de durată și volum, și efectuează reprezentarea vizuală a rezultatelor grupării.

- Clasificarea evenimentelor de consum presupune două etape: antrenare și testare. În etapa de antrenare sunt rulate script-urile corespunzătoare algoritmilor de învățare profundă / deep learning (train_deep.py) și învățare automată / machine learning (train_dtree.py). Se antrenează astfel modelele și se evaluează acuratețea pe setul de date de validare. Pentru evaluarea modelelor în clasificarea efectivă a evenimentelor de consum, se utilizează script-urile eval_results_overview.py (evaluarea acurateței modelelor selectate) și eval_conf_matrix.py (realizarea matricei de confuzie).

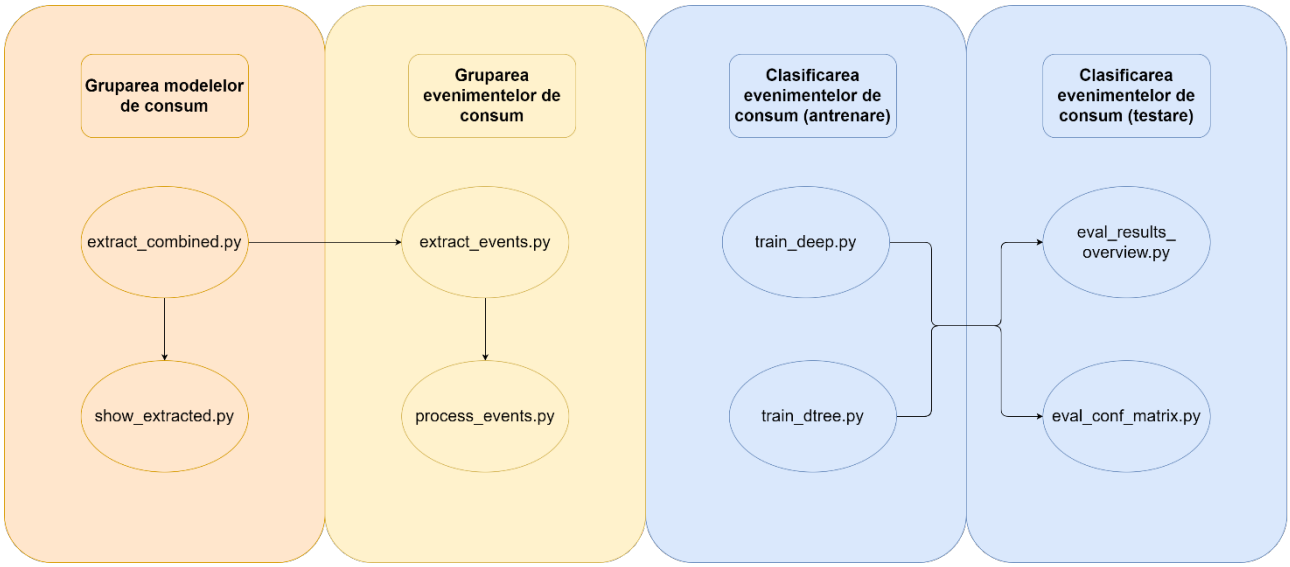


Figura 28 Diagrama de activități pentru rularea script-urilor de procesare

Script-urile de procesare și datele extrase din baza de date a sistemului de colectare sunt accesibile pe GitHub: https://github.com/alexp25/watergame-other

4.1.3 Raportarea evenimentelor cu suport blockchain

Aplicația Watergame este disponibilă în format web, adaptată pentru dispozitive mobile. Aceasta prezintă funcționalitățile principale din soluția de raportare a evenimentelor în infrastructura de apă în mediul urban, conform scenariilor identificate în etapa de proiectare.

Aplicația dispune de un mecanism de autentificare pe bază de email și parolă, prezentat în Figura 29. Datele de autentificare sunt înregistrate în format criptat în baza de date MySQL, folosind algoritmul BCrypt.

Figura 29 Autentificarea în aplicația Watergame

După autentificarea în aplicație, utilizatorul este direcționat către pagina principală, de unde poate accesa meniul principal, așa cum este prezentat în Figura 30. Meniul prezintă cele 3 secțiuni principale ale aplicației: profilul utilizatorului, zona comercială și harta interactivă.

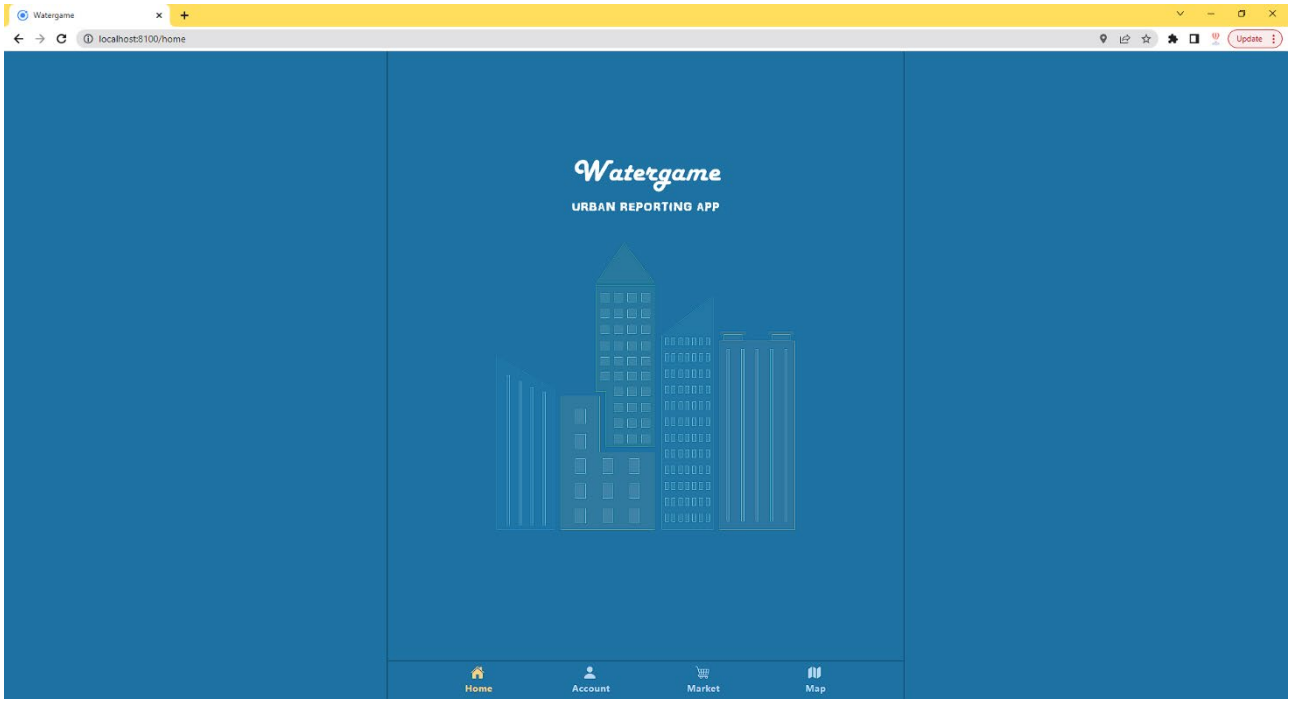


Figura 30 Pagina principală a aplicației Watergame

Profilul utilizatorului este disponibil în pagina de profil, așa cum este prezentat în Figura 31, și constă într-o reprezentare vizuală a progresului acestuia în cadrul jocului serios, conform rolurilor definite în etapa de proiectare: Finder (raportarea evenimentelor), Fixer (soluționarea raportărilor), Validator (validarea raportărilor). Pe baza acestui profil se calculează reputația utilizatorilor ce este definită pentru a calcula ponderea acțiunilor sale în procesul de recompensare interactivă.

Pagina de profil mai permite realizarea operațiunilor cu privire la contul utilizatorului în aplicație, precum modificarea numelui, a parolei, a pozei de profil, delogarea și ștergerea contului.

Figura 31 Profilul utilizatorului în aplicația Watergame

Zona comercială din aplicație este reprezentată prin catalogul de produse ce pot fi tranzacționate folosind moneda virtuală WGT (Watergame Token). Modelul de valorificare a recompenselor este construit pe baza sistemului Blockchain, ce constituie suportul pentru realizarea operațiunilor de validare și preschimbare a monedelor virtuale în beneficii reale, aplicabile în contextul furnizării de servicii în rețeaua de distribuție a apei. În Figura 32 este evidențiată modalitatea de conectare la Blockchain prin extensia MetaMask, ce permite realizarea de operațiuni cu moneda virtuală WGT.

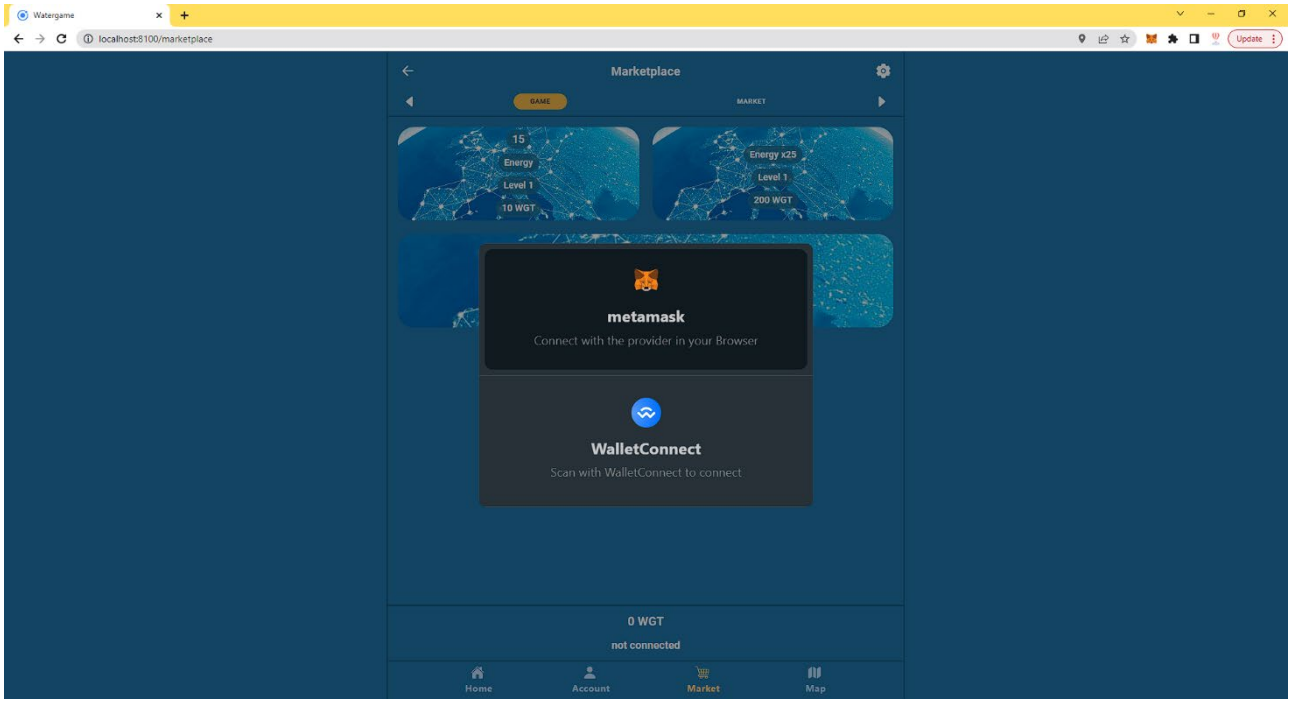


Figura 32 Magazinul virtual cu extensia de blockchain

În Figura 33, este prezentat modul de realizare a conexiunii la contul utilizatorului prin MetaMask. În acest context, se autorizează efectuarea tranzacțiilor prin aplicația Watergame.

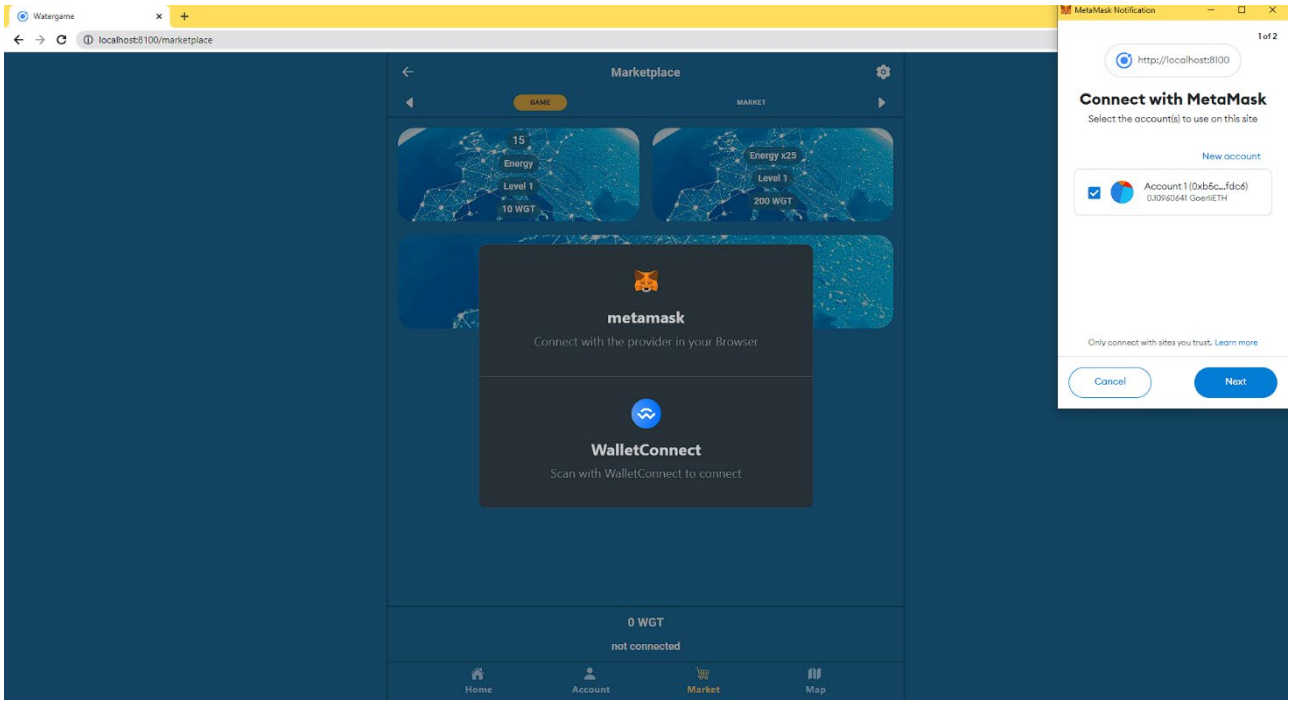


Figura 33 Autentificarea pe rețeaua blockchain folosind extensia MetaMask

În urma autentificării în rețeaua Blockchain, sunt preluate informațiile despre contul utilizatorului (identificatorul contului, balanța) și afișate în aplicație. Catalogul de produse devine disponibil, așa

cum se observă în Figura 34, și listează produsele disponibile pentru a fi cumpărate folosind moneda virtuală WGT. În exemplu, sunt listate două potențiale reduceri oferite de furnizorul de servicii.

Moneda virtuală poate fi obținută în urma activităților desfășurate în cadrul aplicației, și este asimilată recompenselor care pot fi acordate conform scenariilor propuse pentru rolurile Finder, Fixer și Validator.

Achiziționarea unui produs din catalog presupune transferul unei cantități de monedă virtuală din contul utilizatorului în contul furnizorului, în urma căruia produsul va fi înregistrat pe contul utilizatorului, în aplicația Watergame.

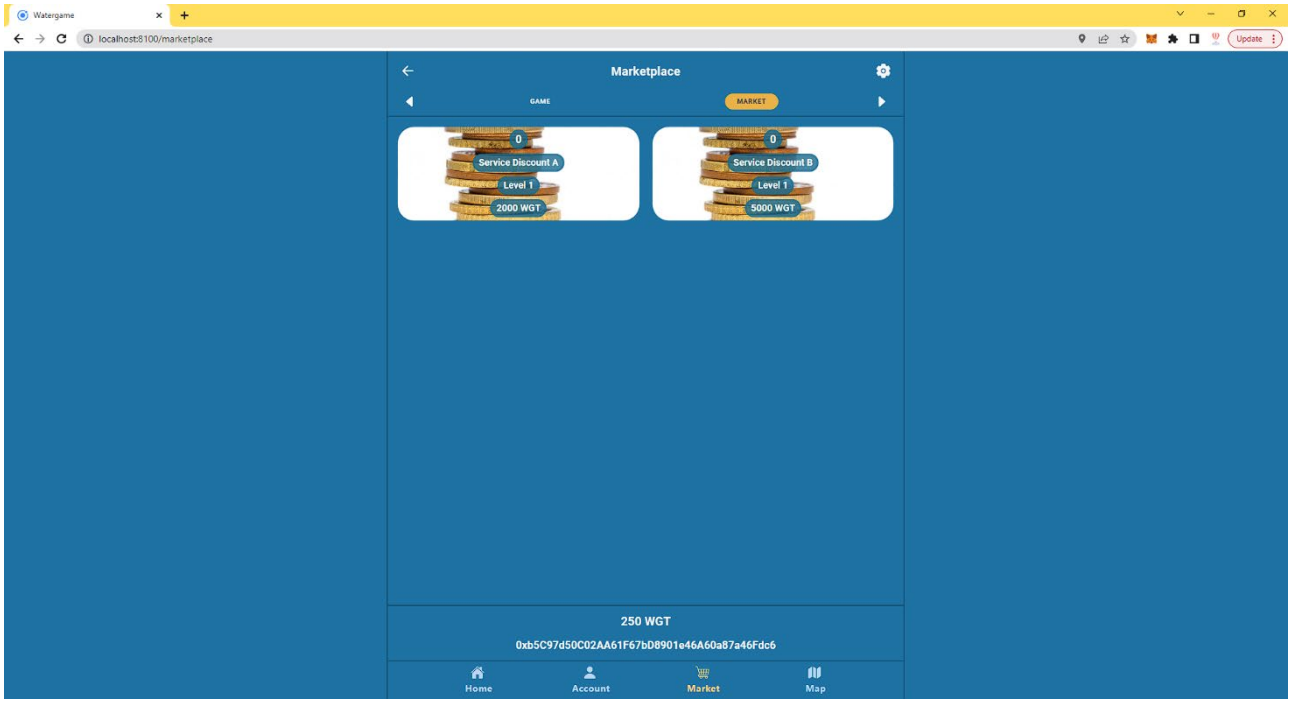


Figura 34 Vizualizarea produselor disponibile prin intermediul monedei virtuale

Pagina de raportare interactivă are la bază o hartă Google Maps, în care se pot vizualiza raportările de incidente din vecinătatea utilizatorului, reprezentate sub formă de puncte (markere), așa cum se pot observa în Figura 35.

Localizarea utilizatorului pe hartă este preluată de la serviciul de localizare disponibil în browser (e.g., localizare prin GPS), și este reprezentată prin marker-ul albastru, pentru utilizatorul de tip Finder (galben sau roșu pentru utilizatorii de tip Fixer sau Validator).

Utilizatorii pot adăuga raportări noi, sau pot vizualiza raportările încărcate de ceilalți utilizatori în mod interactiv. Localizarea acestora se poate ajusta prin selectarea și deplasarea punctelor pe hartă.

Figura 35 Vizualizarea raportărilor pe harta interactivă

În cadrul unei raportări, se poate selecta categoria din care face parte, așa cum este prezentat în Figura 35, pentru o încadrare și reprezentare vizuală corespunzătoare contextului general. Apoi, se poate introduce o descriere și încărca o poză, așa cum se poate observa în Figura 37.

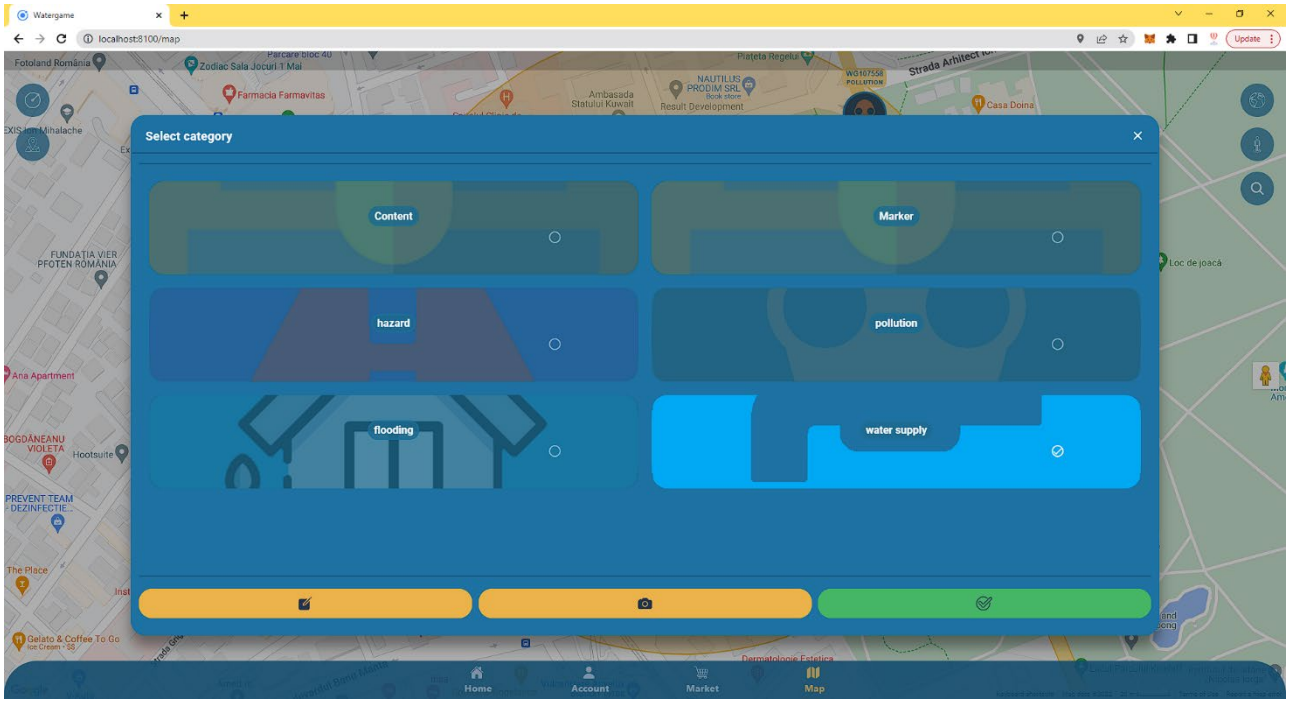


Figura 36 Adăugarea raportărilor pe hartă. Selectarea categoriei.

Fiecare raportare este înregistrată în baza de date cu un identificator unic. Informațiile despre raportări pot fi vizualizate public, iar aplicația permite validarea, soluționarea, actualizarea, sau ștergerea unei raportări.

Atunci când utilizatorii interacționează (validează sau soluționează raportările), informațiile aferente sunt înregistrate în baza de date. Un utilizator poate înregistra validarea unei raportări o singură dată, în timp ce pot exista mai multe validări din partea utilizatorilor diferiți.

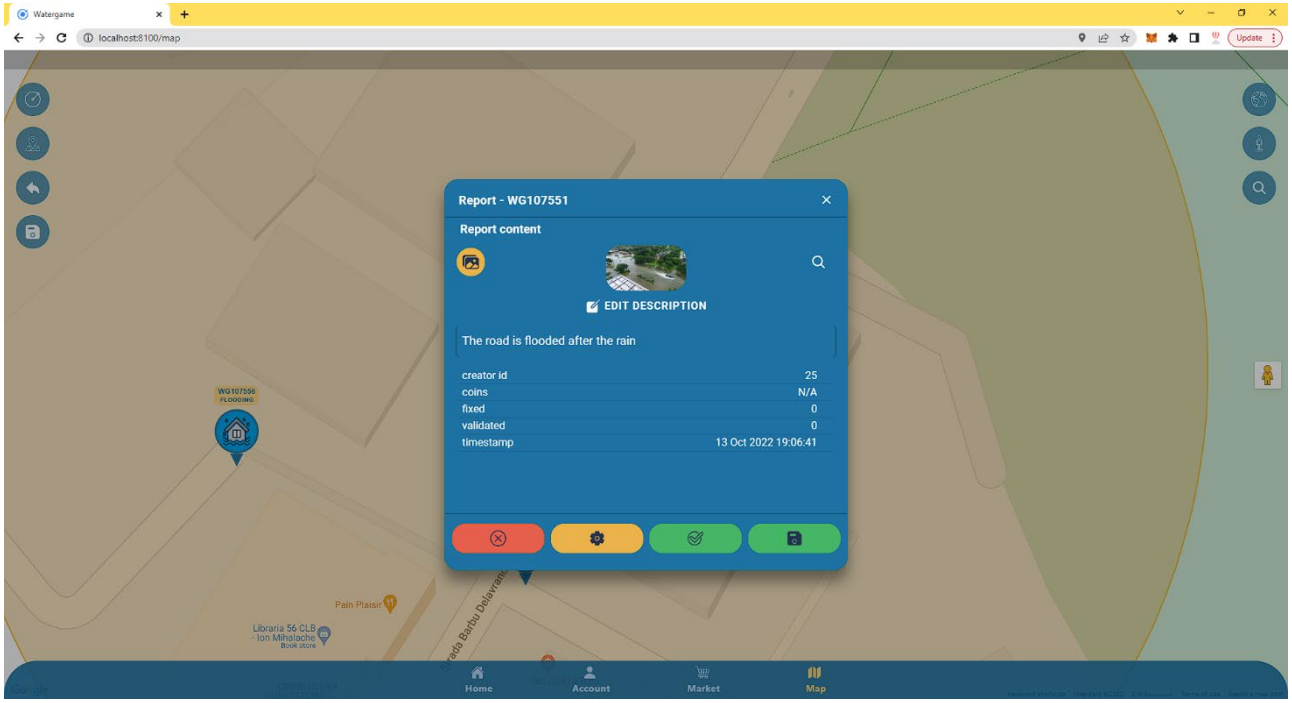


Figura 37 Adăugarea raportărilor pe hartă. Editarea conținutului.

4.2 Evaluarea rezultatelor experimentale obținute din funcționarea sistemului

În această activitate, au fost evaluate rezultatele experimentale obținute în urma evaluării sistemului în ansamblu, pe baza scenariilor experimentale.

4.2.1 Monitorizarea consumului de apă

Scenariul 1

Validarea scenariului UTCN: Monitorizarea consumului de apă în puncte multiple în cadrul unei locuințe unifamiliale în care se colectează date neprocesate direct de la punctele de monitorizare.

În cadrul locuinței s-au determinat punctele de măsurare ale consumului de apă rece la nivelul băilor, a bucătăriei și a centralei termice producătoare de apă caldă menajeră. S-au determinat un total de 8 puncte de măsurare pentru consumatorii de apă rece: 3 puncte la chiuvete, 2 puncte la cadă/duș, 2 puncte la rezervoarele de WC și un punct la admisia de apă rece a centralei termice (Figura 38).

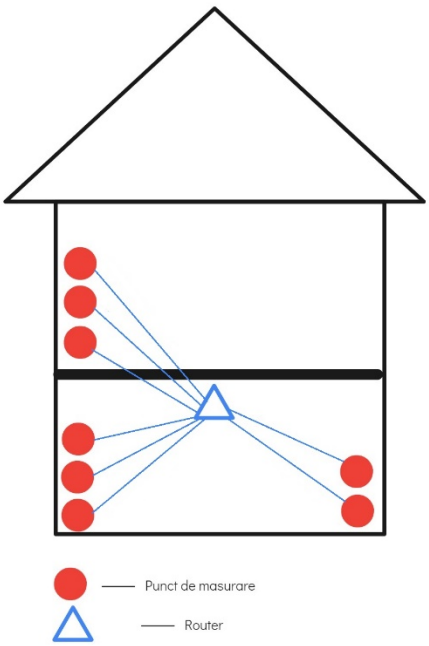


Figura 38 Colectare directă a datelor măsurate

Pentru fiecare dintre punctele de măsurare s-a montat câte un debitmetru pentru colectarea datelor de consum, debitmetru conectat la câte un nod NodeMCU. Alimentarea nodului s-a realizat prin intermediul adaptoarelor AC/DC conectate în apropierea punctelor de consum. La nivelul întregii locuințe fiecare nod a transmis datele neprocesate de debit direct la serverul de colectare a datelor aflat în Cloud (la nivelul infrastructurii UTCN). În Tabel 1 sunt prezentate datele măsurate și transmise neprelucrate și anume valoarea frecvenței furnizate de către debitmetrul YF-S201.

Tabel 1 Măsurători transmise neprelucrate

Nr. senzor	Data	Ora	Frecvența
1	30/06/2022	12:10	13.5
2	30/06/2022	12:37	15
3	30/06/2022	14:07	14.5
4	30/06/2022	15:48	15.5
5	30/06/2022	16:04	16
6	30/06/2022	17:04	17.5
7	30/06/2022	10:09	18
8	30/06/2022	18:23	17.5

Colectarea datelor s-a realizat în mod cursiv, fără întreruperi la punctele de colectare care nu erau expuse eventualelor intervenții din exterior (punctele cum sunt chiuvetele, unde prezența unui

mobilier de chiuvetă ajută la protejarea debitmetrului și a nodului) și cu întreruperi la nivelul punctelor de colectare cum este cazul vaselor de wc sau chiar al dușurilor, unde debitmetrul și nodul trebuie protejate împotriva intervențiilor din exterior sau a umidității. Pentru acest lucru, nodul și debitmetrul trebuie integrate într-un dispozitiv cu o carcasă rezistentă la umezeală și la acțiuni mecanice din exterior. În cazul testat, acțiunile mecanice externe sunt realizate în special de locatarii minori din cadrul locuinței unifamiliale.

Validarea relevanței datelor de consum pentru punctele în care s-au realizat măsurătorile, în mod cumulativ, se efectuează prin comparație cu datele de consum furnizate de către apometrul montat de către compania de apă la nivelul conductei principale de alimentare cu apă a locuinței unifamiliale. Datele de consum obținute pe scurte perioade de timp - 1, 2, 3 zile - sunt valide în raport cu datele de consum furnizate de apometrul general.

Scenariul 2

Validarea scenariului UTCN: Monitorizarea consumului de apă în puncte multiple în cadrul unei locuințe unifamiliale în care se colectează date neprocesate direct de la punctele de monitorizare, date care ulterior sunt agregate și preprocesate la nivelul unui dispozitiv local.

În cadrul aceleiași locuințe s-a adăugat un punct de agregare a datelor colectate de noduri (Figura 39). La nivelul unui dispozitiv Raspberry Pi datele neprocesate de consum, provenite de la toate cele 8 puncte de măsurare au fost colectate prin intermediul aplicației de tip server MQTT (Message Queuing Telemetry Transport) denumită Mosquitto. Mosquitto este o aplicație open-source de tip server MQTT care acționează ca un broker de mesaje în cadrul unei rețele wireless locale nesigure, unde anumite mesaje se mai pot pierde pe parcursul transmisiei.

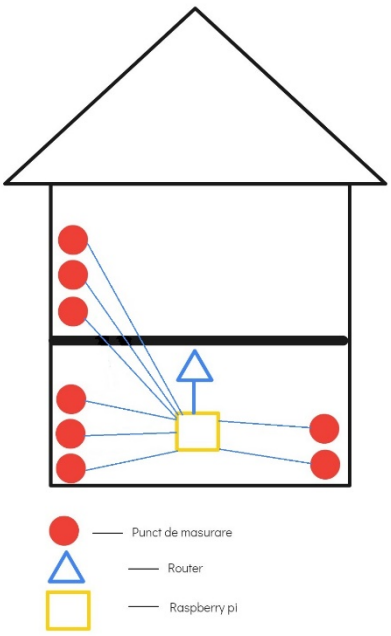


Figura 39 Dispozitiv local de prelucrare a datelor

Datele de consum calculate la nivelul dispozitivului Raspberry Pi sunt transmise ulterior în Cloud (la nivelul infrastructurii UTCN). Un exemplu este prezentat în Tabel 2.

Tabel 2 Date prelucrate după măsurare

Nr. senzor	Data	Ora	Litri/oră
1	15/07/2022	8:29	120
2	15/07/2022	8:32	102
3	15/07/2022	8:40	142
4	15/07/2022	8:36	126
5	15/07/2022	8:44	104
6	15/07/2022	8:49	108
7	15/07/2022	9:15	96
8	15/07/2022	9:33	84

Din nou, datele de consum obținute pentru teste efectuate pe perioade scurte de timp - 1, 2, 3 zile - sunt valide în raport cu datele de consum furnizate de către apometrul companiei furnizoare de apă.

4.2.2 Monitorizare și suport decizional

Scenariul A UPB: Pentru a monitoriza consumul de apa în vederea obținerii unui input consistent pentru suportul decizional, din setul de date original au fost extrase evenimentele individuale de consum. Evenimentele sunt caracterizate prin durata și volum total. Astfel, am analizat evenimentele extrase pentru fiecare tipar de colectare și am realizat gruparea în funcție de caracteristicile identificate, adică durata calculată în minute (x60 s) și volumul calculat în litri (L). În general, s-au găsit multe evenimente cu o durată egală sau mai mică de un minut (adică timpul de eșantionare al sistemului), și un volum total mai mic de 1 L, care au fost eliminate, pentru a filtra caracteristicile nedorite, de exemplu, scurgeri minore.

Clusterele sunt reprezentate în Figura 40 și Figura 41 pentru evenimentele colectate de la senzorii de la chiuveta (apă caldă și apă rece), în timp ce evenimentele de consum de apă de toaletă și duș sunt grupate în Figura 42 și Figura 43. Clusterele sunt prezentate în culori diferite, în timp ce centroizii lor sunt reprezentate de punctele galbene. Un număr optim de 3 grupuri a fost găsit utilizând metoda cotului și analiza siluetei clusterelor, cu excepția datelor de toaletă, care au necesitat 4 grupuri pentru o separare optimă. Cu toate acestea, am folosit 3 clustere pentru fiecare punct de colectare pentru a putea compara rezultatele.

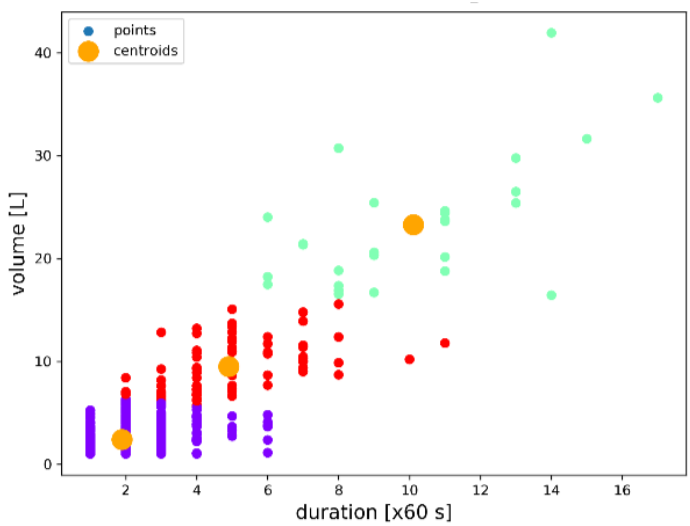


Figura 40 Gruparea evenimentelor (chiuveta apă caldă)

Astfel, se arată că există o variabilitate mai mare în consumul de apă rece de la robinet în comparație cu apa caldă, ceea ce se poate explica prin preferința față de apa caldă pentru spălarea mâinilor.

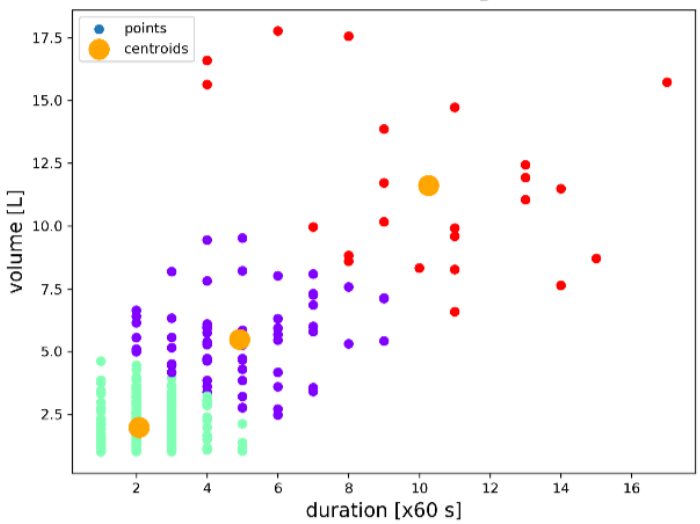


Figura 41 Gruparea evenimentelor (chiuveta apă rece)

Un rezultat foarte diferit se obține din gruparea evenimentelor de consum de apă de la toaletă, așa cum se arată în Figura 42, în sensul că durata este clar subliniată între diferite puncte de măsurare și volumul este în general aliniat cu durata. Acest comportament observat din date poate fi explicat prin volumul fix al rezervoarelor de apă de toaletă, care necesită o anumită perioadă de timp pentru a fi reumplute.

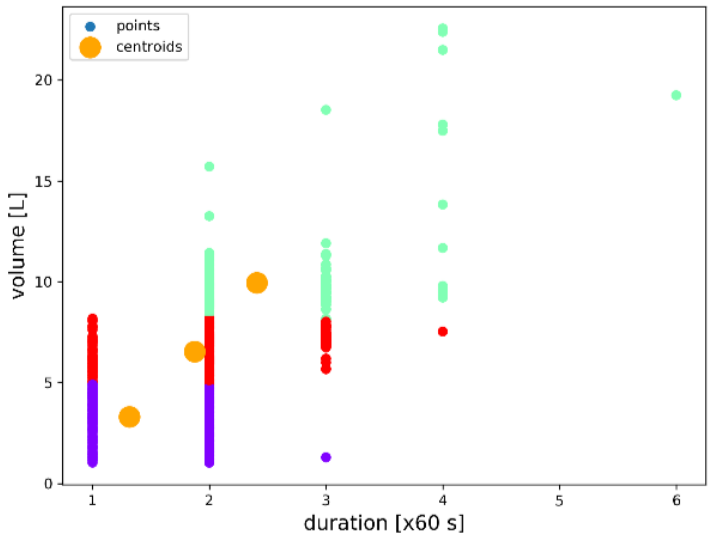


Figura 42 Gruparea evenimentelor (toaletă)

Au fost mai puține evenimente de duș în comparație cu celelalte activități înregistrate, așa cum se arată în Figura 43. Gruparea bazată pe evenimente relevă 3 tipuri de evenimente, crescând liniar atât ca durată, cât și ca volum, ceea ce explică un nivel mai mare de variabilitate în ceea ce privește apa de la duș.

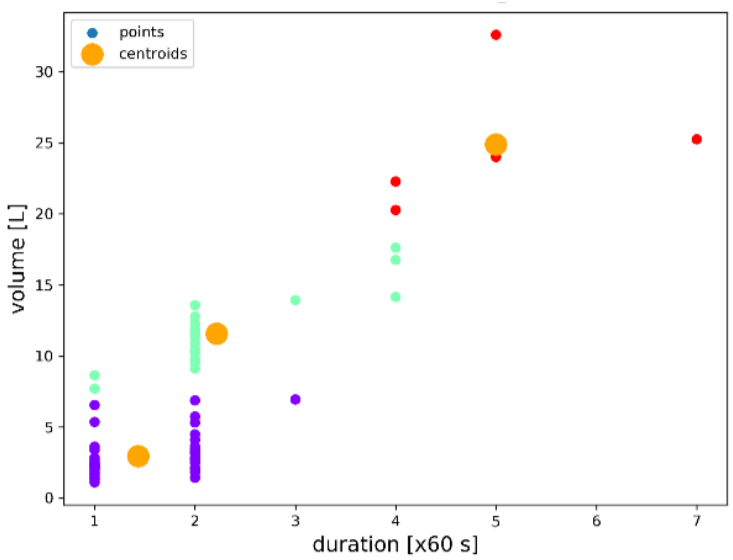


Figura 43 Gruparea evenimentelor (duș)

O monitorizare îndelungată asupra fiecărei grupări de evenimente sugerează că există, în general, un nivel mai mare de variabilitate pentru al treilea cluster, în comparație cu primele două grupări, având o durată și un volum mai mare. Prin urmare, am evaluat distribuția globală a clusterelor în termeni de dispersie. Rezultatele sunt reprezentate în Figura 44 pentru fiecare dintre aparatele măsurate și setul de date combinat. Într-adevăr, o dispersie mai mare este relevată pentru al treilea cluster în fiecare scenariu. Acest lucru sugerează că există mai multe anomalii asociate cu un consum mai mare de apă atât în ceea ce privește volumul, cât și durata, iar comportamentele de

consum mai scăzute sunt mai răspândite. În plus, graficul relevă o comparație între diferite puncte de colectare și/sau evenimente privind variabilitatea în ceea ce privește consumul. După cum era de așteptat, se constată că cea mai mică variabilitate o prezintă consumul de toaletă. Datele despre duș și apa rece arată o variabilitate similară, în timp ce apa caldă este cea mai variabilă în ceea ce privește comportamentul consumatorilor.

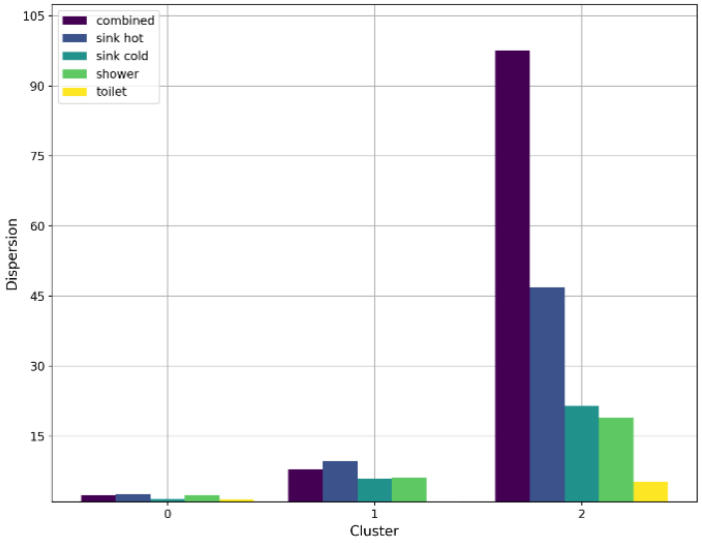


Figura 44 Distribuția clusterelor (folosind dispersia)

Rezultatele subliniază modul prin care comportamentul consumatorilor ar trebui încurajat spre reducerea consumului de apă caldă pentru a se alinia la rezultate mai durabile, deoarece există un nivel ridicat de variabilitate între diferiți consumatori.

În continuare, datele de la chiuvetă au fost combinate (apă caldă și apă rece), creând un alt scenariu pentru a oferi observații relevante cu privire la modelele de consum de apă.

Scenariul B UPB: În urma monitorizării consumului de apă, se dorește determinarea sursei unui eveniment de consum de apă, aceasta fiind caracterizată prin durata (măsurată în minute) și volumul de apă consumat (măsurat în litri). În acest scenariu, am folosit patru algoritmi de clasificare, doi algoritmi cu o abordare de învățare automată: Decision Tree și Random Forest, și doi algoritmi deep learning: adică Dense Neural Network și Recurrent Neural Network.

Pentru evaluarea calității rezultatelor, a fost efectuată o analiză bazată pe matricea de confuzie, așa cum este reprezentată în Figura 45, care arată o bună separare între punctele de consum prevăzute. Modelul Random Forest, care a produs cea mai mare acuratețe dintre modelele de clasificare propuse, a fost capabil să prezică punctul de consum corect pentru datele de toaletă și duș, în timp ce datele de la chiuvetă au fost uneori confundate cu datele de la toaletă.

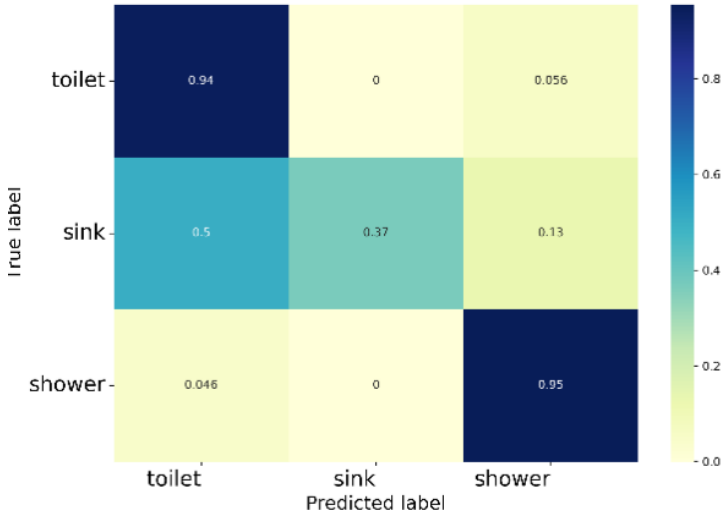


Figura 45 Matricea de confuzie

4.2.3 Raportarea evenimentelor cu suport blockchain

Rezultatul principal al dezvoltării aplicației Watergame pentru raportarea evenimentelor este obținerea unui model de laborator care integrează toate funcționalitățile propuse, și demonstrează capacitatea de integrare a scenariilor de gamificare cu tehnologia blockchain. În urma experimentării sistemului, evidența tranzacțiilor efectuate cu moneda virtuală WGT este accesibilă în rețeaua publică.

Astfel, fiecare acțiune efectuată de utilizatori prin intermediul criptomonedei, atât de către administratorul monedei, cât și de utilizatorii aplicației de crowdsensing sau prin intermediul altor metode publice de interacțiune cu blockchain va fi salvată într-un mod descentralizat în rețeaua publică de test Goerli sub forma unor tranzacții.

Fiecare tranzacție poate reprezenta una din următoarele operațiuni: creare contracte inteligente, apelare metode din contracte inteligente, cumpărare/vânzare token WGT, transfer token WGT și nu numai.

Toate aceste informații se vor putea accesa la adresa: https://goerli.etherscan.io/. Un exemplu de astfel de tranzacție este cea prin care a fost publicată în rețeaua de test moneda virtuală: https://goerli.etherscan.io/address/0xf3696ccc3cced64c20a97b555cd524522c9d0943

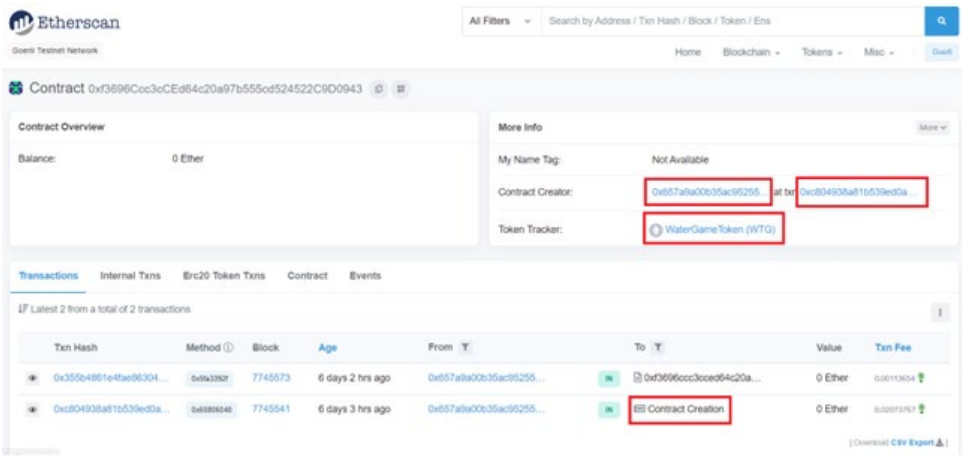


Figura 46 Pagina token-ului WGT

In Figura 46 se poate observa pagina publică a contractului monedei virtuale WGT. În cadrul acestei pagini se observă adresa contractului, adresa care a publicat moneda în rețea, adresa tranzacției corespunzătoare publicării sau numele monedei (WGT) .

Un alt exemplu de vizualizare a unei operații pe blockchain este pagina unei tranzacții (Figura 47). Aici se pot regăsi informații precum: hash-ul tranzacției, statusul, blocul în care a fost salvată, data și ora, adresa persoanei care a inițiat tranzacția și cu ce contract a interacționat sau taxele aferente tranzacției.

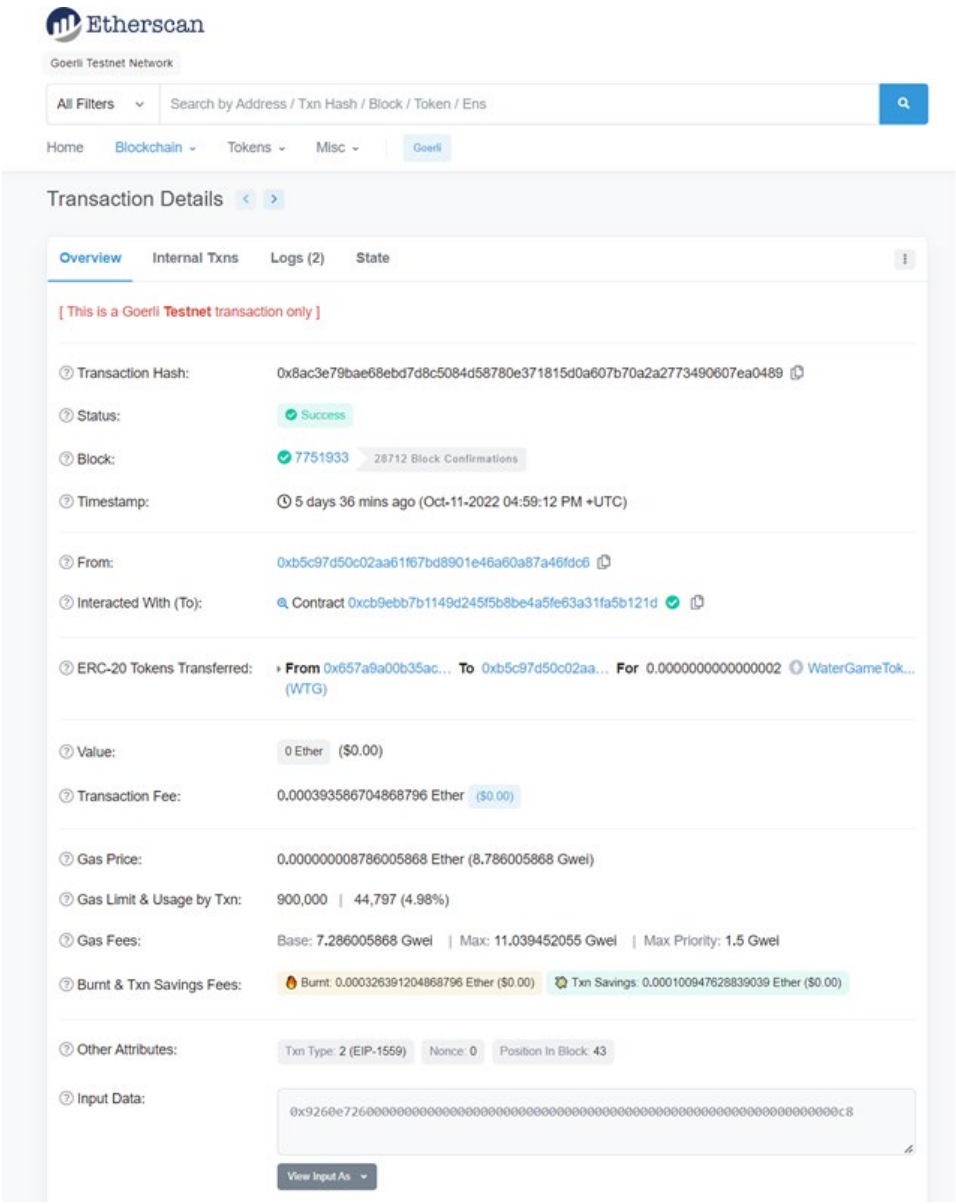


Figura 47 Vizualizarea unei tranzacții in etherscan

5 Bibliografie

Amazon Web Services. (2021). Overview of Amazon Web Services AWS Whitepaper. https://docs.aws.amazon.com/whitepapers/latest/aws-overview/aws-overview.pdf

Bell, C. (2016). MySQL for the Internet of Things. https://doi.org/10.1007/978-1-4842-1293-6

Chatzigeorgakidis, G. (2018). Household Water Consumption (Average), HELIX, 2018. https://data.hellenicdataservice.gr/dataset/236a0883-91b6-4f57-af9a-8d54ae7b154e (accessed Apr. 20, 2021).

Chen, S., Tang, X., Wang, H., Zhao, H., & Guo, M. (2016). Towards Scalable and Reliable In-Memory Storage System: A Case Study with Redis. https://doi.org/10.1109/TrustCom.2016.0255

Czako, Z., Sebestyen, G., & Hangan, A. (2021). AutomaticAI-A Hybrid Approach for Automatic Artificial Intelligence Algorithm Selection and Hyperparameter Tuning. Expert Systems with Applications, 115225, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.115225.

ethereum.org. (2021). Token ERC-20 Standard. https://ethereum.org/ro/developers/docs/standards/tokens/erc-20/

Ethereum Revision a57dd3c5. (2016). Getting Started - web3.js 1.0.0 documentation. https://web3js.readthedocs.io/en/v1.5.2/getting-started.html

Google. (2021a). Firebase Realtime Database. https://firebase.google.com/docs/database

Google. (2021b). Vehicle Routing Problem | OR-Tools | Google Developers. https://developers.google.com/optimization/routing/vrp

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

IBM. (2021). Getting to know MQTT. https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/

Khawas, C., & Shah, P. (2018). Application of Firebase in Android App Development-A Study. International Journal of Computer Applications, 179, 49–53. https://doi.org/10.5120/ijca2018917200

MetaMask. (2021). MetaMask - A crypto wallet & gateway to blockchain apps. https://metamask.io/index.html

Mishra, B., & Kertesz, A. (2020). The Use of MQTT in M2M and IoT Systems: A Survey. IEEE Access, 8, 201071–201086. https://doi.org/10.1109/ACCESS.2020.3035849

Neto, A. (2020). Developing for Ethereum, Test networks, Standalone Nodes, Solidity, and the Remix IDE. Medium. https://medium.com/@arlindont/developing-for-ethereum-9a0551f2b9d9

NodeMcu Team. (2018). NodeMcu -- An open-source firmware based on ESP8266 wifi-soc. https://www.nodemcu.com/index_en.html

Parihar, Y. S. (2019). Internet of Things and Nodemcu A review of use of Nodemcu ESP8266 in IoT products. 6, 1085.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

scikit-learn developers (BSD License). (n.d.). scikit-learn. http://scikit-learn.org/stable/

Xingjian, S. H. I., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In Advances in neural information processing systems (pp. 802-810).

Yermal, L., & Balasubramanian, P. (2017, December). Application of auto arima model for forecasting returns on minute wise amalgamated data in nse. In 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC) (pp. 1-5). IEEE

developer.nvidia.com. (2022). Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nano-developer-kit

espressif.com. (2022). ESP8266. https://www.espressif.com/en/products/socs/esp8266

fraunhoferiosb.github.io. (2022). The SensorThings API Data Model. https://fraunhoferiosb.github.io/FROST-Server/sensorthingsapi/requestingData/STA-Data-Model.html

Harris C. (2020). Array programming with NumPy. Nature, vol. 585, no. 7825, pp. 357–362.

Nair A. (2022). Beginner's Guide To K-Means Clustering. Analytics India Magazine. https://analyticsindiamag.com/beginners-guide-to-k-means-clustering/ (accessed Jan. 31, 2022).

Nakano, M., Takahashi, A. & Takahashi S. (2021). Generalized exponential moving average (EMA) model with particle filtering and anomaly detection. Expert Systems with Applications. Volume 73. 187-200.

NestJS (2022). Documentation | NestJS – A progressive Node.js framework (accesat 20 Oct 2022)

Nodemcu.readthedocs.io. (2022). NodeMCU. https://nodemcu.readthedocs.io/en/latest/

optimusdigital.ro. (2022). YF-S201 Debitmeter. https://www.optimusdigital.ro/ro/senzori-senzori-hall/3078-debitmetru-yf-s201-1-30-l-min.html

Pedregosa F. (2011). Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res., vol. 12, pp. 2825–2830.

Poyrazoglu G. (2021). Determination of Price Zones during Transition from Uniform to Zonal Electricity Market: A Case Study for Turkey. Energies, vol. 14.

Raiyn J. & Toledo T. (2014). Real-Time Road Traffic Anomaly Detection. Journal of Transportation Technologies. 256-266.

Rousseeuw P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math., vol. 20, pp. 53–65.

Syakur M., Khotimah B., Rochman E. & Satoto B. (2018). Integration K-Means Clustering Method and Elbow Method ForIdentification of The Best Customer Profile Cluster. IOP Conf. Ser. Mater. Sci. Eng., vol. 336, p. 12017.

Tim I. (2015). Anomaly detection on social media using ARIMA models. Uppsala University. Department of Information Technology.

Toni T., Resa S. P., Rezzy E. C., Solichatus Z., Youngjo L. & Rung C. C. (2021). Employing long short-term memory and Facebook prophet model in air temperature forecasting. Communications in Statistics - Simulation and Computation.

Yu Y., Si X., Hu C. & Zhang J. (2019). A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Neural Comput.

Zhagparov Z., Buribayev S., Joldasbayev A. & Zhassuzak M. (2021). Building a System for Predicting the Yield of Grain Crops Based On Machine Learning Using the XGBRegressor Algorithm. IEEE International Conference on Smart Information Systems and Technologies (SIST). 1-5.

Zhou Y. & Li J. (2019). Research of Network Traffic Anomaly Detection Model Based on Multilevel Autoregression. 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT), 380-384.