

UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DEPARTAMENTUL CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

PROIECT

MANAGEMENTUL ATACURILOR CIBERNETICE

COORDONATOR ȘTIINȚIFIC:
DRĂGAN MIHĂIȚĂ

STUDENT:
CÎMPEANU ANA-MARIA

BUCUREȘTI
2024-2025
UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DEPARTAMENTUL CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

**ALOCAREA RESURSELOR ȘI TRUST COMPUTING
PENTRU SISTEME DE CALCUL DE TIP EDGE
BLOCKCHAIN-ENABLED**

COORDONATOR ȘTIINȚIFIC:

DRĂGAN MIHĂIȚĂ

STUDENT:

CÎMPEANU ANA-MARIA

BUCUREȘTI

2024-2025

UNIVERSITATEA DIN BUCUREȘTI

Cuprins

1. Introducere	5
1.1 Aplicație	5
1.2 Motivația	5
2. Concepte	5
2.1 Blockchain	5
2.1.1 Ce este Blockchain-ul?	5
2.1.2 Blockchain-ul și securitatea Internet of Things (IoT)	6
2.1.3 Blockchain-ul în managementul atacurilor cibernetice	6
2.2 Edge Computing	6
2.2.1 Ce este edge computing-ul?	6
2.2.2 Edge computing și securitatea Internet of Things (IoT)	7
2.2.3 Edge computing în managementul atacurilor cibernetice	8
2.3 Trust Computing	8
2.3.1 Ce este trust computing-ul ?	8
2.3.2 Trust Computing și Internet of Things (IoT)	8
2.4 Conexiuni între concepte	9
2.4.1 Interacțiunea între blockchain și edge computing	9
2.4.2 Interacțiunea dintre trust computing și edge computing	9
2.4.3 Interacțiunea dintre blockchain și trust computing	9
2.4.4 Relația dintre blockchain, edge computing și trust computing	10
3. Arhitectura aplicației	11
3.1 Componentele principale	11
3.2 Fluxul datelor	11
3.2.1 Colectarea datelor	11
3.2.2 Simularea comportamentului malițios	12
3.2.3 Evaluarea scorurilor de încredere	12
3.2.4 Detectarea și reacția la comportamentul malițios	12
3.3 Funcția de trust computing	12
3.4 Managementul taskurilor	12
3.5 Securitatea sistemului	12
3.5.1 Metode de criptografie utilizate	12

4. Implementarea tehnică.....	13
4.1 Structura aplicației	13
4.2 Algoritmi utilizați	14
4.2.1 blockchain_integration.py	14
4.2.2 edge_device.py	14
4.2.3 task_management.py	15
4.2.4 trust_computation.py	15
4.2.5 malicious_behavior.py	15
4.2.6 main.py	16
4.3 Utilizarea aplicației.....	16
4.3.1 Rularea aplicației.....	16
4.3.2 Interfața aplicației.....	17
4.3.3 Vizualizare Grafice.....	19
5. Concluzie	22
6. Bibliografie	23

1. Introducere

1.1 Aplicație

„Allocation of Resources and Trust Computing for Blockchain-Enabled Edge Computing Systems” este o aplicație de simulare care vizează administrația și securitatea rețelelor de dispozitive edge. Integrează criptografia RSA și criptarea SSE pentru protecția datelor sensibile și testează securitatea împotriva atacurilor. Aplicația calculează scorurile de încredere ale dispozitivelor pe baza performanței și detectează comportamente malițioase, iar activitatea dispozitivelor este înregistrată în blockchain pentru transparență și securitate. Cu o interfață grafică intuitivă, aplicația generează diagrame pentru scorurile de încredere și alocarea sarcinilor, servind ca un instrument educativ pentru tehnologii avansate în edge computing și securitatea rețelelor.

1.2 Motivația

Motivația principală din spatele acestei aplicații derivă din dorința mea de a dezvolta un instrument care să sprijine înțelegerea unor concepte complexe legate de edge computing și securitatea rețelelor distribuite. Aceste domenii sunt, în opinia mea, fundamentale pentru viitorul infrastructurii digitale. Prin intermediul acestei simulări, intenționez să subliniez aspecte esențiale precum securitatea, nivelul de încredere și distribuirea eficientă a sarcinilor între diferitele dispozitive. Consider că este imperativ ca aplicațiile să includă o componentă vizuală ce utilizează grafice clare și intuitive, facilitând astfel interpretarea și analiza rezultatelor într-un mod mai accesibil.

2. Concepte

2.1 Blockchain

2.1.1 Ce este Blockchain-ul?

Aceasta introduce un sistem descentralizat, eliminând dependența de serverele centrale și facilitând interacțiunea directă între participanți, printr-un model peer-to-peer. În același timp, blockchain-ul creează o bază de date complet transparentă și accesibilă tuturor, contribuind semnificativ la creșterea transparenței în domenii precum guvernarea și procesele electorale.

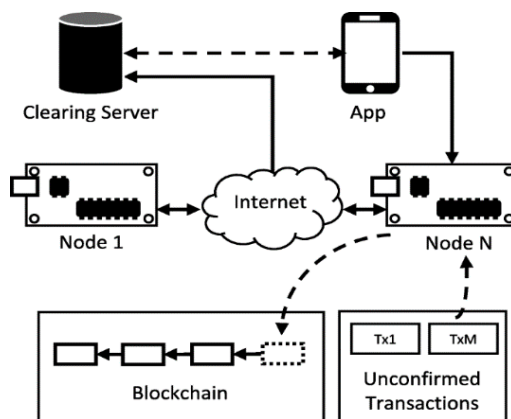


Figura 1 – Implementare blockchain

2.1.2 Blockchain-ul și securitatea Internet of Things (IoT)

Blockchain aduce un plus de securitate pentru Internet of Things (IoT) prin crearea unei rețele descentralizate. Aceasta înseamnă că nu există o autoritate centrală care să controleze totul, ceea ce reduce riscurile. În acest sistem, dispozitivele IoT generează tranzacții criptate pe care le trimit către nodurile blockchain pentru validare și stocare într-un registru distribuit. Acest proces face rețeaua mai rezistentă la accesul neautorizat.

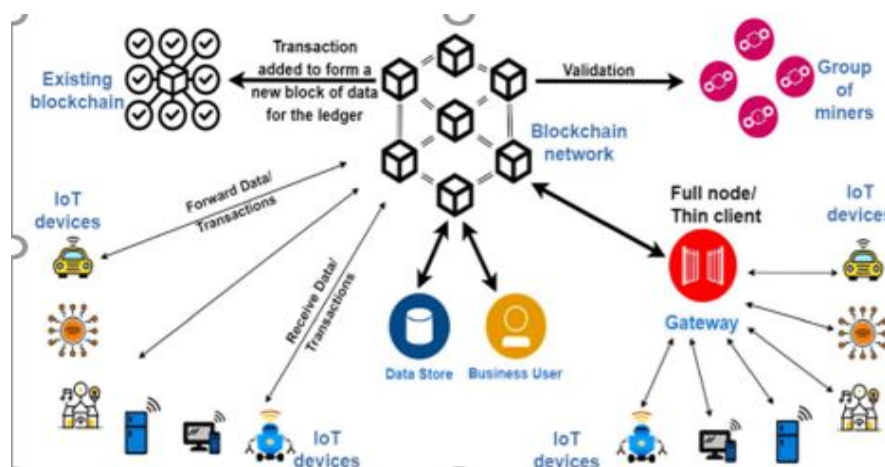


Figura 2 - Interacțiunea Blockchain și IoT

2.1.3 Blockchain-ul în managementul atacurilor cibernetice

Investiția în securitatea cibernetică bazată pe blockchain nu mai este o opțiune, ci o necesitate. Pe măsură ce atacurile cibernetice devin mai sofisticate și mai extinse, organizațiile trebuie să adopte tehnologii avansate pentru a-și proteja datele, operațiunile și încrederea clienților. Arhitectura descentralizată și imutabilă a blockchain-ului oferă soluția pentru construirea unui viitor digital mai sigur, în care atât afaceri cât și indivizi pot opera cu încredere, fără amenințarea constantă a criminalității cibernetice. Blockchain-ul joacă un rol esențial în prevenirea și gestionarea atacurilor cibernetice, oferind o protecție robustă prin criptare, validare descentralizată și transparență.

2.2 Edge Computing

2.2.1 Ce este edge computing-ul?

Edge computing este o modalitate de a procesa datele aproape de locul în care sunt generate, cum ar fi într-un magazin sau pe o linie de producție. În loc să trimitem toate datele la un centru de date departe, le procesăm direct acolo unde apar. Acest lucru ajută la evitarea problemelor de viteză și întreruperi ale rețelei, permițând

companiilor să reacționeze rapid la datele colectate în timp real prin senzori și dispozitive IoT. După ce datele sunt procesate, rezultatele sunt trimise înapoi la centrul de date pentru analiză suplimentară, făcând astfel IT-ul mai eficient și mai adaptat nevoilor afacerilor.

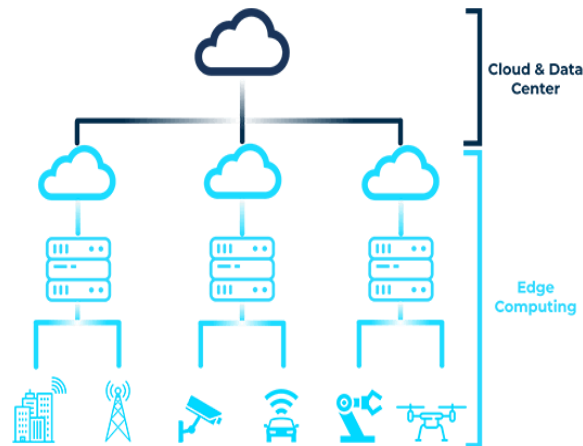


Figura 3 - Implementare Edge Computing

2.2.2 Edge computing și securitatea Internet of Things (IoT)

Edge computing este foarte important pentru îmbunătățirea aplicațiilor IoT. Ajută la reducerea timpilor de reacție și la utilizarea mai bună a lățimii de bandă. Prin procesarea datelor aproape de dispozitivele IoT, edge computing permite reacții rapide, esențiale pentru lucruri precum sistemele de răcire sau întreținerea echipamentelor. De asemenea, ajută la gestionarea eficientă, procesând și filtrând datele direct la sursă, astfel încât să se trimită doar informațiile necesare pentru stocare pe termen lung. Așezarea serverelor mai aproape îmbunătățește și securitatea IoT, păstrând datele în zonă și controlând accesul. Acest model ajută și la respectarea regulilor privind protecția datelor.

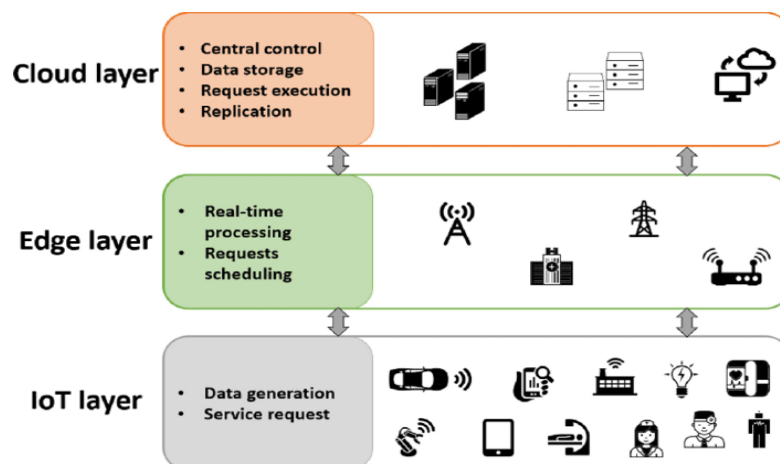


Figura 4 - Interacțiune Edge Computing și Internet of Things

2.2.3 Edge computing în managementul atacurilor cibernetice

Edge computing contribuie semnificativ la managementul atacurilor cibernetice prin îmbunătățirea timpilor de răspuns, procesând datele local, aproape de sursa acestora. Astfel, amenințările pot fi detectate și gestionate mult mai rapid. De asemenea, procesarea locală a datelor ajută la respectarea datelor, reducând riscurile legale și de reglementare. În plus, prin minimizarea transferului de date între dispozitive și servere centrale, se reduce suprafața de atac, limitând astfel posibilitățile hackerilor de a exploata vulnerabilitățile rețelei.

2.3 Trust Computing

2.3.1 Ce este trust computing-ul ?

Trusted computing este o tehnologie care asigură că un sistem informatic funcționează într-un mod previzibil și securizat, protejând datele și software-ul împotriva accesului neautorizat. Prin utilizarea unor mecanisme de autentificare și criptare, aceasta garantează integritatea și confidențialitatea informațiilor, creând un mediu de încredere pentru procesarea și stocarea datelor.

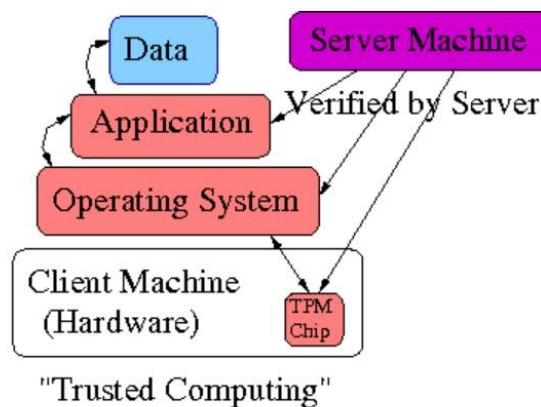


Figura 5 - Implementare Trust Computing

2.3.2 Trust Computing și Internet of Things (IoT)

În domeniul IoT, securitatea este abordată la nivel hardware, software și comunicații. Proiectul Trusted-IoT se concentrează pe securitatea hardware, utilizând tehnici pentru module de securitate pe dispozitive IoT. Acesta include dezvoltarea unui cadru open-source pentru ARMv8 cu TrustZone și demonstrații de criptare și autentificare a datelor. În cazul RISC-V, cercetările au arătat cum centralizarea proceselor pe o platformă FPGA îmbunătățește scalabilitatea și eficiența validării dispozitivelor. Aceste cercetări subliniază importanța Trusted Computing în asigurarea securității IoT.

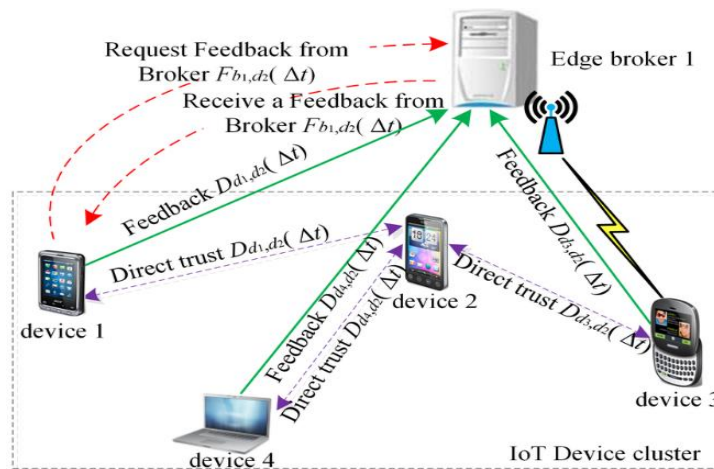


Figure 6 - Un exemplu de calcul al încrederii în Edge Computing IoT.

2.4 Conexiuni între concepte

2.4.1 Interacțiunea între blockchain și edge computing

Blockchain-ul și edge computing-ul colaborează pentru a îmbunătăți securitatea și performanța sistemelor distribuite. Blockchain-ul abordează vulnerabilitățile din edge computing, precum autentificarea dispozitivelor IoT și protecția împotriva atacurilor DoS, asigurând totodată integritatea datelor. Datorită caracteristicilor sale imuabile și descentralizate, oferă trasabilitate și previne modificările neautorizate prin hash-uri și semnături digitale. Contractele inteligente îmbunătățesc transparența și încrederea, facilitând gestionarea automată a tranzacțiilor între dispozitivele IoT.

2.4.2 Interacțiunea dintre trust computing și edge computing

Interacțiunea dintre trust computing și edge computing este esențială pentru asigurarea unui ecosistem distribuit sigur și eficient. Trust computing permite evaluarea fiabilității dispozitivelor edge prin calcularea unor scoruri de încredere bazate pe comportamentul și performanța acestora. Aceste scoruri ajută la identificarea dispozitivelor compromise sau malițioase, prevenind accesul acestora la resurse critice și reducând riscurile de securitate. În contextul edge computing, trust computing optimizează alocarea resurselor prin direcționarea sarcinilor către dispozitive de încredere.

2.4.3 Interacțiunea dintre blockchain și trust computing

Interacțiunea dintre blockchain și trust computing ajută la crearea unui sistem mai sigur și transparent. Blockchain-ul păstrează scorurile de încredere calculate de trust computing într-un mod care nu poate fi modificat, asigurând corectitudinea și verificarea ușoară a acestor date. În același timp, trust computing evaluează cât de încredere sunt dispozitivele și participanții din rețea, contribuind la siguranța generală. Împreună, cele două tehnologii lucrează pentru a construi un sistem distribuit în care datele sunt sigure, iar resursele sunt gestionate corect.

2.4.4 Relația dintre blockchain, edge computing și trust computing

Relația dintre blockchain, edge computing și trust computing este explicată într-un sistem propus de cercetătorii Lejun Zhang, Yanfei Zou, Weizheng Wang, Zilong Jin, Yansen Su și Huiling Chen. Acest sistem combină alocarea resurselor și evaluarea încrederii, folosind blockchain pentru gestionarea sarcinilor și a datelor într-un mod sigur și eficient. Trust computing evaluează încrederea dispozitivelor din rețea, asigurându-se că doar dispozitivele de încredere primesc sarcini importante. De asemenea, sunt folosite tehnici pentru a reduce întârzierile și pentru a permite o căutare rapidă și sigură a datelor. În plus, un algoritm de consens optimizat ajută la îmbunătățirea performanței sistemului. Aceste soluții arată cum blockchain, edge computing și trust computing pot lucra împreună pentru a rezolva problemele rețelelor distribuite.

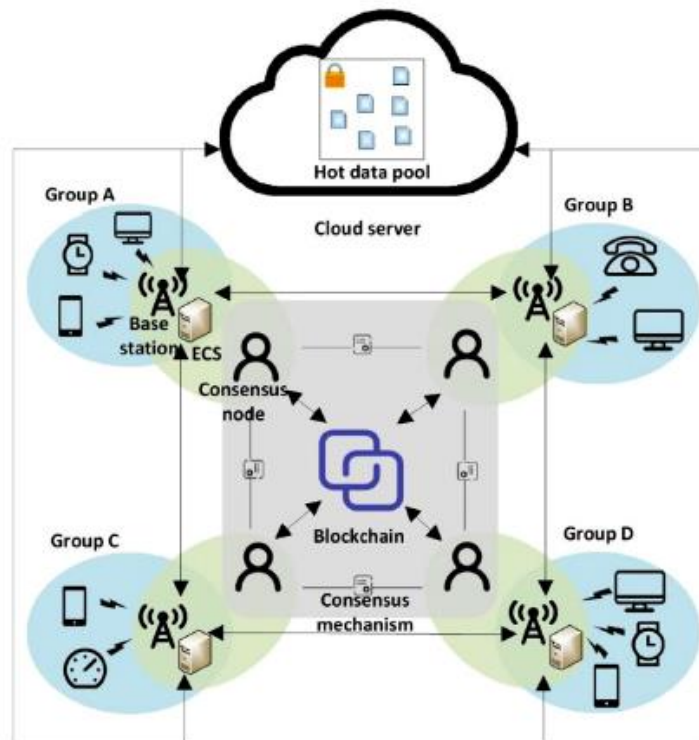


Figure 7 - Interacțiune blockchain , edge computing , trust computing

3. Arhitectura aplicației

3.1 Componentele principale

Aplicația are la bază o arhitectură destul de simplă, cu un scop clar, îmbinând componente care colaborează pentru a îndeplini scopul general al sistemului.

Componentă	Descriere
Dispozitive Edge	Dispozitivele de la marginea rețelei (edge devices) sunt echipamente care colectează și procesează datele local, înainte de a le trimite la un server central. Fiecare dispozitiv are atribute precum performanța CPU, memoria și viteza de procesare, care influențează încrederea atribuită pentru realizarea sarcinilor.
Blockchain	Blockchain-ul în această aplicație are scopul de a urmări comportamentul dispozitivelor și pentru a oferi un mecanism de încredere între entitățile care comunică în cadrul rețelei.
Trust Computing	Trust computing-ul este componenta principală responsabilă de calcularea scorurilor de încredere ale dispozitivelor edge.
Gestionarea taskurilor	Managementul taskurilor reglează distribuirea sarcinilor între dispozitivele edge, ținând cont de capacitățile acestora și de scorurile lor de încredere. Acest modul asigură un flux eficient al muncii, distribuind sarcini dispozitivelor capabile să le rezolve cel mai eficient.
Interfața cu utilizatorul (GUI)	Interfața cu utilizatorul (GUI) permite utilizatorilor să vizualizeze scorurile de încredere ale dispozitivelor și să adauge sarcini. Aplicația oferă opțiuni pentru ajustarea cerințelor CPU, memorie și termene limită pentru sarcini. De asemenea, permite generarea de grafice pentru vizualizarea distribuției sarcinilor și a scorurilor de încredere.

3.2 Fluxul datelor

3.2.1 Colectarea datelor

Datele sunt colectate de la fiecare dispozitiv edge, care include atribute precum **precizia, viteza, CPU și memoria**. Aceste date sunt utilizate pentru a evalua performanța dispozitivelor și pentru a calcula scorurile de încredere pe baza interacțiunilor lor. Fiecare dispozitiv monitorizează performanța altora și generează scoruri de interacțiune.

3.2.2 Simularea comportamentului malițios

Dispozitivele malițioase sunt simulate pentru a testa reacția aplicației la comportamente defectuoase. Aceste dispozitive influențează negativ scorurile de încredere și pot provoca eșecuri în finalizarea sarcinilor. Comportamentele malițioase sunt folosite pentru a evalua securitatea și stabilitatea sistemului.

3.2.3 Evaluarea scorurilor de încredere

Scorurile de încredere sunt ajustate în funcție de performanța dispozitivelor și de succesul în finalizarea sarcinilor. Interacțiunile dintre dispozitive și sarcinile îndeplinite sau eșuate influențează aceste scoruri. Dispozitivele cu scoruri mari primesc sarcini, iar cele cu scoruri scăzute sunt excluse.

3.2.4 Detectarea și reacția la comportamentul malițios

Comportamentele malițioase sunt detectate prin monitorizarea activității dispozitivelor. Dacă un dispozitiv se comportă malițios, scorul său de încredere scade și este exclus din alocarea sarcinilor. Aplicația asigură astfel securitatea rețelei, prevenind efectele negative ale dispozitivelor malițioase.

3.3 Funcția de trust computing

Funcția *trust_computation* este o componentă importantă în parcursul aplicației-simulare pe care am încorporat-o, având rolul de a evalua și gestiona dispozitivele de încredere din rețea. Aceasta evaluează scorurile de încredere ale dispozitivelor pe baza interacțiunilor lor și a caracteristicilor relevante, precum acuratețea și viteza, stabilind un nivel de încredere pentru fiecare dispozitiv.

3.4 Managementul taskurilor

Funcția *task_management* gestionează și organizează sarcinile destinate dispozitivelor edge. Sarcinile sunt stocate într-un task stack, ordonate după prioritate și termen limită, pentru a asigura procesarea celor mai urgente sarcini. Componenta permite adăugarea, accesarea și distribuirea sarcinilor către dispozitive, optimizând utilizarea resurselor și integrându-se eficient cu sistemul de evaluare a încrederii.

3.5 Securitatea sistemului

3.5.1 Metode de criptografie utilizate

3.5.1.1 RSA – autentificare și semnătura mesajelor

RSA este utilizat pentru autentificarea dispozitivelor și asigurarea integrității datelor prin semnături digitale. Fiecare dispozitiv are o pereche de chei publice și private, folosite pentru a semna și verifica mesaje, garantând că acestea provin dintr-o sursă de încredere.

3.5.1.2 SSE – criptarea datelor sensibile

SSE (Searchable Symmetric Encryption) protejează datele sensibile prin criptare, permițând totodată căutarea acestora fără a compromite

confidențialitatea. Această metodă asigură că doar utilizatorii autorizați pot accesa datele și efectua operații asupra lor.

4. Implementarea tehnică

4.1 Structura aplicației

Aplicația a fost dezvoltată în **Python**, folosind **Visual Studio Code** ca mediu de dezvoltare. Structura proiectului este organizată într-un mod organizat, fiecare componentă fiind responsabilă pentru o funcționalitate specifică. Fișierele și directoarele din aplicație sunt structurate după cum urmează:

- Director rădăcină

1. `main.py` – Acest fișier se ocupă cu inițializarea componentelor și cu gestionarea fluxului principal al datelor.
2. `edge_device.py` – Definește clasa `EdgeDevice`, responsabilă pentru gestionarea dispozitivelor.
3. `trust_computation.py` - Se ocupă de calculul scorurilor de încredere și identificarea dispozitivelor fiabile.
4. `task_management.py` – Implementează clasa `TaskManager` pentru gestionarea și distribuirea sarcinilor către dispozitivele edge.
5. `blockchain_integration.py` – Gestionează integrarea blockchain pentru securizarea datelor.
6. `malicious_behavior.py` – Simulează și detectează comportamentele malițioase în rețea.
7. `rsa_auth.py` – Implementează metoda pentru criptografia RSA.
8. `sse.py` – Realizează criptarea datelor sensibile folosind `Searchable Symmetric Encryption`.
9. `plot_utils.py` – Conține funcții pentru generarea și vizualizarea graficelor.
10. `gui_interface.py` – Definește interfața grafică a aplicației, permițând utilizatorilor să interacționeze cu sistemul.
11. `main.spec` – Fișier generat automat pentru construirea executabilului folosind `PyInstaller`.

- Subdirectoare

1. `tests/` - Conține fișierele de testare pentru validarea funcționalităților aplicației.
2. `__pycache__/` și `.pytest_cache` – Directoare generate automat pentru stocarea fișierelor compilate și a rezultatelor testelor.
3. `build/` - Conține fișierele generate în timpul construirii aplicației.
4. `dist/` - Directorul final unde este salvat executabilul.

4.2 Algoritmi utilizați

4.2.1 blockchain_integration.py

1. Algoritmul hash - Funcția hash utilizează algoritmul criptografic SHA-256 pentru a transforma conținutul unui bloc într-un identificator unic de lungime fixă (256 biți). Hash-ul se bazează pe conținutul blocului (date, timestamp, proof), ceea ce asigură integritatea datelor. Dacă un bloc este modificat, hash-ul generat va fi complet diferit. Această funcție transformă blocul într-un string JSON ordonat, îl codifică în bytes și aplică algoritmul SHA-256. Rolul său principal este să genereze un hash care identifică unic fiecare bloc din lanț și leagă blocurile între ele.
2. Algoritmul de verificare a lanțului - Această funcție verifică integritatea lanțului de blocuri asigurându-se că hash-ul fiecărui bloc este corect conectat la previous_hash al blocului următor. Dacă un singur bloc din lanț este modificat, verificarea va eșua, indicând compromiterea datelor. Această buclă iterează prin fiecare pereche de blocuri consecutive din lanț, comparând hash-ul generat pentru blocul precedent cu valoarea previous_hash stocată în blocul curent. Dacă există discrepanțe, funcția returnează False.

Aceste două componente asigură conectarea blocurilor și protecția împotriva alterării datelor. Restul funcționalităților, precum create_block, gestionează adăugarea blocurilor noi, iar add_data permite stocarea datelor în blockchain, menținând integritatea generală.

4.2.2 edge_device.py

1. Scorul de interacțiune - Când două dispozitive interacționează, se calculează un scor pe baza atributelor lor (precum acuratețea, viteza, CPU-ul și memoria). Acest scor reflectă cât de bine se potrivesc cele două dispozitive în funcție de greutatea atribuite fiecărui atribut. Practic, pentru fiecare atribut al dispozitivului, valoarea respectivă este înmulțită cu greutatea atribuită acelui atribut și toate aceste valori sunt adunate pentru a obține scorul final.
2. Verificarea capacității de a îndeplini sarcina - Dând un dispozitiv primește o sarcină, acesta verifică dacă resursele sale (CPU și memorie) sunt suficiente pentru a o îndeplini. Dacă cerințele de CPU și memorie ale sarcinii sunt mai mici decât resursele disponibile pe dispozitiv, sarcina poate fi realizată. În caz contrar, dispozitivul nu va putea să o efectueze.

3. Actualizarea scorului de încredere - După finalizarea unei sarcini, dispozitivul își actualizează scorul de încredere. Dacă sarcina a fost realizată cu succes, scorul crește cu o valoare fixă, iar dacă nu a fost realizată, scorul scade. Totodată, scorul de încredere nu poate depăși valoarea maximă de 1 și nu poate scădea sub 0, astfel că se asigură că acest scor rămâne întotdeauna într-un interval valid.

4.2.3 task_management.py

1. Sortarea sarcinilor -Sortează sarcinile pe baza termenului limită (deadline). Sarcinile cu deadline mai apropiat vor fi procesate mai întâi, asigurându-se astfel că sunt gestionate corect în funcție de urgență.
2. Adăugarea unei sarcini – adaugă o sarcină în stiva de sarcini (task_stack). Sarcina este adăugată la sfârșitul listei.
3. Obținerea următoarei sarcini - eturnează prima sarcină din stiva de sarcini (cea cu deadline-ul cel mai apropiat), utilizând pop(0) pentru a o elimina din listă.

4.2.4 trust_computation.py

1. Inițializare - Constructorul primește dispozitivele edge și un set de ponderi pentru a calcula scorurile de încredere. Dacă nu sunt specificate ponderi, acestea sunt setate implicit la {"accuracy": 0.7, "speed": 0.3}. De asemenea, se stabilește un prag (threshold) pentru scorurile de încredere sub care dispozitivele sunt eliminate.
2. Calcularea scorului de încredere - Funcția *compute_trust* calculează scorurile de încredere pentru fiecare dispozitiv, în funcție de interacțiunile cu celelalte dispozitive. Dacă scorul unui dispozitiv este sub un anumit prag, acesta este eliminat din sistem. De asemenea, dispozitivul cu cel mai mare scor devine agentul principal.

4.2.5 malicious_behavior.py

1. Inițializare - Constructorul primește lista de dispozitive edge și o stochează pentru a putea simula comportamentele malițioase asupra acestora.
2. Simularea comportamentului malițios - Metoda *simulate_malicious_activities* parcurge lista de dispozitive și atribuie un comportament malițios pentru un dispozitiv cu o probabilitate specificată (default 30% - malicious_probability=0.3). Dacă dispozitivul este considerat malițios (se întâmplă dacă valoarea aleatorie generată este mai mică decât probabilitatea), scorul de încredere al acestuia este setat la 0, semnalând că dispozitivul este complet neîncredere. Prin această simulare, dispozitivele malițioase pot fi identificate și excluse din rețea pe baza unui scor de încredere scăzut, afectând astfel dinamica și securitatea rețelei distribuite.

4.2.6 main.py

Funcția `main()` are scopul de a coordona funcționarea întregului sistem, de la inițializarea dispozitivelor edge până la procesarea datelor și afișarea rezultatelor. Mai întâi, se creează un set de dispozitive edge, inclusiv unul malițios, prin apelarea funcției `initialize_devices()`. Apoi, calculul încrederii dispozitivelor se face folosind clasa `TrustComputing`, care evaluează și actualizează scorurile de încredere ale dispozitivelor pe baza interacțiunilor între ele. Comportamentele malițioase sunt simulate cu ajutorul clasei `MaliciousBehaviorSimulator`, care modifică scorurile pentru dispozitivele malițioase. Funcțiile `rsa_authentication_demo()` și `sse_encryption_demo()` sunt utilizate pentru a demonstra criptarea și semnătura mesajelor. După aceea, sarcinile sunt gestionate de clasa `TaskManager`, care prioritizează sarcinile pe dispozitive în funcție de deadline-uri. În final, o interfață grafică este lansată prin Tkinter, iar graficele generate cu `plot_trust_scores()` și `plot_task_distribution()` permit vizualizarea performanței dispozitivelor și distribuția sarcinilor.

4.3 Utilizarea aplicației

4.3.1 Rularea aplicației

Aplicația poate fi rulată pe orice sistem compatibil Windows, fiind disponibilă sub forma unui fișier executabil **main.exe**. Acest fișier se află în subdirectorul „disc” al proiectului **EdgeComputing**.

Dublu-click pe fișierul **main.exe** pentru a lansa aplicația.

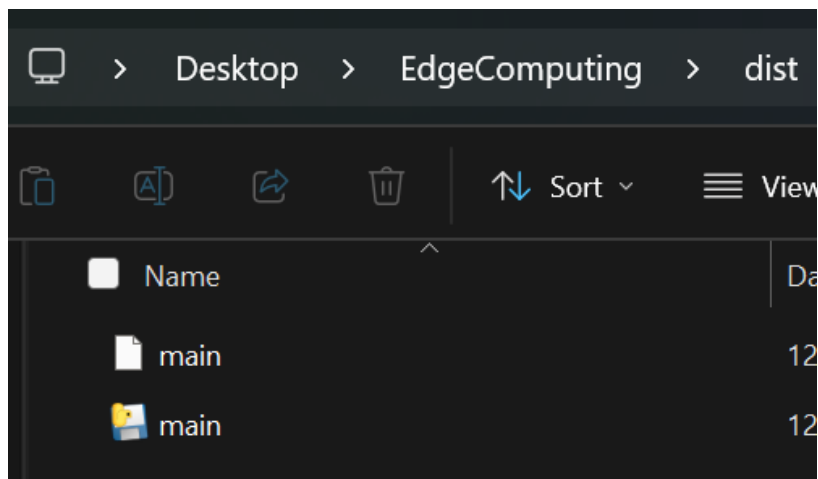
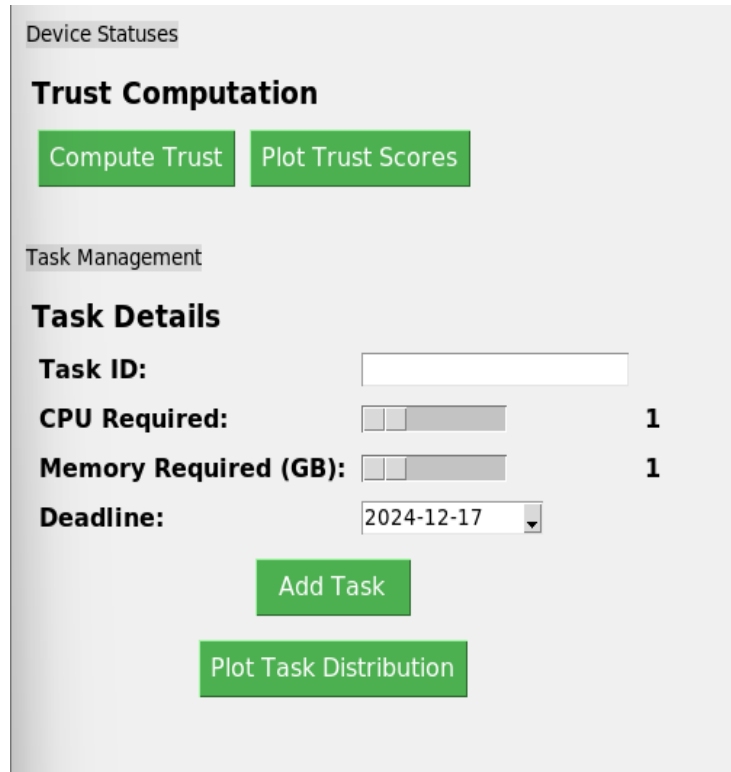


Figura 8 - Fișierul de lansare a aplicației

4.3.2 Interfața aplicației

Aplicația dispune de o interfață grafică intuitivă, care permite utilizatorilor să interacționeze ușor cu funcționalitățile esențiale ale sistemului. Aceste funcționalități sunt grupate în secțiuni principale, care includ gestionarea sarcinilor și calculul încrederii.



The screenshot displays a web application interface with a light gray background. At the top, there is a section titled 'Device Statuses' in a small, dark font. Below this, the 'Trust Computation' section is highlighted with a bold title. It contains two green buttons: 'Compute Trust' and 'Plot Trust Scores'. Further down, the 'Task Management' section is also highlighted with a small, dark font. Below it, the 'Task Details' section is highlighted with a bold title. This section contains four rows of input fields: 'Task ID:' with a text input box; 'CPU Required:' with a progress bar and the number '1'; 'Memory Required (GB):' with a progress bar and the number '1'; and 'Deadline:' with a date input box showing '2024-12-17' and a dropdown arrow. At the bottom of the 'Task Details' section, there are two green buttons: 'Add Task' and 'Plot Task Distribution'.

Figura 9 - Interfața grafică a aplicației

Adăugarea unei sarcini

1. În secțiunea „Task Details” (Detalii Task), utilizatorul poate introduce un ID de task, specificând resursele necesare pentru CPU și memorie, precum și termenul limită (Deadline).

După completarea câmpurilor necesare, utilizatorul apasă pe butonul „Add Task” pentru a adăuga sarcina în stack-ul de sarcini.

Task Management

Task Details

Task ID:

CPU Required: **3.6**

Memory Required (GB): **21.5**

Deadline:

Figura 10 - Exemplu introducere date

2. Calcularea Încrederii:

După adăugarea taskurilor, utilizatorul poate utiliza butonul „Compute Trust” pentru a efectua calculul încrederii dispozitivelor edge, pe baza interacțiunii dintre ele și a performanței acestora.

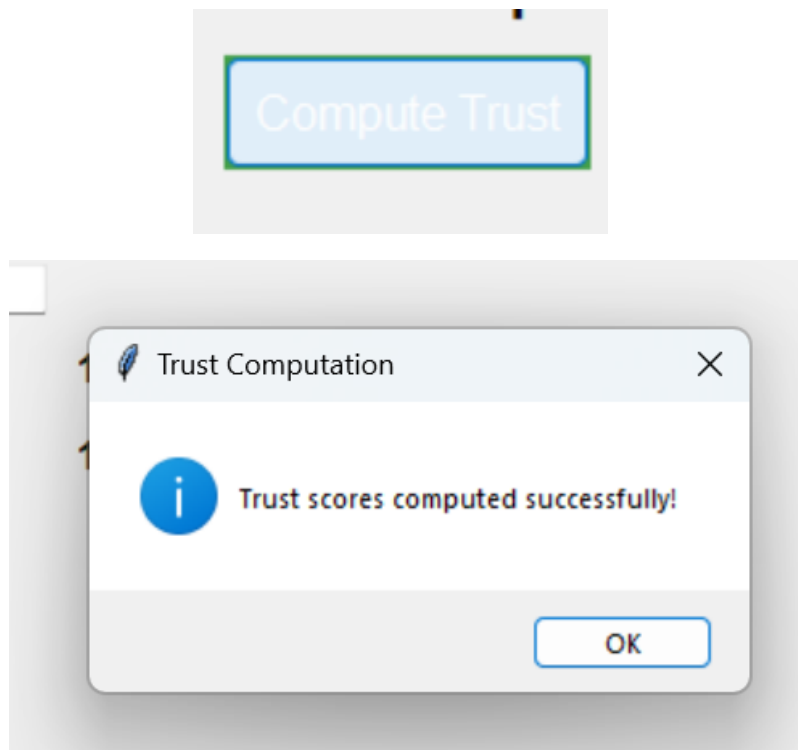


Figura 11 – 12 - Compilarea algoritmului și validarea

4.3.3 Vizualizare Grafice

După calculul încrederii, utilizatorul poate vizualiza două grafice esențiale:

- **Plot Trust Scores:** Acest grafic afișează scorurile de încredere ale dispozitivelor edge.
- **Plot Task Distribution:** Acest grafic arată distribuția sarcinilor adăugate și termenele limită asociate acestora.

Ambele grafice sunt generate atunci când utilizatorul apasă pe butoanele corespunzătoare, iar acestea sunt afișate pe ecran pentru o analiză vizuală clară a performanței dispozitivelor și sarcinilor.

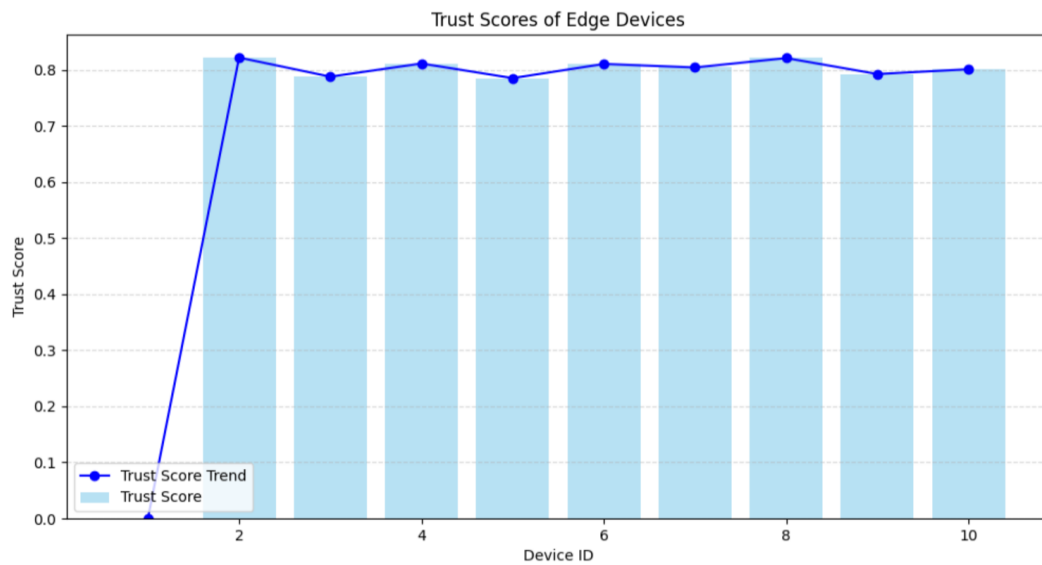
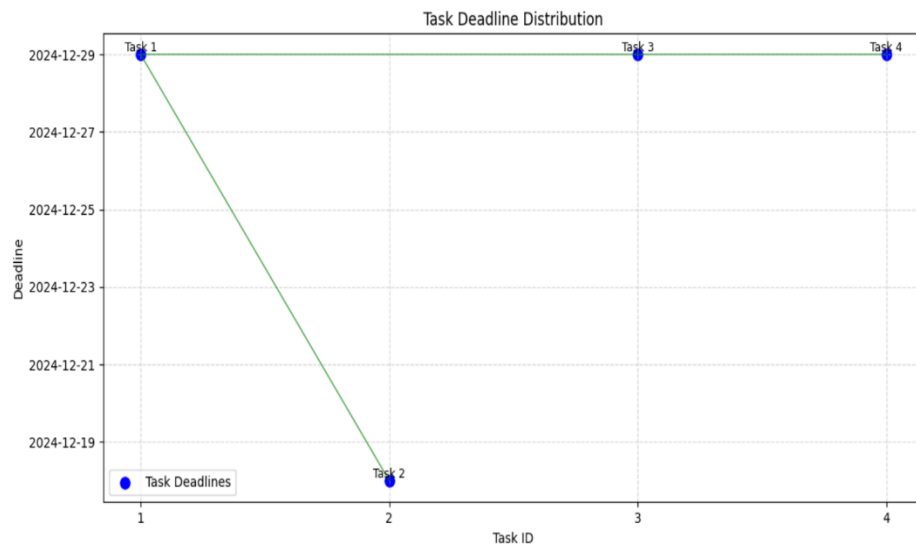


Figura 13 - Exemplu calcul de încredere pentru 10 edge devices și 5 sarcini



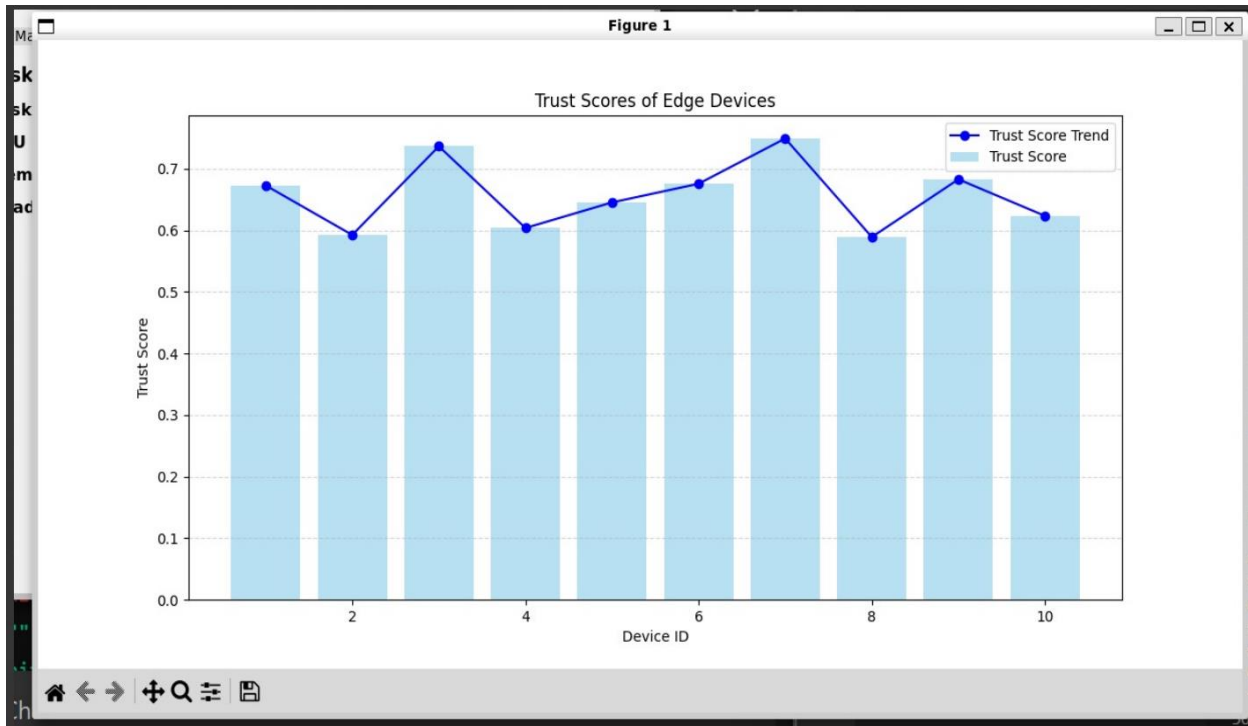


Figura 15 - Exemplu calcul de încredere pentru 10 edge devices și 8 sarcini

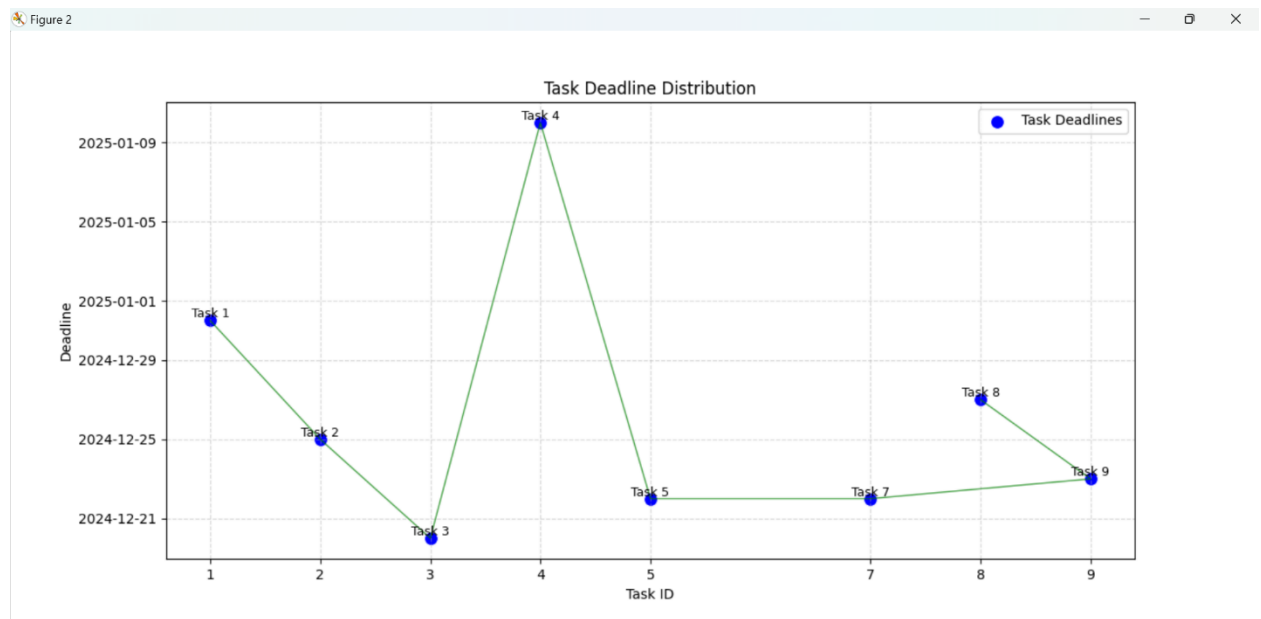


Figure 16 - Distribuirea sarcinilor

În timpul rulării aplicației, terminalul va afișa diverse informații care reflectă procesele interne ale aplicației. De exemplu, pentru fiecare interacțiune între dispozitive, se va calcula un „interaction score” (scor de interacțiune), care indică gradul de încredere între două dispozitive pe baza atributelor lor. Aceste scoruri vor fi afișate sub forma unui mesaj, cum ar fi „Interaction score between Device X and Device Y: Z”. De asemenea, după calculul scorurilor de încredere, va apărea o listă cu ID-urile dispozitivelor și scorurile lor actuale de încredere, ceea ce permite urmărirea evoluției acestora pe măsură ce aplicația procesează datele. Aceste informații sunt utile pentru înțelegerea modului în care dispozitivele interacționează între ele și pentru a valida corectitudinea procesului de calcul a încrederii.

```

Skipping Device 1 (malicious).
Interaction score between Device 2 and Device 1: 0.90
Interaction score between Device 2 and Device 3: 0.88
Interaction score between Device 2 and Device 4: 0.82
Interaction score between Device 2 and Device 5: 0.69
Interaction score between Device 2 and Device 6: 0.69
Interaction score between Device 2 and Device 7: 0.94
Interaction score between Device 2 and Device 8: 0.79
Interaction score between Device 2 and Device 9: 0.58
Interaction score between Device 2 and Device 10: 0.54
Interaction score between Device 3 and Device 1: 0.90
Interaction score between Device 3 and Device 2: 0.60
Interaction score between Device 3 and Device 4: 0.82
Interaction score between Device 3 and Device 5: 0.69
Interaction score between Device 3 and Device 6: 0.69
Interaction score between Device 3 and Device 7: 0.94
Interaction score between Device 3 and Device 8: 0.79
Interaction score between Device 3 and Device 9: 0.58
Interaction score between Device 3 and Device 10: 0.54
Interaction score between Device 4 and Device 1: 0.90
Interaction score between Device 4 and Device 2: 0.60
Interaction score between Device 4 and Device 3: 0.88
Interaction score between Device 4 and Device 5: 0.69
Interaction score between Device 4 and Device 6: 0.69

```

```
Device IDs: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
Trust Scores: [0.0, 0.7582566989625416, 0.7273965436302758, 0.7341083940642209, 0.7490646695974399, 0.7487749137431121,
0.7213904354812386, 0.7374407422811794, 0.7604009915225903, 0.7655533993607019]
```

Figura 17 - Informații terminal - interacțiune , calcul de încredere , devices

5. Concluzie

Dezvoltarea acestei aplicații m-a ajutat să înțeleg mai bine conceptele de alocare a resurselor și calcul al încrederii (trust computing) în cadrul sistemelor Edge Computing, având în vedere și integrarea tehnologiilor blockchain. Aplicația, deși simplă, mi-a oferit o bază solidă pentru a înțelege cum pot fi gestionate resursele și asigurată încrederea în astfel de medii distribuite. Blockchain-ul adaugă un strat suplimentar de securitate și transparență, iar integrarea sa în sistemele Edge Computing este un pas important pentru a crea soluții scalabile și sigure. Deși aplicația nu este o implementare completă pentru scenarii reale, m-a ajutat să înțeleg pașii esențiali pentru implementarea unor astfel de soluții și m-a pregătit pentru explorarea unor aplicații mai complexe în viitor.

6. Bibliografie

- <https://ro.wikipedia.org/wiki/RSA>
- <https://www.run.ai/guides/edge-computing/edge-computing-in-iot>
- <https://www.upguard.com/blog/the-role-of-cybersecurity-in-blockchain-technology>
- <https://www.youtube.com/watch?v=AvSCzPPzTXQ>
- <https://cs.stanford.edu/people/eroberts/cs201/projects/trusted-computing/what.html>
- <https://www.irit.fr/tciot/index.html>
- https://link.springer.com/chapter/10.1007/978-3-031-55673-9_17
- <https://pyinstaller.org/en/stable/installation.html>
- <https://www.chakray.com/blockchain-iot-security/>
- <https://www.geeksforgeeks.org/blockchain-to-secure-iot-data/>
- <https://onlinelibrary.wiley.com/doi/toc/10.1155/2037.si.741656>
- <https://www.sciencedirect.com/science/article/abs/pii/S0167404821000730>
- <https://www.geeksforgeeks.org/trust-management-in-distributed-systems/>
- https://sandbox.engineering/?gad_source=1&gclid=CjwKCAiA34S7BhAtEiwACZzv4YZVGV9OPHbaM6niT1Yuoz2PW1THX2QZyLR4xSuGjJ8Fixk0mrhu8hoC3uwQAvD_BwE
- <https://community.fs.com/blog/edge-computing-iot.html>
- <https://www.askpython.com/python/examples/rsa-algorithm-in-python>
- <https://medium.com/coinmonks/rsa-encryption-and-decryption-with-pythons-pycryptodome-library-94f28a6a1816>
- <https://stackoverflow.com/questions/75926691/how-to-implement-sse-server-sent-events-server-and-client-in-python>
- <https://www.statology.org/sst-ssr-sse-in-python/>
- <https://bytewax.io/blog/data-stream-server-sent-events>
- <https://www.sciencedirect.com/science/article/pii/S2542660524001902https://www.tandfonline.com/doi/full/10.1080/00051144.2024.2314906#d1e166>