**ROMÂNIA**

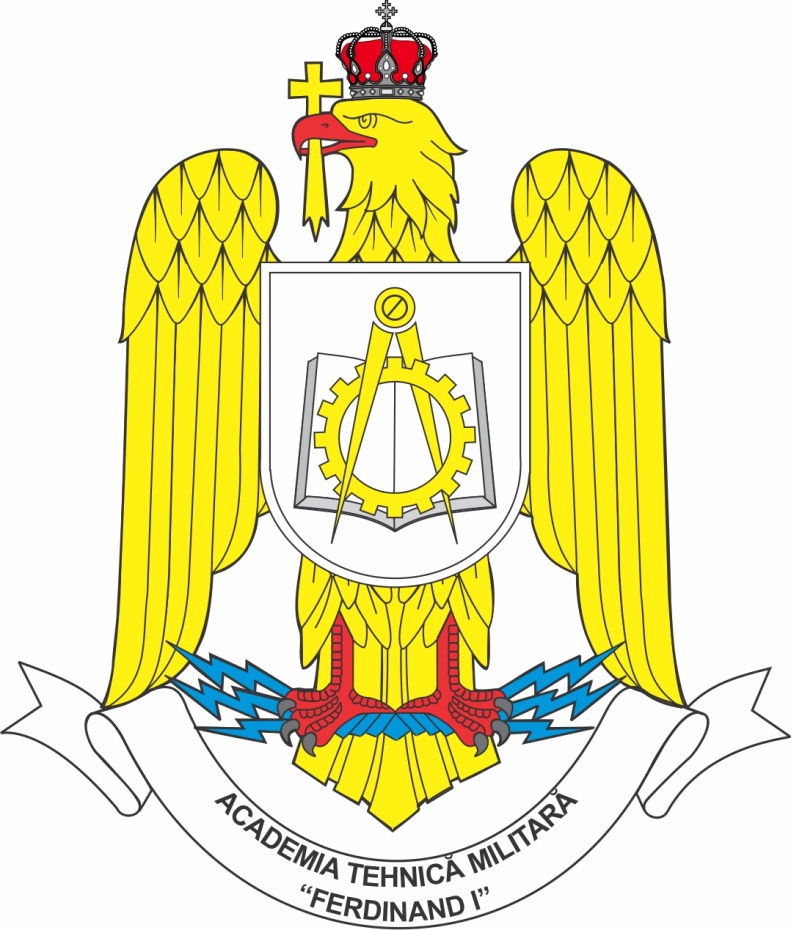
**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**FACULTATEA DE SISTEME INFORMATICE ȘI SECURITATE CIBERNETICĂ**

***Specializarea: Calculatoare și sisteme informatice pentru apărare***

***și securitate națională***



**Servicii pentru validarea atestărilor electronice de atribute**

Coordonator științific:

**Cpt. Conf. Univ. Dr. Ing. Iulian Aciobăniței**

ABSOLVENT:

**Sd. Sg. Maj. Dragoș-Mihai Crihană**

**BUCUREȘTI**

**2025**

# Abstract

In today’s fast-paced digital world, electronic attestations of attributes within digital wallets play a key role in enabling secure and trustworthy digital interactions. This project introduces a mobile application and backend system designed to validate electronic attestations stored and presented by digital wallets, fully aligned with the eIDAS 2 framework for interoperable digital identity services across the European Union.

The solution implements robust verification methods to ensure credential authenticity, integrity, and revocation status. It combines two mechanisms: standardized status lists from issuers and a cascade Bloom filter system, retrieved primarily via blob storage for performance, with automatic fallback to IPFS. References for both methods are securely stored on the blockchain via smart contracts, ensuring integrity and tamper resistance.

While credential validation is extremely simple from a user perspective, scanning a QR code and confirming data sharing, developing a secure, privacy-preserving validation system is highly complex. It requires validation of credential format, issuer authenticity, expiration date, and revocation status using decentralized methods. Moreover, implementing such a system requires mastery of multiple advanced standards, including OpenID for Verifiable Presentations (OpenID4VP), Verifiable Credentials Data Model 2.0 (VCDM 2.0), Revocation List 2020 (RL2020), and the Architecture and Reference Framework (ARF) defining European trust and interoperability architectures.

By integrating all these standards into a single, ready-to-use framework, the project delivers a scalable and efficient solution that significantly reduces implementation effort for developers and organizations aiming to adopt secure digital identity verification services. It demonstrates how a standards-compliant backend and a user-friendly mobile verifier app can provide seamless and trustworthy electronic attestation validation.

# Rezumat

În contextul dinamic al lumii digitale de astăzi, atestările electronice ale atributelor stocate in portofele digitale joacă un rol esențial în facilitarea unor interacțiuni sigure și de încredere. Acest proiect prezintă o aplicație mobile și un sistem backend conceput pentru a valida atestările electronice stocate și prezentate prin portofele digitale, aliniate complet cu cadrul eIDAS 2 pentru servicii interoperabile de identitate digitală la nivelul Uniunii Europene.

Soluția implementează metode robuste de verificare pentru a asigura autenticitatea, integritatea și starea de revocare a credențialelor. Ea combină două mecanisme: listele standardizate de status publicate de emitenți și un sistem de filtre Bloom în cascadă, recuperat în principal prin blob storage pentru performanță, cu fallback automat pe IPFS dacă este necesar. Referințele pentru ambele metode sunt stocate securizat pe blockchain prin smart contracts, asigurând integritatea datelor și rezistența la modificări neautorizate.

Deși, din perspectiva utilizatorului, validarea unui credențial este un proces extrem de simplu, scanarea unui cod QR și partajarea datelor necesare, dezvoltarea unui astfel de sistem sigur și care să respecte confidențialitatea este complexă. Este nevoie de validarea formatului credențialului, a autenticității emitentului, a datei de expirare și a statusului de revocare, folosind metode descentralizate. Implementarea unui astfel de sistem presupune totodată cunoașterea și integrarea unor standarde avansate precum OpenID for Verifiable Presentations (OpenID4VP), Verifiable Credentials Data Model 2.0 (VCDM 2.0), Revocation List 2020 (RL2020) și Architecture and Reference Framework (ARF), care definesc arhitectura europeană de încredere și interoperabilitate.

Prin integrarea acestor standarde într-un cadru unitar și gata de utilizare, proiectul oferă o soluție scalabilă și eficientă care reduce semnificativ efortul necesar dezvoltatorilor și organizațiilor ce doresc să adopte servicii sigure de verificare a identității digitale. Lucrarea demonstrează cum un backend compatibil cu standardele actuale și o aplicație mobile intuitivă pot oferi o validare fluidă și de încredere a atestărilor electronice.

**Cuprins**

[**Abstract** 1](#_Toc202564214)

[**Rezumat** 3](#_Toc202564215)

[**Listă de abrevieri** 7](#_Toc202564216)

[**Listă de figuri** 9](#_Toc202564217)

[**Listă de tabele** 11](#_Toc202564218)

[**1.** **Introducere** 13](#_Toc202564219)

[1.1 Importanța temei 13](#_Toc202564220)

[1.2 Obiectivele lucrării 14](#_Toc202564221)

[1.3 Metodologia de cercetare 14](#_Toc202564222)

[1.4 Rezultatele obținute 15](#_Toc202564223)

[1.5 Rezumatul lucrării pe capitole 16](#_Toc202564224)

[**2.** **Atestările electronice** 17](#_Toc202564225)

[2.1 Introducere 17](#_Toc202564226)

[2.2 Cazuri de utilizare 19](#_Toc202564227)

[2.3 Problemele abordărilor PKI clasice 20](#_Toc202564228)

[2.4 Avantaje ale atestărilor electronice 22](#_Toc202564229)

[2.5 Legislație 24](#_Toc202564230)

[2.6 Standarde 26](#_Toc202564231)

[2.7 Studiu comparativ privind cele mai populare platforme disponibile 29](#_Toc202564232)

[**3.** **Arhitectura serviciilor pentru atestări electronice** 31](#_Toc202564233)

[3.1 Principalele componente 31](#_Toc202564234)

[3.2 Autoritatea de emitere a atestărilor electronice 32](#_Toc202564235)

[3.3 Furnizorii de servicii 33](#_Toc202564236)

[3.4 Registrul pentru starea atestărilor electronice 34](#_Toc202564237)

[3.5 Listele de încredere 36](#_Toc202564238)

[3.6 EUDI Wallet 38](#_Toc202564239)

[3.7 Implementări existente 41](#_Toc202564240)

[3.7.1 Talao 41](#_Toc202564241)

[3.7.2 Walt.id 42](#_Toc202564242)

[3.7.3 Dock Wallet 44](#_Toc202564243)

[3.7.4 Trinsic 45](#_Toc202564244)

[3.7.5 ZealiD Wallet 47](#_Toc202564245)

[**4. Descrierea implementării** 49](#_Toc202564246)

[4.1 Definirea cerințelor 49](#_Toc202564247)

[4.2 Arhitectura soluției 53](#_Toc202564248)

[4.3 Descrierea modulelor componente 55](#_Toc202564249)

[4.3.1 Frontend 55](#_Toc202564250)

[4.3.2 Backend 59](#_Toc202564251)

[4.4 Tehnologii folosite pentru implementare 64](#_Toc202564252)

[4.4.1 Java și Spring Boot 64](#_Toc202564253)

[4.4.2 PostgreSQL 66](#_Toc202564254)

[4.4.3 Keycloak 67](#_Toc202564255)

[4.4.4 Unleash 68](#_Toc202564256)

[4.4.5 Docker 69](#_Toc202564257)

[4.4.6 Cloudflare 71](#_Toc202564258)

[4.5 Rezultatele testelor efectuate și interpretarea acestora 72](#_Toc202564259)

[4.5.1 Testarea unitară 73](#_Toc202564260)

[4.5.2 Testarea de integrare 78](#_Toc202564261)

[4.5.3 Testare de performanță 79](#_Toc202564262)

[4.5.4 Testare de acceptanță 81](#_Toc202564263)

[4.5.5 Raport de testare 82](#_Toc202564264)

[**5. Concluzii** 87](#_Toc202564265)

[5.1 Sinteza principalelor idei din lucrare 87](#_Toc202564266)

[5.2 Direcții pentru continuarea cercetării 88](#_Toc202564267)

[**6. Bibliografie** 89](#_Toc202564268)

# Listă de abrevieri

* eIDAS *Electronic Identification, Authentication and Trust Services*
* IPFS *Interplanetary File System*
* QR *Quick Response*
* SD-JWT *JSON Web Token*
* SD-JWT *Selective Disclosure JSON Web Token*
* TLS *Transport Layer Security*
* HTTPS *Hypertext Transfer Protocol Secure*
* EUDI *European Digital Identity*
* OpenID4VP *OpenID for Verifiable Presentations*
* EAA *Electronic Attestation of Attributes*
* QEAA *Qualified Electronic Attestation of Attributes*
* Pub-EAA *Public Electronic Attestation of Attributes*
* QTSP *Qualified Trust Service Providers*
* PKI *Public Key Infrastructure*
* CA *Certificate Authority*
* CRL *Certificate Revocation Lists*
* OCSP *Online Certificate Status Protocol*
* RSA *Rivest–Shamir–Adleman*
* ECC *Elliptic Curve Cryptography*
* SSI *Self Sovereign Identity*
* GDPR *General Data Protection Regulation*
* VC *Verifiable Credentials*
* OpenID4VCI *OpenID for Verifiable Credential Issuance*
* JSON *JavaScript Object Notation*
* XML *Extensible Markup Language*
* API *Application Programming Interface*
* DID *Decentralized Identifier*
* REST *Representational State Transfer*
* JSON-LD *JavaScript Object Notation for Linked Data*
* SDK *Software Development Kit*

# Listă de figuri

[Figura 1 Aplicabilitatea EUDI Wallet în conformitate cu eIDAS 2.0 24](#_Toc202555310)

[Figura 2 Modelul Issuer–Holder–Verifier în arhitectura Verifiable Credentials 26](#_Toc202555311)

[Figura 3 Structura unui SD-JWT 27](#_Toc202555312)

[Figura 4 Fluxul standard de utilizare a unui Verifiable Presentation 28](#_Toc202555313)

[Figura 5 Arhitectura serviciilor pentru atestări electronice 32](#_Toc202555314)

[Figura 6 Exemplu de listă de revocare în format bitstring 35](#_Toc202555315)

[Figura 7 Exemplu de înregistrare JSON pentru un PID Provider 37](#_Toc202555316)

[Figura 8 Ecranul principal al aplicației EUDI Wallet 38](#_Toc202555317)

[Figura 9 Lista de credențiale electronice stocate în portofel 39](#_Toc202555318)

[Figura 10 Vizualizarea detaliată a unui credential PID (SD-JWT) 40](#_Toc202555319)

[Figura 11 Interfața portofelului Talao 42](#_Toc202555320)

[Figura 12 Fluxul Walt.id 43](#_Toc202555321)

[Figura 13 Arhitectura Dock Wallet 45](#_Toc202555322)

[Figura 14 Interfața soluției Trinsic 46](#_Toc202555323)

[Figura 15 Interfața portofelului ZealiD 48](#_Toc202555324)

[Figura 16 Arhitectura sistemului 53](#_Toc202555325)

[Figura 17 Pagina principală „Verify” a aplicației CredCheck 55](#_Toc202555326)

[Figura 18 Feedback vizual pentru un credential validat cu succes 56](#_Toc202555327)

[Figura 19 Feedback vizual pentru un credential invalid sau respins 56](#_Toc202555328)

[Figura 20 Pagina „History” a aplicației CredCheck 57](#_Toc202555329)

[Figura 21 Pagina „Settings” a aplicației CredCheck 58](#_Toc202555330)

[Figura 22 Fluxul de interacțiune dintre CredCheck, Keycloak, EUDI Wallet și backend 59](#_Toc202555331)

[Figura 23 Fluxul de validare a credențialelor în backend 64](#_Toc202555332)

[Figura 24 Arhitectura fluxului unei aplicații Spring Boot 65](#_Toc202555333)

[Figura 25 Arhitectura PostgreSQL 66](#_Toc202555334)

[Figura 26 Fluxul ,,Resource Owner Password" în Keycloak 68](#_Toc202555335)

[Figura 27 Arhitectura Unleash 69](#_Toc202555336)

[Figura 28 Arhitectura Docker 70](#_Toc202555337)

[Figura 29 Tunel Cloudflare 72](#_Toc202555338)

# Listă de tabele

[Tabelul 1 Analiză comparativă a platformelor utilizate pentru prezentarea, emiterea și verificarea atestărilor electronice 31](#_Toc202564382)

[Tabelul 2 Testele unitare 78](#_Toc202564383)

[Tabelul 3 Testete de integrare 80](#_Toc202564384)

[Tabelul 4 Testele de performanță 82](#_Toc202564385)

[Tabelul 5 Testele de acceptanță 83](#_Toc202564386)

[Tabelul 6 Rezultatele testelor unitare 85](#_Toc202564387)

[Tabelul 7 Rezultatele testelor de integrare 86](#_Toc202564388)

[Tabelul 8 Rezultatele testelor de performanță 86](#_Toc202564389)

[Tabelul 9 Rezultatele testelor de acceptanță 87](#_Toc202564390)

# Introducere

## Importanța temei

Pe măsură ce tot mai multe activități se desfășoară în mediul digital, devine esențial ca utilizatorii să își poată dovedi identitatea într-un mod sigur, rapid și verificabil. Fie că este vorba despre accesarea unor servicii publice, înscrierea într-o instituție educațională sau încheierea unui contract electronic, este nevoie de un mecanism standardizat prin care identitatea și anumite atribute personale să poată fi prezentate în mod controlat, fără riscul de falsificare sau abuz.

Tranziția de la documentele fizice către cele digitale a adus numeroase beneficii, precum automatizarea proceselor, creșterea eficienței și o accesibilitate mai ridicată a serviciilor. Totuși, această schimbare a generat și provocări importante legate de încredere, securitate și interoperabilitatea dintre platforme. În acest context, Uniunea Europeană, prin inițiativa European Digital Identity (EUDI) și noul regulament eIDAS 2.0, propune un cadru unificat pentru identitatea digitală. Acest cadru permite fiecărui cetățean să dețină și să prezinte atestări electronice, cum ar fi numele, vârsta, cetățenia sau statutul educațional, folosind un portofel digital securizat, recunoscut la nivel european[1][2].

Pentru ca acest ecosistem să funcționeze corect, este esențială existența unor mecanisme de verificare automatizată și de încredere, prin care entitățile care solicită aceste atestări să poată valida autenticitatea, integritatea și starea lor (valid/revocat/expirat). Aici intervine importanța lucrării de față, care propune dezvoltarea unui serviciu de validare a atestărilor electronice în conformitate cu aceste standarde. Soluția urmărește să fie sigură, scalabilă și ușor de integrat în infrastructuri existente, fără a compromite experiența utilizatorului sau cerințele de securitate.

Pe măsură ce serviciile digitale devin tot mai prezente în viața de zi cu zi, nevoia de a verifica rapid și corect atestările electronice este tot mai evidentă. Lucrarea de față vine în întâmpinarea acestei nevoi, oferind o soluție practică pentru validarea acestor atestări într-un mod sigur, automatizat și adaptat cerințelor actuale de interoperabilitate și protecție a datelor.

## Obiectivele lucrării

Fiind în contextul dezvoltării unei soluții pentru verificarea atestărilor electronice, lucrarea de față își propune realizarea unui sistem care poate valida în mod automat credențiale digitale prezentate de utilizatori prin intermediul unui portofel digital. Scopul principal este acela de a oferi o metodă sigură, rapidă și conformă cu cerințele actuale de identitate digitală.

Aplicația urmărește să faciliteze procesul de validare a unor atribute personale precum numele, vârsta sau statutul educațional, fără a compromite confidențialitatea datelor. Un alt obiectiv este integrarea acestui proces într-un scenariu practic, care poate fi extins și adaptat în contexte reale, atât în sectorul public, cât și în cel privat.

Totodată, lucrarea își propune explorarea unor metode eficiente de verificare a autenticității și valabilității atestărilor, luând în considerare și nevoia de interoperabilitate în cadrul unei identități digitale europene.

## Metodologia de cercetare

Metodologia de cercetare a avut la bază o etapă de documentare extinsă asupra conceptelor cheie din domeniul identității digitale. Au fost studiate elemente precum regulamentul eIDAS 2.0, structura Verifiable Credentials, formatul SD-JWT și specificațiile asociate fluxurilor OpenID for Verifiable Presentations (OpenID4VP), toate analizate în contextul inițiativei European Digital Identity (EUDI).

În paralel cu documentarea teoretică, a fost investigat comportamentul portofelului digital EUDI Wallet[3], prin sesiuni de testare și debugging. Pentru a înțelege fluxul de prezentare al unui credential și cerințele asociate unui verifier, s-au analizat în detaliu cererile și răspunsurile generate de aplicație. Această etapă a ajutat la înțelegerea modului în care trebuie construite și conectate componentele pentru ca sistemul să funcționeze corect.

Pe baza acestor observații, a urmat etapa de definire a arhitecturii aplicației, stabilind rolul fiecărei componente în cadrul fluxului de validare. Implementarea a fost abordată treptat, începând cu funcționalitățile de bază și continuând cu integrarea componentelor necesare pentru verificarea credențialelor, generarea codurilor QR și comunicarea cu portofelul.

Procesul de cercetare a fost influențat și de stadiul incipient al unor standarde, cum este cazul status list-ului, integrat în lucrare chiar dacă a fost adoptat oficial destul de târziu. În plus, majoritatea implementărilor existente au fost dezvoltate în limbajul Kotlin, necunoscut inițial, ceea ce a făcut adaptarea mai dificilă, având în vedere că soluția propusă a fost construită în Java.

Întregul proces de dezvoltare a fost abordat pas cu pas, prin testare continuă și ajustări bazate pe comportamentul observat în fiecare etapă de integrare. Soluția finală a rezultat din aplicarea teoriei studiate și a observațiilor practice din timpul dezvoltării.

## Rezultatele obținute

Rezultatul obținut în urma procesului de cercetare, documentare și dezvoltare este un sistem funcțional de validare a atestărilor electronice, conceput să fie compatibil cu standardele promovate prin eIDAS 2.0[1] și inițiativa European Digital Identity[2]. Soluția permite inițierea fluxului de prezentare a credențialelor printr-o aplicație mobile și validarea automată printr-un serviciu backend dedicat.

Aplicația este construită să funcționeze cu formatul SD-JWT[4] și suportă verificarea stării credențialelor prin două mecanisme complementare: status list și filtrul Bloom în cascadă, a cărui referință este publicată pe blockchain. Sistemul asigură integrarea cu portofelul EUDI Wallet, facilitând interacțiunea dintre utilizator și serviciul de verificare. Printre funcționalitățile realizate se numără:

* Generarea unui QR code pentru inițierea prezentării (OpenID4VP)
* Arhitectură adaptabilă la evoluția specificațiilor EUDI și eIDAS 2.0
* Suport pentru validarea mai multor tipuri de atestări electronice
* Validarea credențialelor SD-JWT, inclusiv semnătură, sd\_hash și expirare
* Verificarea stării prin status list și filtru Bloom
* Integrare cu EUDI Wallet pentru partajarea credențialelor
* Selectarea atributelor care vor fi prezentate în procesul de validare
* Configurarea comportamentului aplicației prin sistem de feature flags
* Gestionarea autentificării și autorizării utilizatorilor prin Keycloak
* Expunerea serviciului prin HTTPS, cu protecție oferită de Cloudflare.

## Rezumatul lucrării pe capitole

Capitolele lucrării surprind parcursul complet al dezvoltării unei soluții pentru validarea atestărilor electronice, de la justificarea alegerii temei și prezentarea cadrului teoretic, până la detalierea implementării și interpretarea rezultatelor obținute. Structura este concepută astfel încât să reflecte atât componentele tehnice implicate, cât și contextul legislativ și standardele de interoperabilitate care fundamentează soluția propusă.

Primul capitol oferă contextul general al temei, evidențiind importanța validării atestărilor electronice în actualul ecosistem digital. Sunt prezentate obiectivele urmărite, metodologia aplicată în dezvoltarea soluției și rezultatele obținute în urma procesului de cercetare.

În al doilea capitol sunt abordate noțiunile fundamentale privind atestările electronice. Se discută atât avantajele și provocările acestor mecanisme în raport cu soluțiile clasice PKI, cât și aspectele legislative și standardele relevante. Este inclus și un studiu comparativ între platformele existente.

Al treilea capitol este dedicat descrierii arhitecturii serviciilor implicate în procesul de emitere, gestionare și verificare a atestărilor electronice. Sunt analizate componentele principale, cum ar fi autoritățile emitente, furnizorii de servicii, registrele de stare și portofelul digital EUDI Wallet, alături de exemple de implementări existente.

În cel de-al patrulea capitol este detaliată soluția implementată. Sunt definite cerințele funcționale, este prezentată arhitectura generală și sunt descrise modulele componente. Tot aici sunt evidențiate tehnologiile utilizate și sunt interpretate rezultatele testelor efectuate.

Capitolul al cincilea sintetizează concluziile desprinse din elaborarea lucrării și propune direcții de dezvoltare ulterioară care pot consolida sau extinde funcționalitatea soluției propuse.

Ultimul capitol conține bibliografia utilizată în procesul de documentare, reunind surse teoretice, materiale tehnice și resurse oficiale relevante pentru tematica abordată.

# Atestările electronice

## Introducere

Atestările electronice de atribute (Electronic Attestation of Attributes – EAA) sunt documente digitale care confirmă acuratețea unui anumit atribut, cum ar fi adresa de domiciliu sau o calificare profesională. Spre deosebire de atributul în sine, care este doar o simplă informație, EAA-ul are statutul unui document oficial, emis de un furnizor autorizat. Acest lucru garantează autenticitatea informației și oferă atestării aceeași valoare juridică precum un document tipărit. O atestare electronică poate fi adăugată, stocată și prezentată prin intermediul portofelului digital *EUDI Wallet*, oferind un mod standardizat și securizat de a demonstra atributele personale în mediul digital[5][6].

Atestările electronice reprezintă un element esențial în infrastructura identității digitale europene propusă de eIDAS 2.0. Ele permit cetățenilor să-și dovedească anumite caracteristici personale precum vârsta, adresa sau statutul profesional într-un mod sigur, verificabil și interoperabil în întreaga Uniune Europeană.

Conform cadrului propus de eIDAS 2.0, există trei tipuri principale de atestări electronice de atribute:

1. **Electronic Attestation of Attributes (EAA) – pentru utilizări de zi cu zi**

Atestările electronice de atribute (EAA) sunt cea mai simplă formă de credențiale digitale prevăzute de eIDAS 2. Ele sunt emise de organizații publice sau private și confirmă informații specifice despre o persoană sau entitate. Fiind folosite în contexte nereglementate, aceste atestări nu trebuie să respecte cerințe legale stricte, precum cele asociate documentelor oficiale[6].

EAA-urile sunt utile în situații de zi cu zi, unde este necesară o modalitate practică de a demonstra anumite atribute, cum ar fi calitatea de membru, participarea la un eveniment sau statutul de student. De exemplu, pot lua forma unor bilete de acces, carduri de fidelitate, legitimații de angajat sau abonamente la diverse servicii.

Sunt ușor de emis, eficiente din punct de vedere al costurilor și oferă o soluție simplă pentru gestionarea interacțiunilor de zi cu zi între organizații și utilizatori. Cu toate că nu sunt recunoscute în afara mediului în care sunt emise, ele rămân extrem de utile pentru aplicații interne[6].

1. **Qualified Electronic Attestation of Attributes (QEAA)**

Atestările calificate de atribute (QEAA) sunt credențiale digitale cu un nivel ridicat de încredere, prevăzute de regulamentul eIDAS 2. Acestea sunt emise exclusiv de *Qualified Trust Service Providers* (QTSP) și sunt supuse unor cerințe stricte privind securitatea, auditarea și conformitatea legală[6].

*QEAA*-urile sunt utilizate în contexte reglementate, unde este esențială validarea legală și securizată a informațiilor, cum ar fi serviciile publice, domeniul medical, financiar sau juridic. Pot atesta, de exemplu, o diplomă universitară, o licență profesională sau un certificat de stare civilă[6].

Aceste atestări au aceeași valoare juridică precum documentele tipărite și sunt protejate prin mecanisme criptografice avansate, precum sigiliile electronice. Datorită standardelor ridicate de încredere, QEAAs sunt acceptate în toate statele membre ale UE, fiind esențiale în schimburile digitale oficiale[6].

1. **Public Electronic Attestation of Attributes (PuB-EAA)**

Atestările publice de atribute (PuB-EAA) sunt emise în numele instituțiilor publice sau al autorităților oficiale, atunci când datele provin din registre sau baze de date guvernamentale. Acestea au aceeași valoare juridică precum documentele fizice și sunt folosite în special în servicii publice, cum ar fi emiterea certificatelor de naștere sau a permiselor de conducere[5].

Furnizorii PuB-EAA nu sunt considerați furnizori calificați de servicii de încredere (QTSP), dar trebuie să dețină un certificat calificat emis de un QTSP pentru a putea semna aceste atestări. În acest mod, se asigură că atestările respectă cerințele legale aplicabile documentelor oficiale, în special pentru verificarea unor atribute precum vârsta, adresa, cetățenia sau calificările profesionale[5].

## Cazuri de utilizare

Atestările electronice de atribute permit cetățenilor să interacționeze în mod securizat, standardizat și eficient cu servicii publice și private. Acestea sunt deja testate în mai multe scenarii concrete, acoperind domenii precum guvernare digitală, sănătate, educație sau transport. Mai jos sunt prezentate principalele categorii de utilizare definite în cadrul proiectului european:

1. **Permis de conducere -** Portofelul digital permite stocarea și prezentarea permisului de conducere în format electronic. Utilizatorul îl poate folosi la controlul în trafic, la închirierea de autovehicule sau în alte situații unde este necesară dovedirea dreptului de a conduce.
2. **Accesarea serviciilor guvernamentale -** Cetățenii pot accesa servicii administrative online, precum actualizarea datelor personale, solicitarea de documente oficiale sau înregistrarea la vot, utilizând atestări verificate din portofelul digital.
3. **Plăți** - Atestările pot fi folosite pentru autorizarea plăților, verificarea identității la tranzacții sensibile sau pentru a demonstra eligibilitatea în cazul reducerilor/subvențiilor. De asemenea, asigură autentificarea în doi pași în sistemele financiare.
4. **Educație** - Instituțiile pot emite diplome, certificate de studii sau atestări privind statutul de student. Acestea pot fi utilizate în procesul de admitere, la angajare sau pentru mobilitate academică în cadrul UE.
5. **Sănătate** - Portofelul digital poate conține prescripții electronice, cardul european de asigurări sociale de sănătate sau alte documente medicale.
6. **Călătorii** - Check-in-ul la aeroport, rezervările de bilete sau identificarea la hotel pot fi efectuate direct din EUDI Wallet, eliminând nevoia de documente tipărite.
7. **Telecomunicații** - Înrolarea într-un serviciu de telefonie mobilă, portarea unui număr sau semnarea unui abonament pot fi realizate exclusiv online.
8. **Semnătură elecronică** - Utilizatorii pot aplica semnături electronice calificate pe documente digitale.
9. **Identități organizaționale** - Angajații pot primi legitimații digitale cu acces securizat în spații fizice sau în sistemele informatice ale instituției[7].

## Problemele abordărilor PKI clasice

Sistemele clasice de identitate digitală, bazate pe *Public Key Infrastructure* (PKI) și certificate X.509, au fost esențiale în securizarea comunicațiilor și autentificarea utilizatorilor în ultimele decenii. Ele funcționează prin emiterea de certificate digitale semnate de autorități de certificare, care garantează legătura dintre o cheie publică și identitatea unei entități.

Totuși, odată cu apariția unor nevoi noi, cum ar fi controlul direct al utilizatorului asupra datelor proprii, verificarea selectivă a atributelor și eliminarea punctelor unice de încredere, abordările PKI clasice au început să devină insuficiente. Limitările lor nu mai răspund cerințelor moderne privind identitatea digitală descentralizată, interoperabilă și orientată pe confidențialitate, așa cum sunt definite în inițiative precum EUDI Wallet sau standarde precum Verifiable Credentials și SD-JWT.

PKI-ul are mai multe limitări care încep să se vadă tot mai clar atunci când încercăm să îl folosim în scenarii moderne de identitate digitală. Printre cele mai importante probleme se numără:

1. **Single point of failure**

PKI funcționează pe baza încrederii în autorități de certificare (CA), care au rolul de a valida identitatea unei entități și de a semna certificatele digitale. Problema este că întregul sistem depinde de integritatea acestor CA-uri. Dacă una singură este compromisă, atacatorii pot emite certificate aparent valide pentru orice domeniu, ceea ce compromite securitatea comunicațiilor la scară largă. Cazuri precum DigiNotar sau Comodo au arătat exact acest risc: în urma unor atacuri, au fost generate certificate false care au fost apoi folosite în atacuri de tip man-in-the-middle. Astfel de situații arată cât de periculos este să ai un model în care toată încrederea este concentrată într-un singur punct[8][13][14].

1. **Gestionarea complexă a lanțului de încredere**

În PKI, verificarea unui certificat nu înseamnă doar validarea lui individuală, ci presupune parcurgerea unui întreg lanț de încredere. Acest lanț pornește de la certificatul utilizatorului, trece prin una sau mai multe autorități intermediare și ajunge în final la o autoritate root, recunoscută de sistemul de operare sau de browser.

Această structură introduce o complexitate semnificativă. Dacă o verigă din lanț este configurată greșit sau este compromisă, întregul mecanism de încredere poate fi afectat. Un exemplu clar este cazul CNNIC din 2015, când această autoritate root din China a emis un certificat intermediar către un partener numit MCS Holdings. Partenerul a folosit acel certificat într-un proxy HTTPS care permitea interceptarea traficului. Ca urmare, Google și Mozilla au revocat încrederea în CNNIC și au eliminat-o din listele de CA-uri de încredere[9].

1. **Revocare greoaie și ineficientă**

În PKI, mecanismele clasice de revocare se bazează pe CRL (Certificate Revocation Lists) și OCSP (Online Certificate Status Protocol). CRL-urile sunt liste statice care pot ajunge rapid la dimensiuni mari, ceea ce le face greu de distribuit și ineficiente în sisteme care au nevoie de verificări rapide.

OCSP a fost introdus pentru a rezolva aceste probleme, permițând verificarea în timp real a unui certificat direct de la autoritatea care l-a emis. Totuși, în practică, OCSP vine cu propriile limitări. În primul rând, implică o întârziere vizibilă la stabilirea conexiunii, întrucât browserul trebuie să aștepte răspunsul de la serverul OCSP. În al doilea rând, compromite confidențialitatea: la fiecare verificare, serverul OCSP poate vedea exact ce site accesează utilizatorul, construind astfel un istoric al comportamentului acestuia online[10][15].

1. **Criptografie învechită și pregătire slabă pentru viitor**

Una dintre slăbiciunile PKI-ului clasic este faptul că se bazează pe algoritmi criptografici care, în unele cazuri, nu mai sunt considerați siguri. De exemplu, SHA‑1 a fost folosit timp îndelungat pentru semnături digitale, dar în ultimii ani s-a dovedit vulnerabil la coliziuni, adică două fișiere complet diferite pot avea același hash. Acest lucru a fost demonstrat în 2017, când cercetători de la Google și CWI Amsterdam au generat prima coliziune reală de SHA‑1[11].

La fel, cheile RSA mai scurte, cum sunt cele de 1024 de biți, nu mai sunt recomandate pentru uz actual, iar pe termen lung chiar și RSA‑2048 va deveni insuficient. Motivul principal ține de progresul în domeniul calculatoarelor cuantice. Algoritmi precum RSA și ECC sunt vulnerabili la algoritmi cuantici speciali, cum ar fi algoritmul lui Shor, care ar putea sparge aceste metode mult mai rapid decât calculatoarele clasice[12][14].

1. **Lipsa controlului utilizatorului**

În PKI, identitatea digitală este emisă și controlată de autorități centralizate. Utilizatorul nu poate decide ce informații sunt incluse într-un certificat și nici cui sau în ce condiții acestea sunt prezentate. Certificatele sunt rigide, iar toate datele conținute devin vizibile în momentul utilizării, indiferent dacă sunt relevante sau nu pentru acel context.

Acest model intră în contradicție cu principiile de *self-sovereign identity* (SSI), care susțin că identitatea digitală ar trebui să aparțină utilizatorului, nu unei entități externe. Într-un sistem SSI, persoana își gestionează propriile credentiale și poate alege ce date să dezvăluie, când și cui. De asemenea, principiul *privacy by design* presupune ca sistemele să fie construite astfel încât să minimizeze colectarea și expunerea de date personale încă din faza de proiectare, lucru pe care PKI-ul clasic nu îl oferă[16].

## Avantaje ale atestărilor electronice

Utilizarea atestărilor electronice de atribute vine cu o serie de avantaje importante, atât pentru cetățeni, cât și pentru instituții sau companii. Aceste beneficii se datorează în primul rând faptului că vorbim despre documente digitale care pot fi verificate automat, sunt recunoscute la nivel european și sunt integrate într-o infrastructură sigură, cum este cea oferită de EUDI Wallet și de cadrul legal eIDAS.

Câteva dintre avantajele principale oferite de aceste atestări sunt prezentate mai jos:

1. **Creșterea eficienței operaționale**

Prin eliminarea documentelor fizice, a copiilor după acte și a verificărilor manuale, atestările electronice reduc semnificativ timpii de procesare în activități precum verificarea identității pentru acces la servicii, înscrierea la o universitate, autentificarea în platforme educaționale sau validarea statutului de student. Pentru organizații, acest lucru înseamnă automatizare, mai puține erori și economii importante de resurse. Organizațiile pot astfel să economisească timp, să reducă volumul de muncă repetitivă și să minimizeze riscurile de eroare[2][17]..

1. **Securitate sporită**

Atestările electronice sunt semnate digital de furnizori autorizați sau calificați (QTSP) și pot fi verificate criptografic în timp real. Acest mecanism asigură că informațiile nu au fost modificate și că provin dintr-o sursă de încredere. Astfel, atestările devin rezistente la falsificare și oferă un nivel ridicat de încredere în procesul de validare a identității sau a atributelor unei persoane[2][17]..

1. **Interoperabilitate la nivel european**

EUDI Wallet este construit pe baza regulamentului eIDAS 2.0, ceea ce asigură recunoașterea și acceptarea atestărilor electronice în toate statele membre. Potrivit Comisiei Europene, această interoperabilitate este esențială pentru ca cetățenii să poată folosi aceleași date de identitate în orice stat membru[2][17]..

1. **Control deplin asupra datelor**

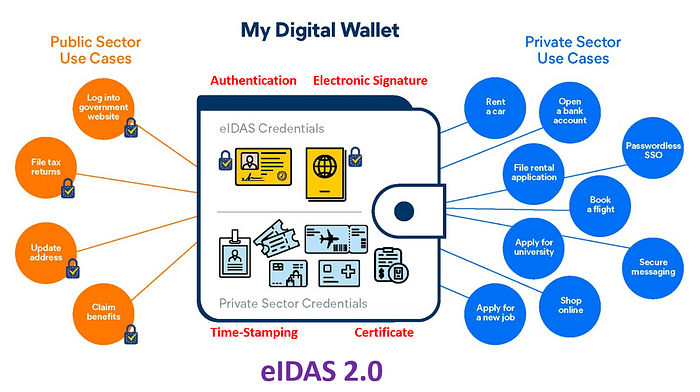
Atestările electronice sunt păstrate direct în portofelul digital al utilizatorului, nu într-o bază de date centralizată. Acest lucru îi oferă control total asupra modului în care își folosește datele: poate alege ce informații să partajeze, cui le transmite și în ce scop. Modelul respectă principiile *privacy by design* și *self-sovereign identity*, în sensul că utilizatorul nu trebuie să dezvăluie mai mult decât este necesar. Astfel, se evită expunerea inutilă a datelor personale și se menține un control real asupra confidențialității[2][17].

1. **Scalabilitate și adaptabilitate tehnologică**

Atestările electronice sunt gândite să se integreze ușor în aplicații digitale actuale, fie că vorbim de platforme publice, fie de soluții din mediul privat. Ele pot susține procese digitale în întregime, de la verificarea identității, până la utilizarea semnăturilor electronice și gestionarea accesului la servicii. Pentru că respectă standarde deschise și sunt compatibile cu diverse sisteme, aceste atestări pot fi adaptate ușor în funcție de nevoile unei organizații sau de schimbările legislative. Această flexibilitate le permite să evolueze odată cu tehnologia și să răspundă cerințelor tot mai diverse ale instituțiilor și utilizatorilor[2][17].

## Legislație

eIDAS 2.0[1] este o actualizare a regulamentului european privind identificarea electronică și serviciile de încredere (eIDAS), adoptat inițial în 2014. Noua versiune, publicată oficial în 2024, adaugă noi funcționalități importante și propune o abordare comună la nivelul Uniunii Europene pentru gestionarea identității digitale. Principalul său obiectiv este să ofere cetățenilor, companiilor și instituțiilor un mod sigur și simplu prin care oamenii își pot dovedi identitatea și folosi datele personale oriunde în UE.

Elementul central introdus de eIDAS 2.0 este European Digital Identity Wallet (EUDI Wallet)[2], un portofel digital standardizat, care va permite cetățenilor UE să stocheze și să utilizeze diverse tipuri de credențiale, cum ar fi datele de identitate (PID), diplome, statut profesional sau alte atestări electronice de atribute. Acest portofel va putea fi folosit atât în relația cu instituțiile publice, cât și cu furnizori de servicii private, fiind recunoscut automat în toate statele membre.

[[1]](#footnote-1)

Figura 1 Aplicabilitatea EUDI Wallet în conformitate cu eIDAS 2.0

Spre deosebire de versiunea inițială a regulamentului, care era orientată în principal pe semnăturile electronice și identificarea online, eIDAS 2.0[1] vine cu o viziune mult mai largă: construirea unei infrastructuri digitale europene, care să permită utilizatorilor să acceseze servicii din orice stat membru cu aceleași date și nivel de încredere ca în propria țară.

Portofelul digital EUDI va trebui să fie pus la dispoziție gratuit de către fiecare stat membru până cel târziu în 2026. Utilizatorii vor avea control complet asupra datelor lor, putând alege ce informații partajează, cu cine și în ce context. Acest principiu reflectă alinierea eIDAS 2.0 la cerințele GDPR și la conceptele de *privacy by design* și *self-sovereign identity*. Mai exact, datele din portofel nu vor fi partajate automat, ci doar la inițiativa utilizatorului și doar în măsura în care sunt necesare pentru un anumit scop[16][18][19].

Totodată, regulamentul introduce cerințe stricte pentru furnizorii de servicii de încredere (QTSP), care pot emite atestări calificate și pot furniza servicii de semnătură electronică. Aceștia vor trebui să respecte standarde de securitate, interoperabilitate și auditabilitate, pentru a garanta încrederea în ecosistemul european[1].

eIDAS 2.0 stabilește și recunoașterea juridică automată a documentelor și credentialelor digitale emise într-un stat membru, ceea ce înseamnă că o diplomă digitală eliberată într-o țară UE poate fi folosită direct în altă țară fără traduceri, legalizări sau verificări suplimentare[1].

Deși inițiativa este primită pozitiv la nivel european, există și opinii critice, mai ales în ceea ce privește riscul de centralizare a datelor și posibilitatea ca portofelul digital să devină un instrument de urmărire sau de colectare excesivă a datelor. Aceste preocupări sunt contrabalansate prin mecanismele de control oferite utilizatorilor și prin obligațiile impuse furnizorilor de identitate și atestări.

eIDAS 2.0 încearcă să facă identitatea digitală mai ușor de folosit, mai sigură și mai clară pentru toată lumea. Cu ajutorul portofelului EUDI, oamenii vor putea să-și dovedească identitatea și alte informații personale într-un mod simplu, fără să piardă controlul asupra datelor lor[2].

## Standarde

Pentru a asigura interoperabilitate, securitate și transparență în sistemele digitale, implementarea eIDAS 2.0 se sprijină pe o serie de standarde internaționale. Acestea fac posibilă emiterea, stocarea și validarea atestărilor electronice în mod descentralizat, oferind utilizatorilor control real asupra datelor lor[1][2].

Unul dintre cele mai importante standarde este *Verifiable Credentials*(VC), dezvoltat de W3C[20]. Acesta definește un format standardizat pentru credențiale digitale, documente care pot conține informații precum identitatea unei persoane, statutul de student, diploma de absolvire sau orice alt atribut verificabil. Credențialele sunt semnate digital de către un emitent și stocate într-un portofel digital, de unde pot fi prezentate unui verificator fără a fi nevoie ca acesta să contacteze emitentul.

Această arhitectură se bazează pe trei roluri esențiale:

* Issuer – entitatea care emite și semnează digital credentialul (ex: o universitate, o instituție guvernamentală)
* Holder – persoana sau organizația care primește și stochează credentialul într-un portofel digital
* Verifier – entitatea care solicită și verifică validitatea credentialului (ex: un angajator, o instituție publică sau o platformă digitală)

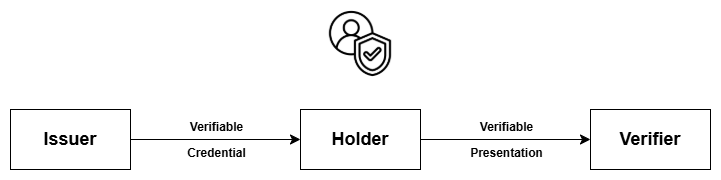
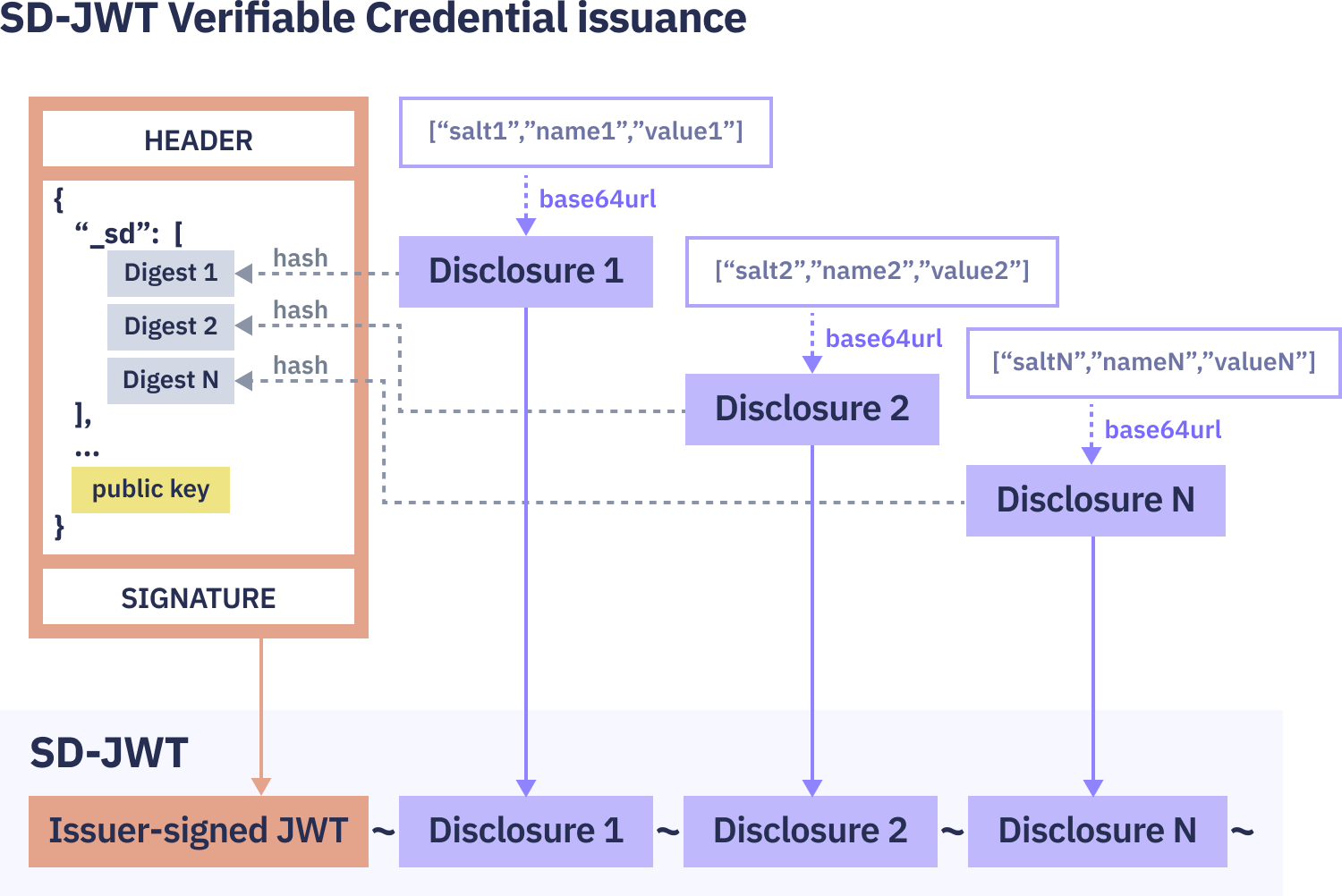


Figura 2 Modelul Issuer–Holder–Verifier în arhitectura Verifiable Credentials

Un alt standard important utilizat în lucrarea de față este SD-JWT (Selective Disclosure JSON Web Token). Acesta extinde formatul clasic JWT și permite ca doar anumite informații să fie partajate dintr-un credențial, păstrând restul datelor confidențiale. De exemplu, un utilizator poate dovedi că are peste 18 ani fără să dezvăluie data completă a nașterii. Tehnic, acest lucru se realizează prin înlocuirea valorilor vizibile cu hash-uri criptografice semnate de emitent, în timp ce valorile reale (numite disclosure-uri) sunt păstrate separat și pot fi prezentate doar la nevoie.

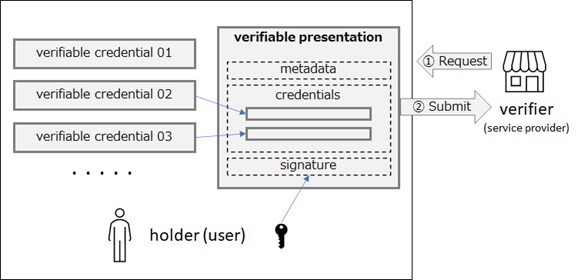
La momentul verificării, portofelul digital trimite prezentarea compusă din JWT-ul semnat și doar disclosure-urile relevante. Verificatorul poate valida integritatea acestora prin recalcularea hash-urilor, fără a avea acces la restul informațiilor din credential. Acest mecanism este compatibil cu infrastructurile existente bazate pe JWT și a fost adoptat în cadrul inițiativelor europene, inclusiv în prototipurile EUDI Wallet. În felul acesta, SD-JWT oferă un control mai bun asupra datelor personale și sprijină aplicarea principiilor din GDPR și eIDAS 2.0 legate de confidențialitate și limitarea informațiilor partajate[2][4].

[[2]](#footnote-2)

Figura 3 Structura unui SD-JWT

Pentru prezentarea credențialelor dintr-un portofel digital către un *verifier*, se utilizează protocolul *OpenID for Verifiable Presentations* (OpenID4VP)[21]. Acesta extinde standardul *OpenID Connect* și introduce posibilitatea ca o aplicație să trimită o cerere, numită presentation definition, prin care specifică tipul și structura credentialelor dorite. Răspunsul la această cerere constă într-un *Verifiable Presentation* (VP), un document digital semnat care conține unul sau mai multe credențiale și care poate fi validat criptografic de către verificator.

*Verifiable Presentation*[21] este formatul standard prin care un utilizator își poate dovedi atributele digitale într-un mod controlat, selectiv și verificabil. Acest format poate încorpora mecanisme precum SD-JWT VP, care permit dezvăluirea doar a anumitor informații din credentiale, păstrând restul datelor confidențiale. Procesul se desfășoară într-un cadru securizat, în care utilizatorul are control deplin asupra datelor pe care le partajează, iar verificatorul poate valida autenticitatea și integritatea prezentării fără a contacta emitentul.



[[3]](#footnote-3)

Figura 4 Fluxul standard de utilizare a unui Verifiable Presentation

## Studiu comparativ privind cele mai populare platforme disponibile

Atunci când analizăm platformele disponibile pentru gestionarea atestărilor electronice, este important să ținem cont de diferențele semnificative dintre ele în modul în care pot fi utilizate și implementate, funcționalitățile suportate și gradul de conformitate cu standardele europene, precum eIDAS 2.0. Fiecare platformă este dezvoltată pentru un anumit tip de utilizare, de la aplicații de tip portofel digital pentru utilizatori finali, până la soluții complete pentru emiterea și verificarea credentialelor digitale.

De asemenea, unele platforme pun accentul pe simplitate și integrare rapidă, în timp ce altele oferă capabilități extinse și flexibilitate tehnică, fiind dedicate echipelor de dezvoltare sau infrastructurilor publice. Aceste diferențe sunt sintetizate în tabelul următor, care oferă o privire de ansamblu asupra celor mai reprezentative soluții disponibile în prezent:

|  |  |  |  |
| --- | --- | --- | --- |
| **Platformă** | **Tip acces** | **Avantaje** | **Dezavantaje** |
| **EUDI Wallet** | Demo public / gratuit | * Suportă Verifiable Presentations conform specificațiilor eIDAS 2.0 * Asigură interoperabilitate perfectă între statele membre UE și protejează datele utilizatorilor * Ideal ca portofel digital pentru cetățeni, oferind funcționalitate de stocare și prezentare simplă | * Nu permite emiterea sau verificarea credențialelor, doar stocare și prezentare * Funcționalitatea este limitată în versiunea demo * Nu oferă API sau SDK pentru integrare programatică, rol strict de wallet pentru utilizator |
| **walt.id** | Open-source / Enterprise | * Suport complet pentru emiterea și verificarea VC-urilor în formate W3C, SD-JWT și mDoc * Suportă OID4VCI, OID4VP, key-binding, revocare și QR/URL * Administrează chei DID și X.509 prin SDK-uri puternice | * Interfața principală e realizată prin SDK/API, necesită timp și resurse pentru implementare * Complexitatea tehnică ridicată poate reprezenta o provocare pentru echipe mici sau proiecte POC |
| **Talao Wallet** | Open-source | * Emite și verifică Verifiable Credentials, inclusiv formate SD-JWT, prin OID4VCI * Open-source, compatibil EBSI/EUDI, cu încorporare a mecanismelor de disclosure selectiv * Permite integrare rapidă în fluxuri cross-device, QR & URL, plus demo-uri publice pentru utilizatori | * Lipsesc componente grafice și interfețe pentru utilizator final, majoritatea fluxurilor sunt prin CLI/API * Necesită cunoștințe tehnice pentru integrare și scalare |
| **Dock Wallet** | Open-source / comercial | * Permite emiterea și verificarea completă a VC printr-un API și portal no‑code accesibil dezvoltatorilor și organizațiilor * Suport integrat pentru biometrie și backup în cloud * Este compatibil cu ecosisteme EBSI/EUDI și standardul OpenID4VP | * Documentația este uneori incompletă și poate necesita clarificări tehnice suplimentare pentru configurări complexe * Platforma se bazează pe blockchain-ul proprietar Dock |
| **Trinsic** | Comercial (SaaS) | * Oferă un portal no-code * Suportă emitere și verificare de VC cu OpenID4VP și posibilitatea de integrare prin API/SDK * Beneficiază de securitate enterprise | * Model comercial (SaaS) cu costuri lunare, fără opțiune open-source * Nu include mecanisme de selective disclosure (SD-JWT) ieftin și nativ |
| **ZealiD Wallet** | Comercial | * Oferă semnături electronice calificate, conform eIDAS, compatibile cu platforme precum Adobe * Permite stocarea și prezentarea de VC | * Nu oferă emitere directă de credențiale digitale standard prin API * Nu are suport pentru SD‑JWT sau OpenID4VP * Model comercial cu abonament |

Tabelul 1 Analiză comparativă a platformelor utilizate pentru prezentarea, emiterea și verificarea atestărilor electronice

Prin urmare, alegerea unei platforme pentru atestări electronice ține de scopul aplicației, complexitatea integrării și compatibilitatea cu standardele relevante.

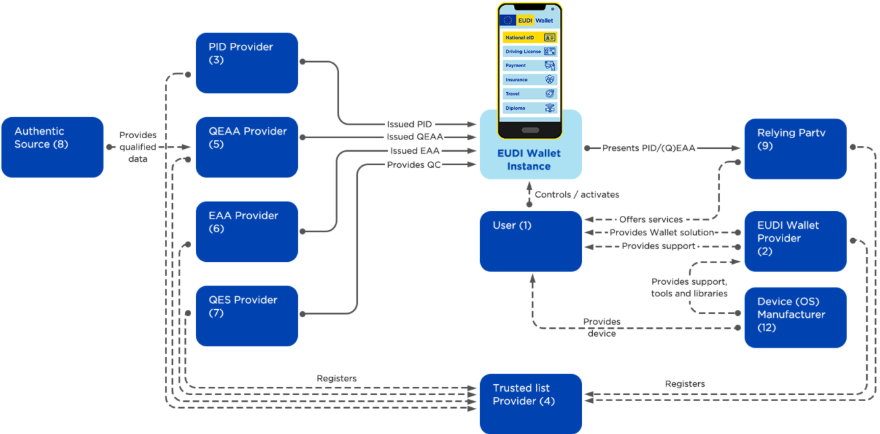
# Arhitectura serviciilor pentru atestări electronice

## Principalele componente

Pentru ca un sistem de atestări electronice să fie funcțional și de încredere, este necesară existența unui set de componente care colaborează într-un mod bine structurat. Fiecare dintre aceste elemente are un rol bine definit în procesul de emitere, stocare, prezentare și validare a credentialelor. Împreună, ele formează o arhitectură construită pe fundamente de securitate, protecția datelor și compatibilitate între sisteme, în conformitate cu cerințele cadrului legal eIDAS 2.0 și cu standardele tehnice internaționale[1][2][[22].

În această secțiune sunt prezentate, pe scurt, componentele principale care stau la baza unui astfel de sistem.

1. **Autoritatea emitentă -** entitatea responsabilă cu generarea și semnarea digitală a credentialelor. Aceasta validează datele inițiale despre utilizator și le încorporează într-o structură standardizată, semnată criptografic, care poate fi verificată ulterior de alte părți
2. **Furnizorul de servicii -** entitatea care solicită prezentarea unui credential și realizează verificările necesare privind autenticitatea, integritatea și validitatea acestuia. În funcție de rezultat, stabilește acordarea accesului la un serviciu sau resursă
3. **Portofelul digital –** aplicația prin care utilizatorul își gestionează credentialele.
4. **Deținătorul credențialelor -** utilizatorul final care primește credentialele de la emitent și le prezintă către diferiți verificatori
5. **Trust Framework -** ansamblul de reguli, politici și cerințe tehnice care stabilesc condițiile de funcționare ale ecosistemului
6. **Registrul de încredere -** listă oficială care conține informații despre emitenții, verificatorii și autoritățile recunoscute din sistem
7. **Mecanismele de status –** soluții care permit verificatorilor să determine dacă un credential este încă valabil
8. **Stratul de interoperabilitate -** componenta care cuprinde protocoalele și formatele standardizate ce asigură compatibilitatea între platforme[22]



[[4]](#footnote-4)

Figura 5 Arhitectura serviciilor pentru atestări electronice

## Autoritatea de emitere a atestărilor electronice

Autoritatea de emitere, denumită și „issuer”, este entitatea care se ocupă cu generarea și semnarea digitală a credentialelor electronice. Aceasta are responsabilitatea de a verifica datele despre utilizator și de a construi pe baza acestora un credential digital, semnat criptografic. Semnătura oferă garanția că informațiile sunt autentice și pot fi ulterior verificate de alte entități din sistem. Emitentul poate fi o instituție publică, cum ar fi un furnizor de identitate sau o universitate, sau o organizație privată, dacă este autorizată conform legislației, prin regulamentul eIDAS 2.0[22].

Procesul de emitere respectă standarde tehnice precum *OpenID4VCI*[23] și *SD-JWT*[4]. După validarea informațiilor, credentialul este semnat digital și transmis portofelului digital al utilizatorului, printr-un flux securizat și bine definit. Acest schimb are loc prin API-uri standardizate, care asigură interoperabilitatea între diferitele componente ale sistemului.

Un aspect important este faptul că emitentul nu are vizibilitate asupra utilizării ulterioare a credentialului. Acest principiu, numit *unlinkability*, este esențial pentru respectarea vieții private, fiind aliniat atât la cerințele eIDAS, cât și la cele prevăzute de GDPR[1].

Pe lângă emiterea propriu-zisă, autoritatea de emitere are și rolul de a furniza informații privind starea credentialelor. Acestea pot fi puse la dispoziție prin liste de status[24] sau prin mecanisme optimizate, cum ar fi filtrele Bloom[25], care permit verificarea eficientă a validității unui credential.

## Furnizorii de servicii

Furnizorii de servicii, denumiți adesea *verifiers*, reprezintă entitățile care solicită utilizatorilor să prezinte unul sau mai multe credentiale electronice, în scopul validării anumitor informații. Aceste entități pot fi organizații publice sau private care oferă acces la un serviciu sau resursă condiționat de furnizarea unei dovezi, cum ar fi identitatea unei persoane, vârsta, statutul educațional sau calitatea de membru al unei instituții[20][22].

În arhitectura sistemelor bazate pe *verifiable credentials*, rolul verificatorului este de a defini ce informații dorește să obțină, de a primi prezentarea din partea portofelului utilizatorului și de a valida conținutul acesteia. Validarea presupune mai multe etape:

* verificarea semnăturii digitale
* verificarea structurii și formatului datelor
* consultarea *status list-urilor*
* conslutarea listelor de încredere
* analiza informațiilor primite pentru a verifica dacă acestea corespund cerințelor impuse

Pentru a iniția o prezentare, furnizorul de servicii definește o *presentation definition*, un set de reguli tehnice care descriu tipul de credențiale acceptate și atributele care trebuie prezentate. Această definiție este apoi pusă la dispoziția portofelului utilizatorului printr-un cod QR sau un link, de obicei în contextul unui protocol standardizat precum OpenID4VP[21].

Furnizorul nu are acces la mai multe date decât cele solicitate explicit, ceea ce face ca acest model să respecte principiile de minimizare a datelor și de confidențialitate. Prezentările sunt deseori generate prin mecanisme care permit dezvăluirea selectivă a informațiilor, cum este cazul formatului SD-JWT, oferind utilizatorului control asupra a ceea ce partajează.

Pentru a valida semnătura unui credential, furnizorul se bazează pe informațiile disponibile în registrul de încredere, care conține date despre emitenți recunoscuți și cheile publice asociate acestora. De asemenea, poate interoga un registru de stare, furnizat de emitent, pentru a verifica dacă credentialul a fost revocat[22][24].

În funcție de contextul de utilizare, furnizorul de servicii poate fi integrat într-o aplicație mobile, un portal web sau un sistem automatizat. Indiferent de implementare, acesta trebuie să respecte regulile definite de trust framework-ul ecosistemului din care face parte, pentru a asigura interoperabilitatea și încrederea în întregul proces.

## Registrul pentru starea atestărilor electronice

Registrul pentru starea atestărilor electronice este componenta care permite verificatorilor să determine dacă un credential este încă valabil sau dacă a fost revocat sau expirat. Această verificare este esențială pentru menținerea încrederii în sistem și pentru prevenirea utilizării unor credențiale care nu mai sunt de actualitate sau au fost compromise.

Responsabilitatea pentru publicarea și menținerea unui astfel de registru revine emitentului. În funcție de implementare, acest registru poate lua mai multe forme. Una dintre cele mai răspândite este *status list-ul*[24], un mecanism standardizat care folosește o structură de tip listă de biți (bitstring), unde fiecare poziție corespunde unui credențial emis, iar valoarea indică starea acestuia. Această abordare este eficientă din punct de vedere al spațiului și poate fi publicată ca un document static, accesibil oricărui verificator.

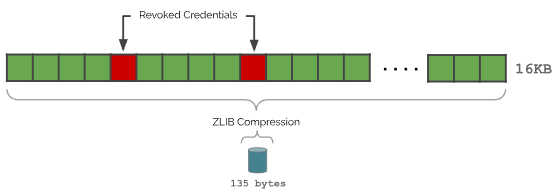
 [[5]](#footnote-5)

Figura 6 Exemplu de listă de revocare în format bitstring

O altă soluție este utilizarea filtrelor Bloom în cascadă, o extensie a filtrelor Bloom clasice, care permite reprezentarea compactă a unui set de credențiale revocate, păstrând în același timp un control precis asupra ratelor de eroare. Acest mecanism presupune o structură pe niveluri, în care fiecare filtru este folosit pentru a corecta eventualele rezultate fals pozitive generate de nivelul anterior. Mai mult decât atât, această abordare permite protejarea confidențialității, prin adăugarea de elemente fictive care maschează informații sensibile, precum numărul de credentiale emise sau revocate. Filtrele Bloom în cascadă sunt deosebit de utile în sisteme care gestionează volume mari de credentiale, unde este esențial să se obțină un echilibru între performanță, economie de spațiu și protejarea metadatelor[25].

În proiectarea acestor registre, este important să fie luat în considerare și aspectul legat de confidențialitate, în special proprietatea numită *unlinkability*. Acest principiu presupune că prezentarea unui credential nu ar trebui să poată fi corelată cu o altă prezentare a aceluiași utilizator. În anumite cazuri, modul în care este construit registrul (de exemplu, dacă pozițiile dintr-o status list sunt fixe și identificabile) poate duce la riscul de corelare între sesiuni diferite. De aceea, implementările moderne caută soluții care să reducă această expunere, astfel încât verificarea stării unui credențial să nu compromită anonimitatea sau controlul utilizatorului asupra propriei identități.

Indiferent de forma tehnică adoptată, registrul de stare trebuie să fie accesibil public, actualizat periodic și semnat digital, pentru a garanta integritatea informației. Verificatorul consultă acest registru în momentul validării unei prezentări, pentru a se asigura că credențialul prezentat este încă acceptat de emitent.

Gestionarea corectă a stării credentialelor este o cerință fundamentală pentru orice sistem de încredere. Indiferent de soluția aleasă, ceea ce contează este ca verificarea stării unui credențial să se integreze în procesul de validare, fără să introducă întârzieri sau riscuri suplimentare legate de securitate sau confidențialitate.

## Listele de încredere

Listele de încredere (Trust Lists) sunt componente esențiale în arhitectura ecosistemului EUDI, deoarece permit stabilirea și menținerea unui cadru de încredere între entitățile participante: emitenți, verificatori, furnizori de portofel, autorități de certificare și alți actori relevanți. Aceste liste oferă un mecanism standardizat prin care se comunică, în mod transparent și verificabil, care entități sunt recunoscute oficial și în ce condiții[26][27].

În cadrul EUDI Wallet Architecture & Reference Framework (ARF), listele de încredere sunt structurate și menținute la nivel național de către autoritățile competente, care sunt responsabile pentru înscrierea entităților relevante. Mai apoi, aceste liste pot fi notificate Comisiei Europene pentru recunoaștere între statele membre. Fiecare listă este semnată digital și accesibilă public, pentru a putea fi interogată automat de portofele și verificatori.

Fiecare înregistrare dintr-o listă de încredere conține, în general, informații despre identitatea entității, cheile publice asociate, tipul de servicii oferite, date despre înregistrare și validitate, precum și *endpoint-urile* pentru serviciile suportate. În plus, există și un câmp *status*, care indică starea actuală a entității în cadrul listei. Valori precum *granted*, *suspended* sau *revoked* permit portofelului sau verificatorului să stabilească dacă o entitate este autorizată să emită sau să verifice credentiale[26][27].

Ecosistemul EUDI include mai multe tipuri de liste distincte, fiecare având un scop bine definit: lista pentru furnizorii de portofel, pentru *PID Providers*, pentru autoritățile de emitere a atributelor calificate, pentru entitățile publice eligibile â sau pentru *Relying Parties* care solicită credențiale.

În prezent, conform documentației EUDI, înregistrările din listele de încredere nu sunt publicate într-un format unificat accesibil direct, dar arhitectura specifică modul în care aceste liste trebuie construite pentru a putea fi verificate automat. Fiecare entitate trebuie să fie descrisă într-un format *machine-readable*, precum JSON sau XML, iar lista trebuie semnată digital și să conțină metadate precum versiune, dată de emitere și entitate emitentă.

Un exemplu simplificat de structură pentru un PID Provider poate fi următoarea:

{

"id": "did:example:pid",

"type": "pid\_provider",

"status": "granted",

"name": "Registrul Național de Identitate",

"country": "RO",

"registered\_at": "2025-04-01T00:00:00Z",

"valid\_until": "2026-04-01T00:00:00Z",

"public\_keys": ["https://example.ro/keys/jwks.json"],

"service\_endpoints": {

"openid4vci": "https://example.ro/oid4vci",

"status": "https://example.ro/statuslist.json"

},

"policy\_uri": "https://example.ro/pid-policy"

}

Figura 7 Exemplu de înregistrare JSON pentru un PID Provider

Prin integrarea acestor liste în fluxurile de validare, portofelele și verificatorii pot determina automat dacă o semnătură digitală aparține unei entități autorizate. Astfel, listele de încredere contribuie direct la securitatea și interoperabilitatea întregului sistem, fără a adăuga complexitate pentru utilizatorul final.

## EUDI Wallet

EUDI Wallet[3] este aplicația digitală propusă la nivelul Uniunii Europene pentru a permite cetățenilor să își gestioneze și utilizeze identitatea electronică într-un mod standardizat, sigur și interoperabil. Conform cerințelor regulamentului eIDAS 2.0, portofelul este dezvoltat pentru a funcționa ca un instrument de încredere, prin care utilizatorii pot stoca, accesa și prezenta în mod controlat credentiale digitale emise de entități autorizate.

Aplicația este concepută cu o structură modulară, care combină o interfață ușor de utilizat cu o arhitectură bine structurată. În ecranul principal, utilizatorul are acces rapid la funcțiile principale oferite de portofel: autentificare, partajarea documentelor și semnătura electronică. Interfața este gândită astfel încât funcțiile esențiale să fie ușor de înțeles și folosit, fără să ceară cunoștințe tehnice din partea utilizatorului.

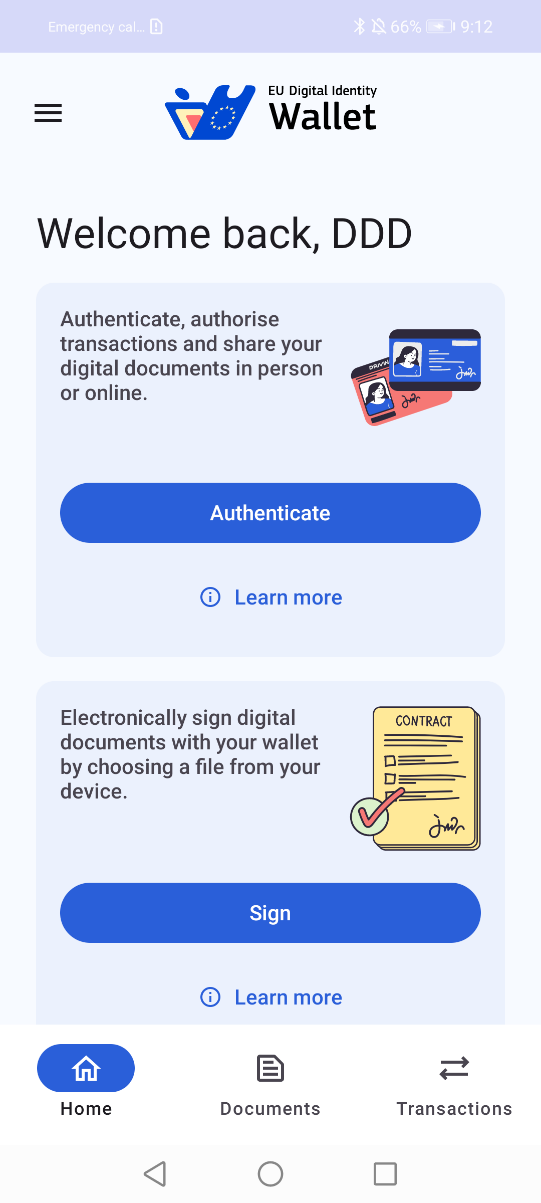


Figura 8 Ecranul principal al aplicației EUDI Wallet

Portofelul permite stocarea de credențiale în formate conforme cu standardele tehnice adoptate la nivel european. Unul dintre cele mai importante este SD-JWT, care oferă mecanisme avansate pentru dezvăluirea selectivă a informațiilor. Astfel, în loc ca un utilizator să prezinte întregul set de date dintr-un document, poate alege să partajeze doar informația necesară în context, cum ar fi faptul că este major, fără a furniza data completă a nașterii.

Credențialele sunt afișate într-o secțiune dedicată a aplicației. În captura următoare se observă două astfel de credentiale: unul pentru identificarea personală (PID), iar celălalt privind absolvirea unei instituții de învățământ.

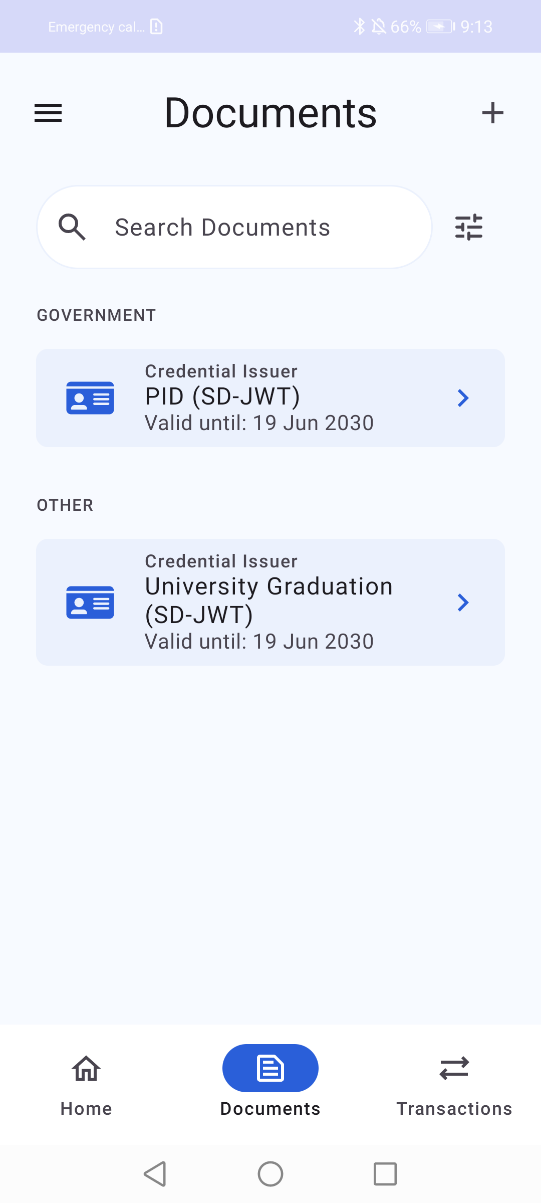


Figura 9 Lista de credențiale electronice stocate în portofel

Fiecare credențial poate fi explorat în detaliu, iar aplicația permite consultarea tuturor atributelor incluse în document. În figura următoare este afișat conținutul unui credențial PID, cu informații precum vârsta, data nașterii, numele și țara emitentă. Formatul este structurat și lizibil, pentru ca utilizatorul să înțeleagă exact ce informații deține și va partaja:

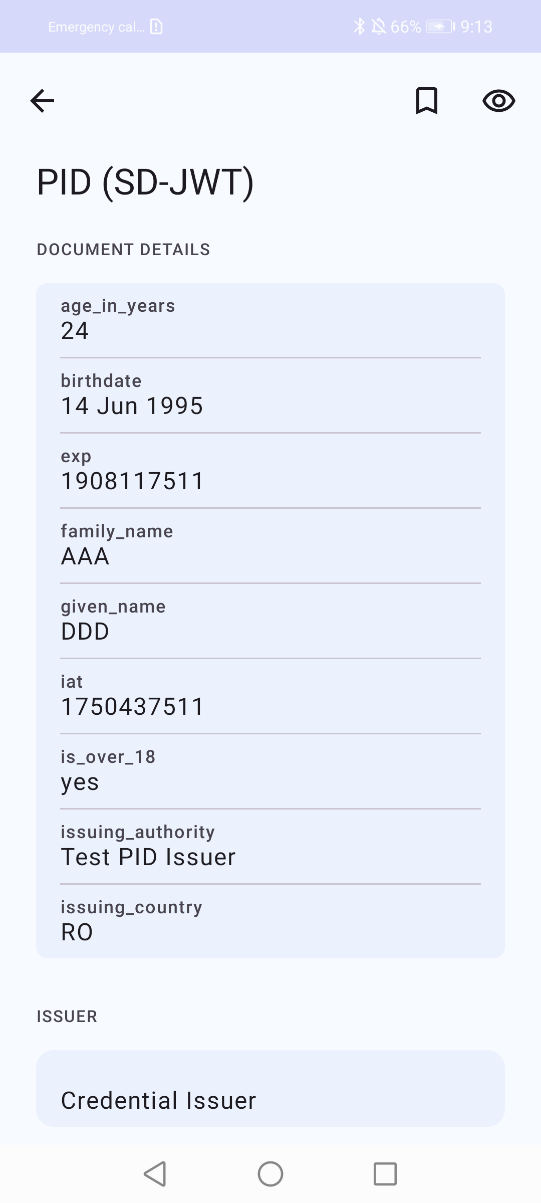


Figura 10 Vizualizarea detaliată a unui credential PID (SD-JWT)

Dincolo de vizualizare și prezentare, portofelul integrează și funcționalități avansate precum generarea de semnături electronice calificate. Acestea sunt conforme cu cerințele eIDAS și pot fi utilizate pentru semnarea de documente digitale într-un mod recunoscut legal. Semnătura este produsă prin utilizarea unei chei private protejate în interiorul unui element de securitate, păstrând cheia strict în interiorul dispozitivului. Acest mecanism asigură atât conformitatea cu legislația europeană, cât și o protecție sporită împotriva abuzurilor[2][3][22].

Aplicația funcționează pe baza unor protocoale standardizate, printre care se numără OpenID4VCI pentru obținerea credentialelor, OpenID4VP pentru prezentarea acestora și ISO/IEC 18013-5 pentru scenarii de utilizare în proximitate fizică. Aceste protocoale permit interoperabilitatea portofelului între toate sistemele conforme din Uniune, indiferent de statul în care este utilizat.

Pe tot parcursul funcționării sale, aplicația respectă principiile de confidențialitate și securitate prin design. Prezentările sunt construite astfel încât să fie imposibil de corelat între ele *unlinkability*, iar portofelul nu înregistrează centralizat istoricul utilizatorului. Toate aceste măsuri contribuie la păstrarea anonimității și la susținerea unui model sigur și transparent de identitate digitală europeană.

## Implementări existente

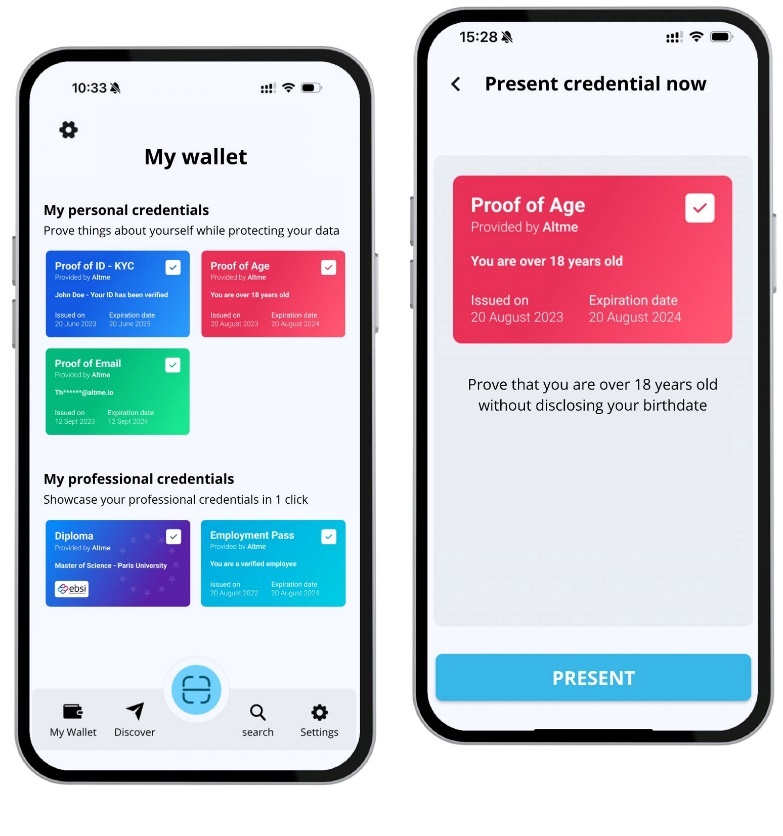
În ceea ce privește implementările existente, în prezent se remarcă mai multe soluții care oferă funcționalități de tip portofel digital, emitere și verificare a atestărilor electronice.

### Talao

Talao[28] este o soluție open-source completă axată pe identitate digitală descentralizată (SSI), care oferă portofel, emitere și verificare într-o singură suită interoperabilă. Apărută inițial în spațiul blockchain, Talao a evoluat către o platformă compatibilă EUDI, conformă cu cerințele eIDAS 2.0, ARF și EBSI‑V3.

În privința componentelor, Talao are un portofel mobile disponibil pentru Android și iOS, programat în Dart (Flutter), cu suport pentru stocare de VC și prezentare selectivă de date. Emiterea este realizată printr-un server *issuer*, capabil să genereze credențiale PID, de vârstă sau alte atribute, cu notificări către portofel utilizând OpenID4VCI. Verificarea se face printr-un serviciu *verifier*, care gestionează cereri OpenID4VP și validează prezentările primite .

Din punct de vedere tehnic, sistemul implementează protocoalele standard: OpenID for Verifiable Credential Issuance, OpenID for Verifiable Presentations și suport SD‑JWT, oferind și compatibilitate cu profile EBSI și DIIP. Componenta de status folosește *status list* (bitstring) pentru gestionarea revocărilor.

Avantajul distinct este flexibilitatea arhitecturală: componentele sunt separate și independente, dar pot fi folosite în combinație sau integrat într-un ecosistem mai larg. De exemplu, poți folosi doar *talao-issuer* și integra alt *wallet* propriu, sau doar componenta *verifier*.

[[6]](#footnote-6)

Figura 11 Interfața portofelului Talao

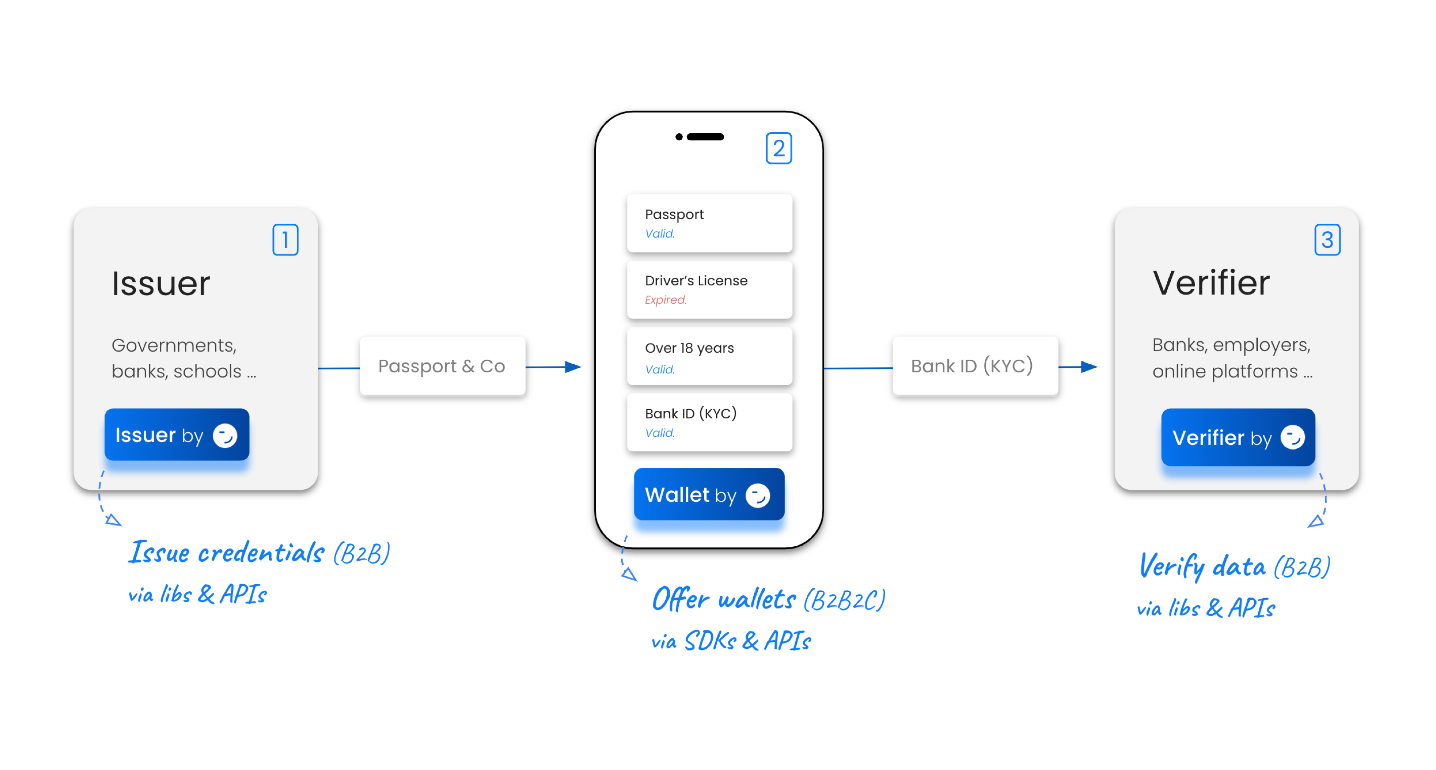
### Walt.id

Walt.id[29] este o platformă open-source de infrastructură pentru identitate digitală și portofel. Scopul său este de a oferi un set complet de instrumente care simplifică adoptarea arhitecturii de tip issuer‑holder‑verifier, respectând standarde cheie precum W3C Verifiable Credentials, OpenID4VCI, OpenID4VP, SD‑JWT și regulamente precum eIDAS 2 și GDPR.

Platforma include un SSI‑Kit, o suită de biblioteci și API‑uri care permit generarea de chei și DIDs, emiterea și verificarea credențialelor, și gestionarea portofelului de identitate. Acest kit este acum integrat în *Walt.id identity stack*, o soluție ce oferă interfețe REST, CLI și aplicații web pentru fiecare rol: *issuer*, *verifier* și *wallet*. Interfața *web‑wallet* permite stocarea, vizualizarea și prezentarea credențialelor, iar aplicațiile *issuer* și *verifier* facilitează emiterea prin OIDC4VCI și verificarea prin OIDC4VP/SIOPv2.

Fluxul începe cu generarea DID‑urilor prin API‑uri sau CLI, urmată de emiterea credentialelor în formate JSON‑LD sau JWT/SD‑JWT, stocarea acestora în portofel și prezentarea selectivă către verificatori. Totul este susținut de un sistem de gestionare a revocării prin token status list.

Un avantaj major al Walt.id este că oferă o documentație clară și bine structurată, ceea ce ajută la procesul de integrare și dezvoltare pentru echipele tehnice. Echipa oferă suport comunitar și commercial și consultanță pentru entități guvernamentale și enterprise .

Totuși, utilizarea Walt.id implică operarea și mentenanța infrastructurii backend, ceea ce poate necesita resurse DevOps și atenție la securitate. În funcție de nevoi, pot fi utilizate doar anumite componente ale platformei, fără a fi necesară integrarea întregului sistem.

[[7]](#footnote-7)

Figura 12 Fluxul Walt.id

### Dock Wallet

Dock[30] este o platformă de identitate descentralizată care oferă infrastructură completă pentru emiterea, stocarea și verificarea credențialelor digitale. Soluția include o rețea blockchain proprie (Dock Chain), dar și un portofel digital disponibil pentru Android și iOS, denumit Dock Wallet.

Dock Wallet este conceput să funcționeze atât pentru persoane fizice, cât și pentru organizații. Utilizatorul poate primi credențiale de la entități acreditate, le poate vizualiza în aplicație și le poate transmite ulterior către verificatori prin fluxuri de prezentare. Aplicația folosește formate de tip JWT și JSON-LD pentru credențiale, iar verificările se bazează pe semnăturile asociate și, acolo unde este necesar, pe statusul publicat de emitenți.

Pe lângă aplicația mobile, Dock pune la dispoziție și un SDK pentru emiterea și verificarea credențialelor, precum și o consolă web destinată organizațiilor care doresc să devină furnizori de atestări electronice.

Un avantaj al Dock este integrarea cu propria rețea blockchain, care permite înregistrarea metadatelor referitoare la credențiale și ancore criptografice pentru verificare. Această abordare oferă un strat suplimentar de încredere și integritate, fără a compromite confidențialitatea datelor utilizatorului. Portofelul nu stochează datele pe blockchain, ci doar semnături și identificatori anonimi, menținând astfel principiul de minimizare a datelor.

Platforma Dock este disponibilă atât sub formă open-source, cât și ca serviciu comercial. Soluția este în continuă dezvoltare, iar compatibilitatea cu standardele europene este în curs de consolidare.

Totuși, Dock nu este încă integrat oficial într-un cadru reglementat precum cel al portofelului european, dar oferă o bază solidă pentru proiecte care vizează adoptarea standardelor deschise și construirea unor sisteme de identitate descentralizată.

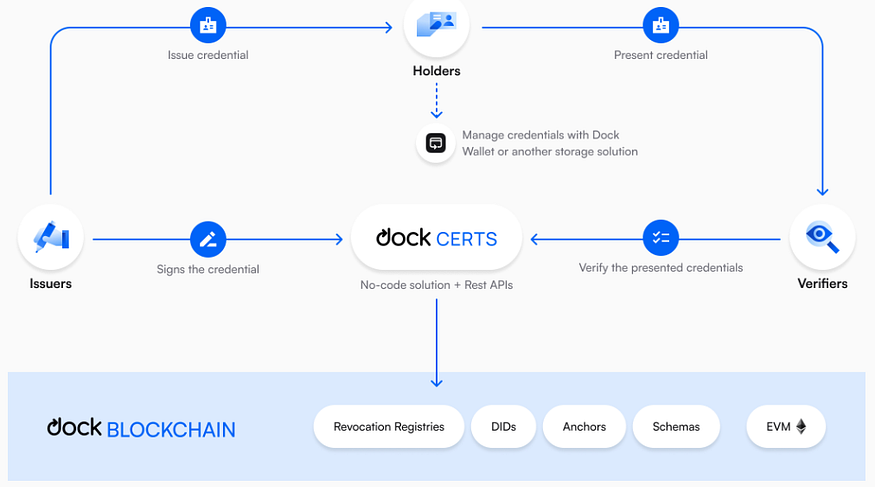
 [[8]](#footnote-8)

Figura 13 Arhitectura Dock Wallet

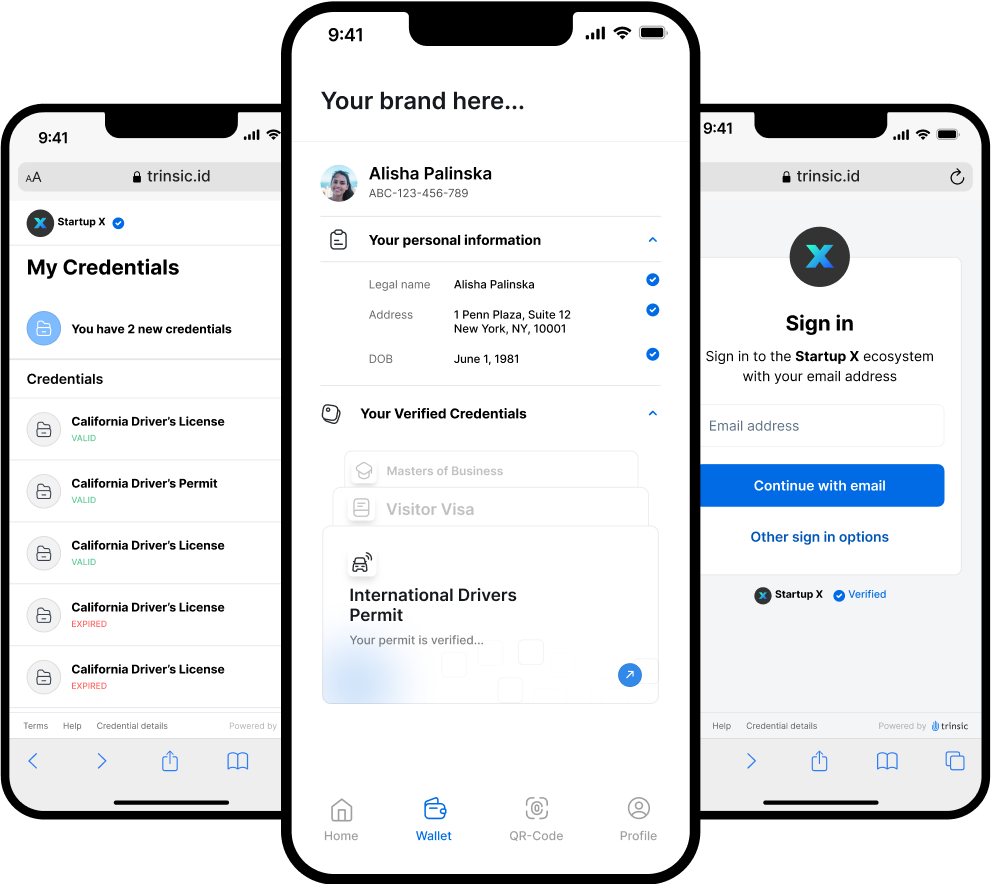
### Trinsic

Trinsic[31] este o platformă comercială dedicată construirii de aplicații și servicii bazate pe identitate digitală descentralizată, punând la dispoziția dezvoltatorilor o infrastructură completă pentru emiterea, stocarea și verificarea credențialelor. Soluția este oferită ca serviciu și include un SDK, API-uri REST și o aplicație portofel care poate fi personalizată sau integrată în aplicații existente.

Platforma este compatibilă cu standardele W3C pentru Verifiable Credentials și utilizează formate de tip JSON-LD. Trinsic sprijină fluxuri de prezentare care respectă principiile dezvăluirii selective. Aplicația este construită astfel încât utilizatorul să păstreze controlul complet asupra propriilor date, fără ca acestea să fie accesibile platformei.

Un element central al ecosistemului Trinsic este Trinsic Studio, o interfață web intuitivă prin care organizațiile pot crea, emite și gestiona credențiale digitale. Prin intermediul acestei console, se pot defini scheme de credențiale, se pot trimite invitații către utilizatori și se pot urmări răspunsurile la cererile de prezentare. Studio-ul este gândit pentru a reduce barierele tehnice și a permite adoptarea rapidă a identității descentralizate chiar și de către organizații fără experiență tehnică avansată.

Pe partea de integrare, Trinsic oferă SDK-uri pentru mai multe limbaje, precum și API-uri standardizate care permit crearea rapidă de fluxuri de autentificare, onboarding sau verificare. Portofelul furnizat de Trinsic poate fi utilizat ca aplicație de sine stătătoare sau poate fi încorporat în alte aplicații mobile prin SDK.

Limitarea principală a Trinsic este caracterul său comercial, deși oferă o versiune gratuită pentru testare, funcționalitățile avansate și scalarea sistemului necesită un plan plătit. De asemenea, pentru proiecte care urmăresc conformitate strictă cu standardele europene, este necesară o evaluare atentă a compatibilității și eventual integrarea unor componente externe.

[[9]](#footnote-9)

Figura 14 Interfața soluției Trinsic

### ZealiD Wallet

ZealiD[32] este o soluție de identitate digitală certificată la nivel european, care oferă un portofel mobil cu funcționalitate extinsă, inclusiv semnături electronice calificate și gestionarea unui certificat calificat eIDAS.

Portofelul este disponibil pentru Android și iOS, iar funcția centrală o constituie identificarea la distanță a utilizatorului și emiterea unui certificat digital calificat direct în aplicație. Procesul de înregistrare implică verificarea actului de identitate, selfie video cu detectare facială, verificarea numărului de telefon și a unui cont bancar asociat.

Soluția este concepută pentru a funcționa ca portofel de identitate digitală, oferind interacțiune cu servicii externe prin autentificare și semnătură, cu scopul de a înlocui identificarea clasică cu username și parolă în scenarii cu valoare legală ridicată.

Spre deosebire de alte portofele din zona *verifiable credentials*, ZealiD nu este orientată în primul rând către modelul *issuer–holder–verifier* din ecosistemul EUDI/SSI, ci mai degrabă către un model tradițional PKI,dar centrat pe utilizator. Totuși, poziția sa ca QTSP și integrarea cu sisteme de semnătură electronică o fac extrem de relevantă pentru viitoarele cerințe de identitate digitală în Europa, inclusiv în contextul eIDAS 2.0 și al viitorului EUDI Wallet oficial.

Un avantaj major al ZealiD este faptul că soluția este deja în producție, este recunoscută legal și funcționează în regim live pentru semnături calificate, fără intervenția fizică a operatorului uman în procesul de identificare. Limitarea, în schimb, este că ZealiD funcționează mai degrabă ca o aplicație verticală, cu o serie funcționalități fixe, și nu oferă aceeași flexibilitate pentru dezvoltatori precum platforme precum Walt.id sau Trinsic. De asemenea, nu există o infrastructură deschisă pentru *issuer/verifier*, ci mai degrabă o platformă închisă, certificată și gestionată integral de furnizor.



[[10]](#footnote-10)

Figura 15 Interfața portofelului ZealiD

# Descrierea implementării

## Definirea cerințelor

Implementarea acestui proiect are ca scop dezvoltarea unei soluții care să faciliteze validarea atestărilor electronice, în conformitate cu cerințele cadrului european eIDAS 2.0 și ale ecosistemului EUDI. Aplicația oferă suport pentru verificarea semnăturilor și stării credențialelor, precum și integrarea cu portofele digitale.

Scopul general este de a oferi un sistem sigur, scalabil și interoperabil, care să permită instituțiilor sau actorilor privați să verifice în mod automat, rapid și standardizat autenticitatea și validitatea unui credențial digital prezentat de un utilizator. Aplicația oferă o interfață intuitivă, un API accesibil și suport pentru standardele și formatele actuale de credentiale electronice, fiind adaptabilă pentru contexte diverse, precum educație, administrație publică sau domeniul juridic.

Punctele forte ale implementării sunt evidențiate prin satisfacerea unor cerințe esențiale, precum:

**Verificarea stării unei atestări electronice**

Una dintre funcționalitățile esențiale ale aplicației este posibilitatea de a verifica dacă o atestare electronică prezentată de un utilizator a fost revocată sau este încă activă. Pentru aceasta, sistemul consultă un registru de stare asociat cu emitentul credentialului. În cadrul implementării, această verificare se realizează prin două mecanisme complementare: un *status list*[24] de tip *bitstring* (conform specificațiilor VC Status List) și o structură avansată de filtre Bloom[25] organizate în cascadă, stocată sub formă de blob stocat în blockchain. Aplicația extrage din credențial informația necesară pentru a determina poziția corespunzătoare și evaluează starea acestuia în mod automat, fără intervenția utilizatorului.

**Verificarea veridicității unei atestări electronice prin intermediul listelor de încredere**

Pe lângă starea credențialului, sistemul trebuie să asigure și validitateasemnăturii sale și a emitentului. În acest sens, aplicația consultă o listă de încredere, ce conține entitățile autorizate să emită anumite tipuri de credențiale. Lista include, de regulă, metadate precum chei publice, adrese URL, tipuri de credențiale suportate și nivelul de certificare al fiecărei entități. Verificarea presupune extragerea identității emitentului din credențial, compararea acesteia cu informațiile din listă și validarea semnăturii folosind cheia publică corespunzătoare. Doar credențialele emise de entități recunoscute vor fi acceptate de aplicație.

**Verificarea criptografică a atestărilor electronice**

O componentă fundamentală a validării este verificarea criptografică a credențialelor primite din partea portofelului. După ce utilizatorul trimite credențialul, aplicația execută mai mulți pași de validare internă. Sistemul verifică semnătura digitală aplicată de emitent, apoi compară informațiile dezvăluite cu hash-urile stocate în structura credențialului (SD Hashes). Această verificare asigură că datele prezentate nu au fost modificate și că provin din credențialul original semnat. Procesul este automat, iar eventualele erori sunt raportate direct furnizorului de servicii.

**Interfață (API) ușor de utilizat furnizorilor de servicii**

Pentru ca aplicația să poată fi utilizată în contexte reale, de către furnizori de servicii care nu dispun neapărat de infrastructură tehnică proprie, este esențială existența unei interfețe API simplu de integrat. Aceasta oferă funcții precum trimiterea unei cereri de verificare, primirea unei prezentări, rularea validărilor interne și obținerea unui rezultat clar. API-ul este documentat, oferă răspunsuri structurate și este gândit astfel încât să poată fi integrat rapid în aplicații web sau mobile existente.

**Integrare cu EUDI Wallet**

Validarea atestărilor presupune existența unui portofel digital prin care utilizatorul poate transmite credențialele sale. Aplicația este compatibilă cu EUDI Wallet și folosește protocolul OpenID4VP pentru a iniția un proces de prezentare. La nivel practic, acest lucru presupune generarea unui cod QR care conține o cerere de prezentare, iar wallet-ul utilizatorului răspunde la această cerere, trimițând credențialul într-un format standardizat. Acest flux este gândit pentru a funcționa fără intervenții complicate din partea utilizatorului.

**Aplicație mobile pentru verificator**

Pentru a sprijini utilizarea practică a platformei, a fost dezvoltată o aplicație mobile destinată entităților care doresc să verifice atestări electronice. Aplicația joacă rolul unui verificator și poate fi utilizată, spre exemplu, de instituții sau operatori privați care trebuie să confirme anumite informații despre o persoană, cum ar fi vârsta minimă sau statutul de student.

Aplicația generează un cod QR corespunzător unei cereri de prezentare, interacționează cu EUDI Wallet pentru a primi credentialul și comunică ulterior cu backend-ul pentru efectuarea tuturor verificărilor necesare. În urma procesării, aplicația afișează rezultatul validării într-un format clar și ușor de înțeles, simulând experiența reală a unui verificator.

Aplicația este gândită să poată fi distribuită către terți, iar entitățile care doresc să o folosească vor primi un cont dedicat, configurat în funcție de context de propriul context.[22]

**Configurabilitate prin feature flags**

Aplicația mobile include un sistem de *feature flags* care permite adaptarea comportamentului în funcție de nevoile verificatorului, fără a modifica aplicația sau a relansa codul. Prin aceste mecanisme, se poate controla dinamic ce tipuri de informații sunt solicitate dintr-un credențial.

Acest sistem oferă flexibilitate ridicată și permite personalizarea cerințelor de validare pentru diferite scenarii. Activarea sau dezactivarea anumitor funcționalități se face din configurare, iar modificările au efect imediat, fără întreruperi în utilizare.

**Instalare și rulare în containere Docker**

Platforma este concepută astfel încât toate componentele sale să poată fi instalate și rulate în containere Docker. Această abordare simplifică procesul de instalare, oferă un mediu izolat pentru fiecare serviciu și permite reproducerea facilă a mediului de execuție, indiferent de sistemul pe care este implementată aplicația.

Folosirea containerizării asigură o gestionare mai ușoară a dependențelor și permite dezvoltatorilor sau operatorilor tehnici să pornească rapid întregul sistem printr-un set minim de comenzi, fără configurări complexe suplimentare.

**Conformitate cu standardele privind atributele electronice**

Credentialele procesate de aplicație conțin atribute electronice conforme cu formatele agreate de comunitatea internațională (W3C Verifiable Credentials) și cele aflate în curs de standardizare în cadrul ecosistemului EUDI. Aplicația este capabilă să interpreteze corect aceste atribute și să aplice reguli de validare asupra lor, de exemplu, verificarea vârstei, a calității de student sau a unei anumite identități juridice.

**Interfață grafică intuitivă**

Aplicația este însoțită de o interfață grafică concepută să fie clară și ușor de utilizat, indiferent de nivelul tehnic al utilizatorului. Navigarea prin aplicație este logică, iar acțiunile principale sunt accesibile în mod direct, fără a necesita cunoștințe avansate.

Elementele vizuale și mesajele afișate în timpul procesului de verificare au fost gândite pentru a oferi o experiență coerentă și prietenoasă, reducând la minimum pașii necesari pentru finalizarea unei validări.

**Respectarea bunelor practici de securitate**

Aplicația este construită respectând bunele practici din domeniul securității software, atât la nivel de comunicare, cât și la nivel de acces și control al datelor. Toate schimburile de informații se realizează exclusiv prin conexiuni criptate, folosind protocolul HTTPS, pentru a preveni interceptarea sau modificarea datelor în tranzit.

Pentru autentificare și autorizare, sistemul integrează Keycloak, o soluție open-source robustă, care permite gestionarea conturilor, a rolurilor și a permisiunilor într-un mod centralizat și sigur.

În cazul aplicației mobile, este implementată și autentificarea biometrică, care asigură protecția accesului la datele locale și la funcțiile critice ale aplicației, fără a compromite experiența utilizatorului.

## Arhitectura soluției

Figura 16 Arhitectura sistemului

Aplicația a fost construită pe o arhitectură bine gândită, care combină mai multe tehnologii moderne pentru a oferi o soluție sigură și ușor de folosit. Fiecare componentă joacă un rol important, iar împreună asigură funcționarea corectă a sistemului, astfel încât utilizatorii să poată valida atestări electronice într-un mod rapid și de încredere. Arhitectura soluției se bazează pe două mari elemente:

* Frontend (interfața grafică, aplicația mobile prin care utilizatorul interacționează cu sistemul)
* Backend (sistemul care gestionează logica aplicației și procesarea datelor)
* Infrastructură și comunicație (componentele care susțin securitatea și distribuția serviciilor)

Fiecare dintre cele trei componente principale ale aplicației este alcătuită din unul sau mai multe module, care colaborează pentru a permite funcționarea corectă a întregului sistem. Aceste module sunt:

* Frontend:
* Aplicația mobile CredCheck, dezvoltată în Android Studio cu Java
* Backend:
* Serviciul principal este dezvoltat cu Spring Boot (Java) și expune o serie de endpoint-uri REST prin care gestionează procesul de validare al credentialelor
* Integrarea cu portofelul digital se face conform specificațiilor OpenID4VP și SD-JWT
* Baza de date PostgreSQL este utilizată pentru logarea evenimentelor
* Registrul de stare este implementat în două forme:
* Status List bitstring, publicabil ca document static
* Filtre Bloom în cascadă, serializate într-un blob și distribuite printr-un nod blockchain
* Keycloak folosit pentru gestionarea autentificării și autorizării serviciilor și a aplicației mobile, pe baza protocolului OAuth2
* Unleash, folosit pentru configurarea dinamică a regulilor de validare, prin intermediul unui sistem de feature flags
* Infrastructură și comunicație:
* Docker folosit pentru containerizarea serviciilor backend, oferind instalare rapidă, izolare și portabilitate între medii
* Cloudflare Tunnel folosit pentru securizarea comunicațiilor externe prin HTTPS și expunerea controlată a serviciilor către utilizatori sau aplicații terțe
* Fiecare serviciu este disponibil printr-un subdomeniu dedicat:
* **auth.credcheck.site** pentru Keycloak
* **flags.credcheck.site** pentru Unleash
* **backend.credcheck.site** pentru serviciul de validare

## Descrierea modulelor componente

### Frontend

Aplicația mobile CredCheck este componenta prin care furnizorii de servicii pot verifica rapid atestările electronice prezentate de utilizatori. Dezvoltată în Java pentru platforma Android[33], CredCheck oferă o interfață simplă și intuitivă, astfel încât oricine o utilizează să poată efectua verificările fără a avea nevoie de cunoștințe tehnice avansate.

Unul dintre principalele roluri ale aplicației este generarea unui cod QR, care conține cererea de prezentare formulată conform standardului OpenID4VP. Codul este scanat de către portofelul digital al utilizatorului, care trimite direct credențialul semnat către backend-ul platformei. Ulterior, aplicația interoghează backend-ul pentru a obține starea tranzacției.

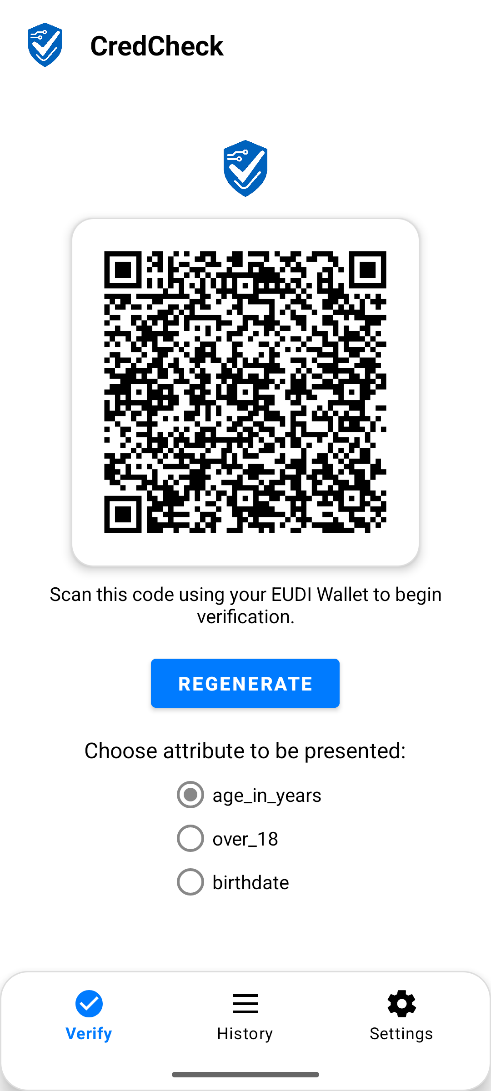


Figura 17 Pagina principală „Verify” a aplicației CredCheck

După generarea și trimiterea credențialului, aplicația afișează rezultatul verificării într-un mod ușor de înțeles, folosind icon-uri vizibile și mesaje clare, astfel încât utilizatorul să poată interpreta imediat răspunsul primit. Sunt afișate atât mesajele de acceptare, cât și cele de respingere a credențialelor verificate.

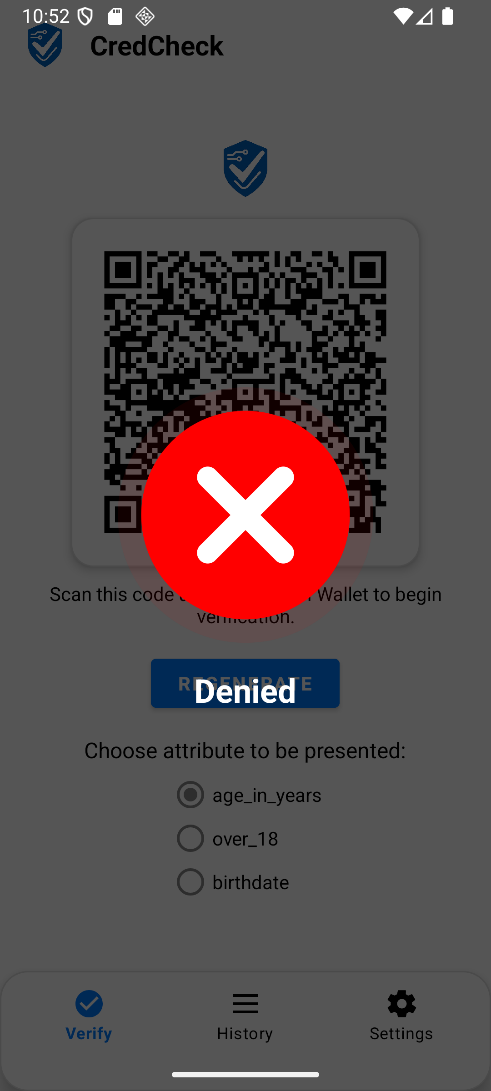
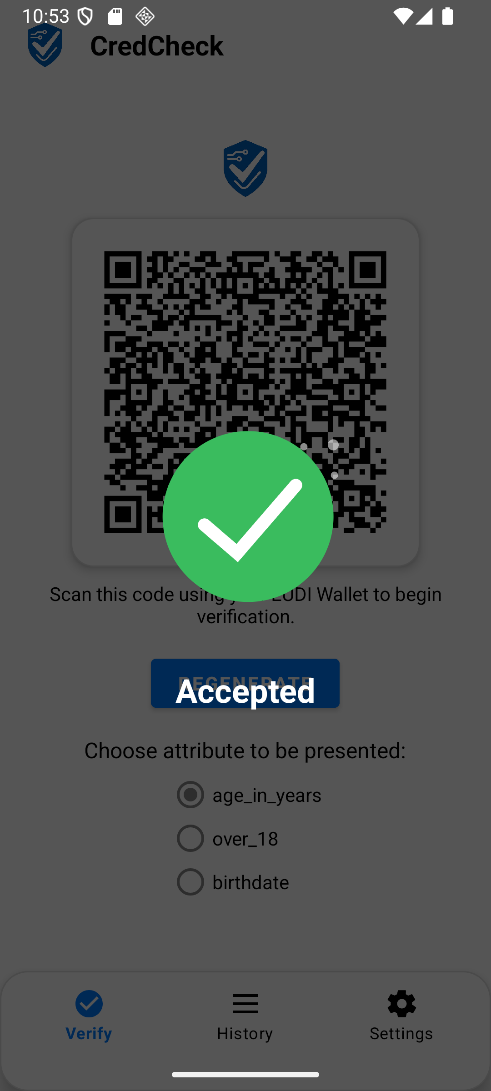


Figura 18 Feedback vizual pentru un credential validat cu succes

Figura 19 Feedback vizual pentru un credential invalid sau respins

Aplicația permite configurarea atributele ce urmează a fi solicitate din credențial, prin intermediul unui radio button gestionat dinamic prin feature flags. Astfel, în funcție de scenariul de utilizare, se pot solicita doar anumite date, precum vârsta, confirmarea că utilizatorul este major sau data completă a nașterii.

De asemenea, CredCheck include o secțiune dedicată istoricului tranzacțiilor efectuate, unde utilizatorul poate consulta lista tuturor verificărilor anterioare. Pentru o vizualizare rapidă a statisticilor, aplicația afișează și un grafic de tip pie chart, care evidențiază proporția credențialelor validate cu succes față de cele respinse.

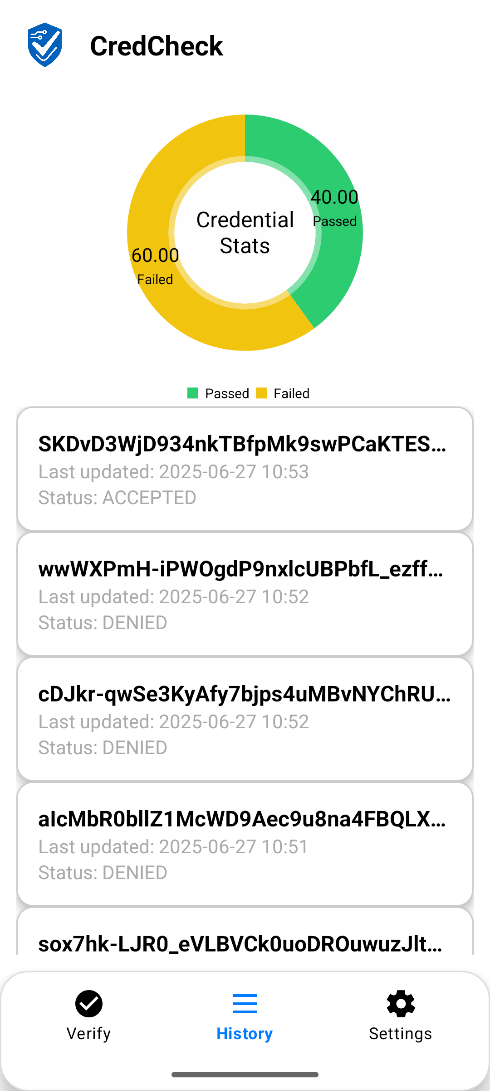


Figura 20 Pagina „History” a aplicației CredCheck

Nu în ultimul rând, aplicația oferă o pagină de setări, unde utilizatorul poate vizualiza informații despre aplicație, cum ar fi versiunea curentă și datele de contact pentru suport. Tot aici există opțiunea de a modifica tema aplicației, alegând între modurile disponibile, precum și posibilitatea de a activa sau dezactiva autentificarea biometrică, pentru un plus de securitate și acces rapid.

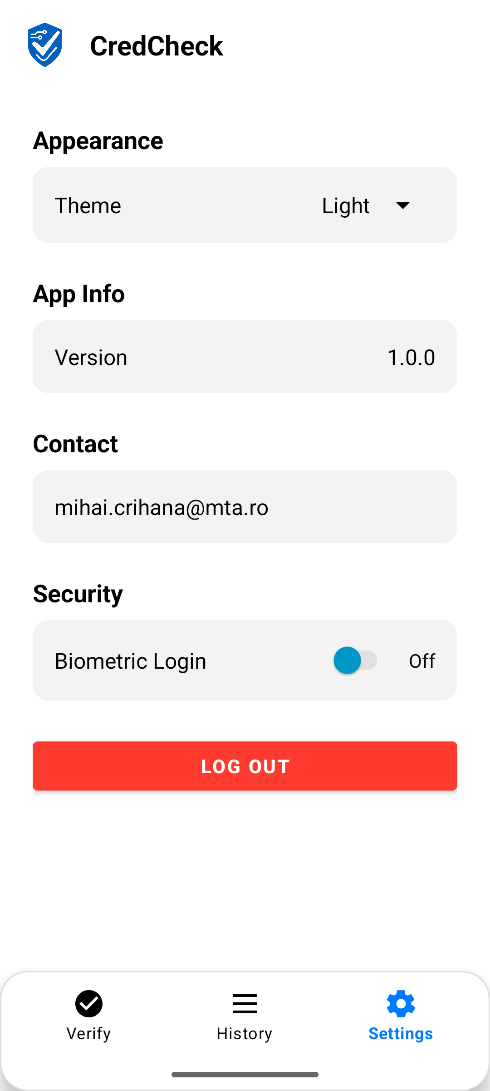


Figura 21 Pagina „Settings” a aplicației CredCheck

Prin integrarea tuturor acestor funcționalități într-o singură aplicație, CredCheck reușește să facă din validarea atestărilor electronice un proces accesibil și rapid, astfel încât utilizatorii să poată folosi această soluție fără dificultăți, indiferent de contextul în care au nevoie de ea.

Fluxul complet al interacțiunilor dintre aplicația mobilă CredCheck, portofelul digital al utilizatorului, serviciul de autentificare Keycloak și backend-ul platformei este ilustrat în *Figura 22*, evidențiind fiecare pas important din procesul de autentificare, generare a cererii de prezentare și validare a credentialelor, pentru o mai bună înțelegere a modului în care aceste componente colaborează în arhitectura sistemului.

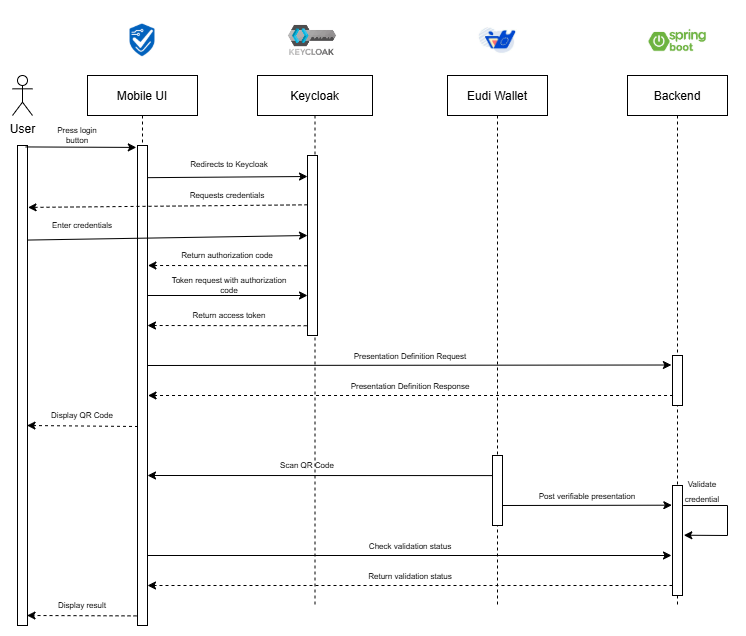


Figura 22 Fluxul de interacțiune dintre CredCheck, Keycloak, EUDI Wallet și backend

### Backend

La baza aplicației se află serverul dezvoltat folosind Spring Boot, un framework Java modern, dedicat dezvoltării rapide a aplicațiilor web și a microserviciilor. Backend-ul reprezintă modulul principal al aplicației, având rolul de a procesa toate cererile inițiate de aplicația mobile client și de a implementa logica de validare a atestărilor electronice de atribute.

Serverul utilizează arhitectura REST (Representational State Transfer) pentru a primi cereri și a furniza răspunsuri structurate, folosind metodele HTTP standard. Astfel, fiecare cerere inițiată de aplicația mobile este procesată prin intermediul unor apeluri REST, care asigură un transfer de date fiabil, rapid și scalabil. Acest model arhitectural permite separarea clară a funcționalităților backend-ului și o integrare ușoară cu aplicația mobile și cu serviciile externe utilizate în proiect.

Comunicația între client și backend se realizează prin protocolul HTTPS, asigurând astfel transmiterea securizată a datelor și protejarea utilizatorilor împotriva interceptării acestora.

Backend-ul este organizat în mai multe module, fiecare având o responsabilitate bine definită, precum generarea definiției de prezentare, validarea credentialelor, verificarea statusului acestora sau gestionarea evidențelor și logurilor. Împreună, aceste module colaborează pentru a oferi aplicației mobile funcționalitatea completă necesară, într-un mod sigur, eficient și conform cu standardele actuale din domeniul identității digitale.

1. **VerifierService**

*VerifierService* este serviciul care gestionează inițierea procesului de verificare a credențialelor. Rolul său principal este de a pregăti toate datele necesare pentru ca wallet-ul digital să poată trimite ulterior credențialele solicitate. Mai exact, atunci când aplicația mobile inițiază o cerere de verificare, *VerifierService* creează o nouă tranzacție, generează datele de identificare necesare și construiește request-ul ce urmează a fi folosit de wallet pentru prezentarea credențialelor. Astfel, acest serviciu asigură punctul de pornire al fluxului de validare, pregătind infrastructura necesară pentru ca utilizatorul să poată prezenta atestările electronice într-un mod sigur și organizat.

1. **WalletResponseValidator**

*WalletResponseValidator* este serviciul care se ocupă cu verificarea completă a credențialelor primite de la wallet-ul digital după prezentare. Rolul său principal este de a asigura că datele transmise sunt autentice, corecte și conforme cu așteptările definite în sistem. Mai exact, serviciul validează semnătura credențialului, verifică dacă acesta este sau nu revocat, confirmă că issuer-ul este unul de încredere și se ocupă de procesarea dezvăluirilor selective pentru a extrage informațiile necesare. Pe baza acestor verificări, *WalletResponseValidator* decide dacă credențialul primit poate fi acceptat sau nu, actualizând statusul tranzacției corespunzătoare. Astfel, acest serviciu are un rol esențial în garantarea încrederii și siguranței procesului de validare din backend.

1. **StatusListService**

*StatusListService* este serviciul responsabil cu verificarea statusului credentialelor pe baza *status list*-urilor publicate de issueri. Rolul său principal este de a determina dacă un credențial este valid sau a fost revocat, consultând lista de status asociată issuer-ului respectiv. În implementare, *StatusListService* folosește un mecanism de caching local: atunci când este necesară verificarea unui credențial, serviciul caută mai întâi *status* *list*-ul în baza de date, iar dacă aceasta nu există sau este expirată, o va descărca din nou de la sursa oficială și o va actualiza. Prin acest comportament, StatusListService asigură atât eficiența procesării prin reducerea numărului de cereri externe, cât și siguranța sistemului, garantând că doar credențialele active și nerevocate sunt considerate valide în procesul de verificare.

1. **BlockchainRevocationChecker și CascadeRevocationService**

*BlockchainRevocationChecker* este serviciul care inițiază procesul de verificare a revocării credențialelor prin interogarea smart contractelor de pe blockchain. Acesta obține de la contract adresa pointer-ului necesar pentru verificare și, în funcție de metoda configurată, folosește fie o structură de tip blob, fie un fișier stocat în IPFS.

În fluxul implementat, *BlockchainRevocationChecker* încearcă mai întâi metoda de verificare prin blob, apelând *CascadeRevocationService* pentru a verifica direct structura optimizată asociată credențialelor. Metoda blob presupune accesul rapid la o structură preprocesată care conține informațiile despre statusul credențialelor, fiind mai eficientă și mai rapidă în multe situații.

Dacă metoda blob indică faptul că este necesar un fallback (de exemplu, structura nu este disponibilă), *BlockchainRevocationChecker* inițiază automat metoda de rezervă, care utilizează IPFS. În acest caz, *CascadeRevocationService* este apelat pentru a descărca din rețeaua IPFS fișierul asociat pointer-ului obținut de pe blockchain și pentru a realiza verificarea statusului credentialului pe baza datelor conținute în acel fișier.

Astfel, *CascadeRevocationService* reprezintă serviciul care implementează efectiv logica de verificare atât pentru metoda blob, cât și pentru metoda IPFS, gestionând inclusiv caching-ul local al fișierelor sau structurilor descărcate, pentru a optimiza timpii de răspuns la apelurile ulterioare.

*BlockchainRevocationChecker* are rolul de a iniția verificarea revocării credențialelor folosind blockchain-ul și de a decide care metodă de verificare este potrivită în fiecare caz. *CascadeRevocationService* este cel care realizează efectiv verificarea revocării, fie prin metoda blob, fie prin IPFS, în funcție de datele obținute.

1. **JwtDecryptionService**

*JwtDecryptionService* este serviciul responsabil cu parsarea și decriptarea tokenurilor JWT primite în cadrul aplicației. Rolul său principal este de a extrage în siguranță informațiile conținute în aceste tokenuri și de a verifica structura lor, astfel încât datele obținute să poată fi folosite mai departe în procesele de validare și procesare din backend.

1. **EvidenceService**

*EvidenceService* este serviciul care se ocupă cu logarea detaliată a evenimentelor ce apar în timpul procesării tranzacțiilor. Mai exact, el înregistrează momentele cheie, cum ar fi inițierea unei tranzacții, preluarea request object-ului sau primirea răspunsului de la wallet, împreună cu datele relevante asociate fiecărui eveniment. Prin aceste funcționalități, *EvidenceService* creează o evidență completă și organizată a tuturor acțiunilor desfășurate în backend, facilitând astfel auditarea, monitorizarea și analiza ulterioară a funcționării aplicației.

1. **TransactionCleanupService**

*TransactionCleanupService* este serviciul care se ocupă cu ștergerea automată a tranzacțiilor care au rămas într-un status intermediar prea mult timp, fără a fi finalizate. Mai exact, acest serviciu rulează periodic și identifică tranzacțiile aflate în statusul „PENDING” de mai mult de două minute, eliminându-le din baza de date. Prin această funcționalitate, *TransactionCleanupService* contribuie la menținerea unei baze de date curate, prevenind acumularea de tranzacții incomplete care nu mai sunt relevante și asigurând astfel buna funcționare și performanța backend-ului.

1. **CreateJarService**

*CreateJarService* este serviciul responsabil cu generarea request object-urilor semnate, necesare pentru inițierea procesului de prezentare a credentialelor de către wallet-ul digital. Practic, acest serviciu construiește un JWT care conține toate informațiile despre cererea de prezentare, îl semnează folosind cheia privată a aplicației și îl pregătește pentru a fi trimis către wallet.

1. **TrustListService**

*TrustListService* este serviciul care verifică autenticitatea issuer-ilor ce emit credentialele. El parsează trust list-ul, o structură ce conține informații despre issuer-ii de încredere și cheile lor publice, pentru a confirma că credențialele prezentate provin de la un emitent valid și recunoscut în sistem.

1. **DefinitionService**

*DefinitionService* este serviciul care se ocupă cu încărcarea definițiilor de prezentare necesare wallet-ului digital. El citește fișierele JSON care conțin aceste definiții dintr-un director configurat și le transformă în obiecte utilizabile în backend, generând totodată un identificator unic pentru fiecare definiție încărcată.

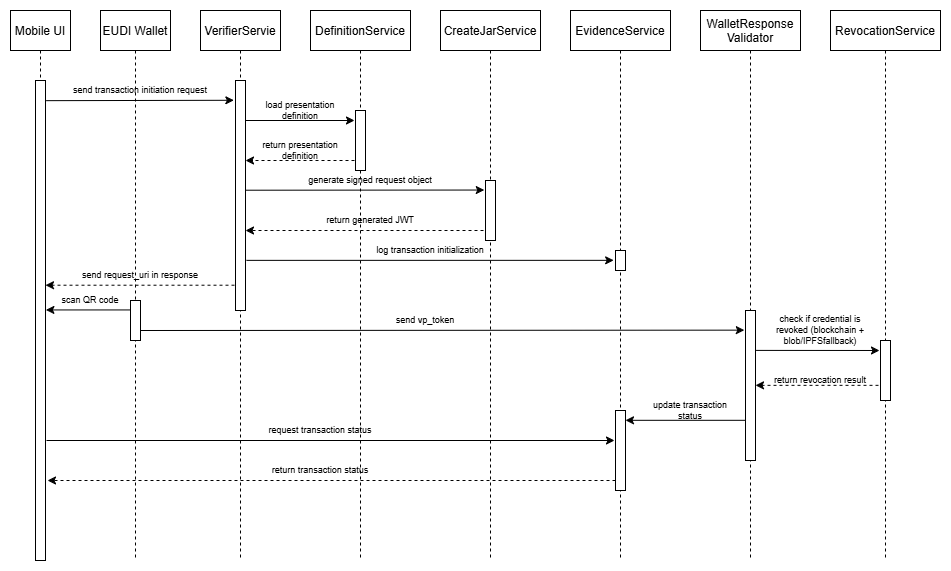


Figura 23 Fluxul de validare a credențialelor în backend

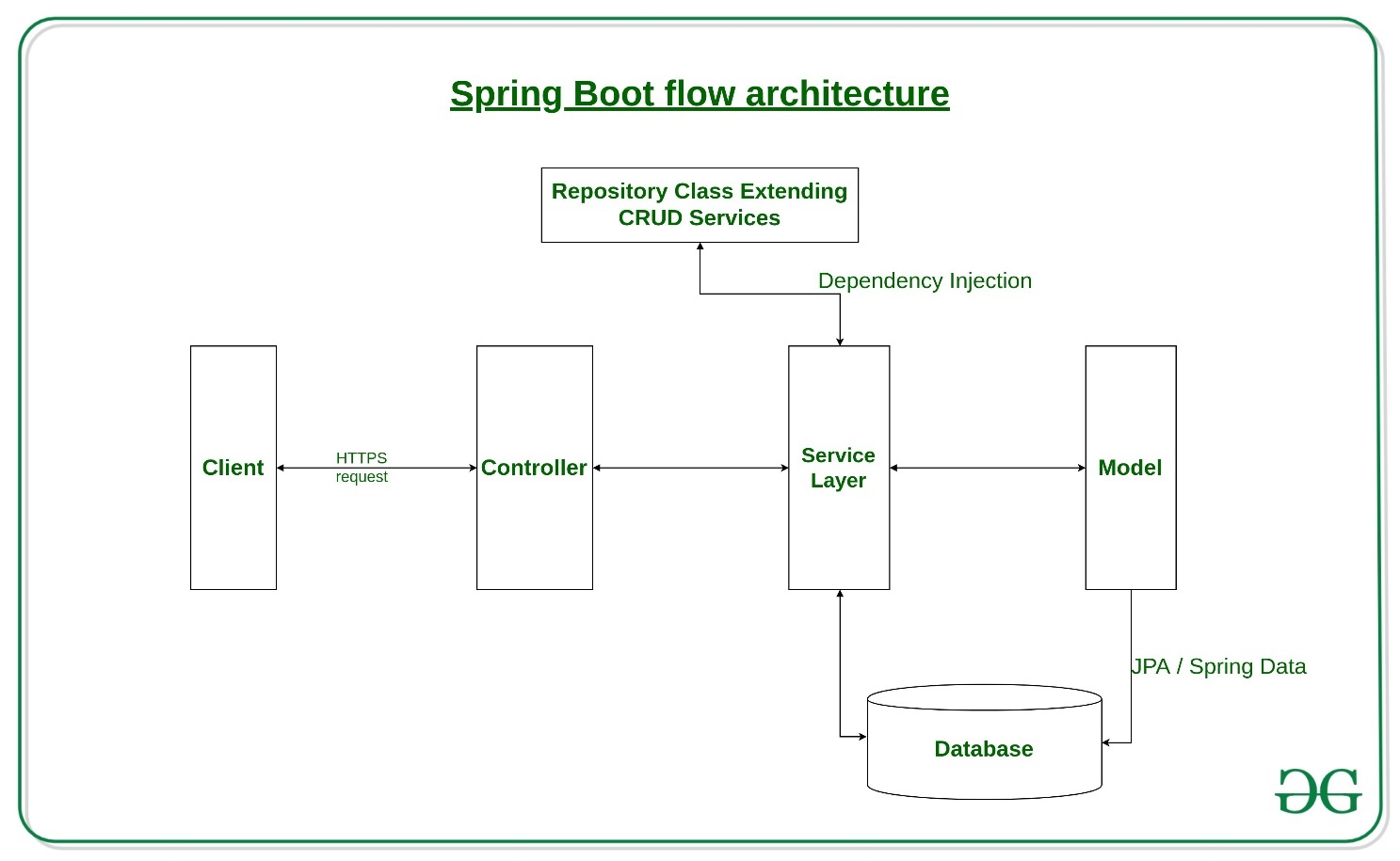
## Tehnologii folosite pentru implementare

### Java și Spring Boot

Java[34] este un limbaj de programare orientat obiect, dezvoltat inițial de Sun Microsystems și deținut acum de Oracle, fiind unul dintre cele mai utilizate limbaje din lume în aplicații enterprise și web. Popularitatea Java se datorează portabilității sale („write once, run anywhere”), stabilității și ecosistemului extins de librării și frameworkuri care ușurează dezvoltarea rapidă a aplicațiilor.

În proiectul realizat, limbajul Java a fost ales datorită suportului său extins pentru criptografie, stabilității și fiabilității sale ca limbaj de programare, precum și compatibilității excelente cu framework-ul Spring Boot[35], care a permis implementarea rapidă și eficientă a backend-ului.

Spring Boot este un framework open-source care extinde Spring Framework, având ca scop simplificarea procesului de creare a aplicațiilor Java. Spre deosebire de Spring tradițional, care presupunea configurări detaliate și fișiere XML, Spring Boot vine cu un sistem de auto-configurare și convenții implicite, eliminând astfel configurarea manuală a majorității componentelor. Această caracteristică a permis structurarea rapidă a proiectului, pornind de la o arhitectură REST clară și scalabilă.

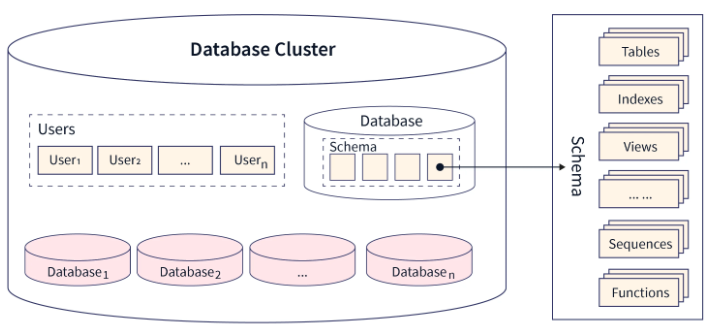
 Spring Boot oferă, de asemenea, un server web embedded, ceea ce înseamnă că aplicația poate fi pornită direct ca un JAR, fără a mai fi nevoie de configurarea separată a unui server de aplicații, ușurând astfel procesul de deployment. În proiectul meu, Spring Boot a fost utilizat pentru implementarea serviciilor REST ce gestionează fluxurile de validare a credentialelor, integrarea cu baza de date PostgreSQL și conectarea la Keycloak pentru autorizarea utilizatorilor. De asemenea, datorită compatibilității excelente cu Spring Security și librăriile pentru procesarea JWT-urilor (ex: Nimbus JOSE), Spring Boot a oferit toate instrumentele necesare dezvoltării rapide și sigure a backend-ului.

[[11]](#footnote-11)

Figura 24 Arhitectura fluxului unei aplicații Spring Boot

### PostgreSQL

PostgreSQL[36] este un sistem de baze de date relaționale open-source, apreciat pentru fiabilitatea, performanța și flexibilitatea lui. Oferă suport complet pentru standardul SQL, dar include și funcționalități suplimentare care îl fac potrivit pentru aplicații complexe, cum ar fi posibilitatea de a defini tipuri de date personalizate, funcții proprii și operatori adaptați nevoilor proiectului.

 Arhitectura sa este de tip client-server, ceea ce înseamnă că aplicația trimite interogări către serverul PostgreSQL, care le procesează și trimite înapoi rezultatele. Un avantaj important este sistemul MVCC (Multi-Version Concurrency Control), care permite mai multor tranzacții să ruleze în același timp fără blocaje, asigurând astfel performanță bună chiar și când sunt multe operații simultane.

[[12]](#footnote-12)

Figura 25 Arhitectura PostgreSQL

PostgreSQL respectă proprietățile ACID (Atomicitate, Consistență, Izolare și Durabilitate), garantând că datele rămân corecte și consistente, inclusiv în caz de erori de sistem. Suportă mai multe tipuri de indecși, precum B-tree, Hash sau GIN, care pot fi folosiți pentru optimizarea interogărilor, iar compatibilitatea sa cu tipurile de date JSON și JSONB îl face util inclusiv pentru aplicații care lucrează cu structuri de tip document.

În proiectul meu, PostgreSQL este folosit pentru a stoca tranzacțiile inițiate, statusurile acestora, logurile generate și configurările aplicației. Integrarea cu Spring Boot prin Spring Data JPA a permis definirea rapidă a entităților și a operațiilor necesare asupra datelor, fără a fi nevoie de interogări SQL scrise manual. De asemenea, PostgreSQL poate fi rulat ușor în containere Docker, lucru care a simplificat implementarea aplicației.

### Keycloak

Keycloak[37] este o soluție open-source pentru gestionarea autentificării și autorizării utilizatorilor, fiind recunoscut ca un instrument complet de Identity and Access Management (IAM). Este dezvoltat de Red Hat și este folosit în multe aplicații enterprise datorită flexibilității și integrării rapide cu diverse tehnologii.

Unul dintre principalele avantaje ale Keycloak este că permite adăugarea rapidă a mecanismelor de login în aplicații, fără a mai fi nevoie de implementarea manuală a acestora. Keycloak oferă suport pentru autentificare bazată pe username și parolă, login social (ex: Google, Facebook), Single Sign-On (SSO), precum și pentru protocoale standardizate precum OAuth 2.0 și OpenID Connect.

Arhitectura Keycloak este de tip client-server, în care aplicația client redirecționează utilizatorul către serverul Keycloak pentru login. După autentificare, Keycloak returnează un token JWT (JSON Web Token) care conține informații despre utilizator și rolurile sale. Acest token este trimis apoi de client către backend la fiecare request, iar backend-ul îl validează pentru a permite sau restricționa accesul la anumite resurse.

În proiectul meu, Keycloak a fost folosit pentru a gestiona autentificarea aplicației mobile, precum și accesul la endpoint-urile din backend. Astfel, utilizatorii pot fi autentificați într-un mod standardizat, iar backend-ul poate controla accesul pe baza rolurilor atribuite fiecărui utilizator, fără a fi nevoie să gestioneze direct credentialele acestora.

Un alt avantaj important al Keycloak este interfața sa web intuitivă, care permite administrarea ușoară a utilizatorilor, a grupurilor, a rolurilor și a permisiunilor. De asemenea, Keycloak oferă și posibilitatea configurării de politici de securitate, precum durata de valabilitate a token-urilor, parole temporare și reguli pentru resetarea parolelor, contribuind astfel la creșterea securității aplicației.

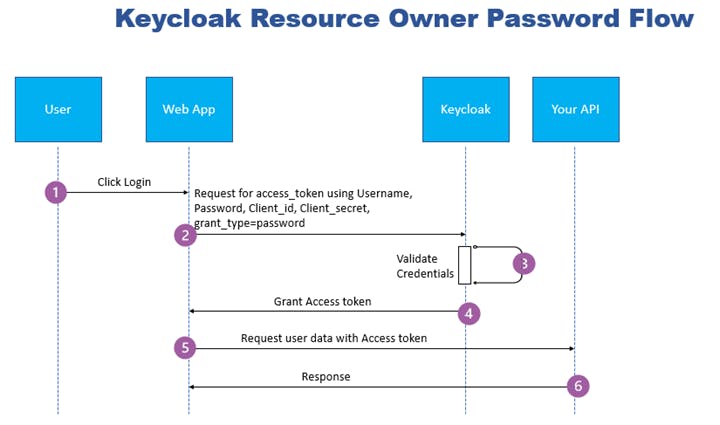
 [[13]](#footnote-13)

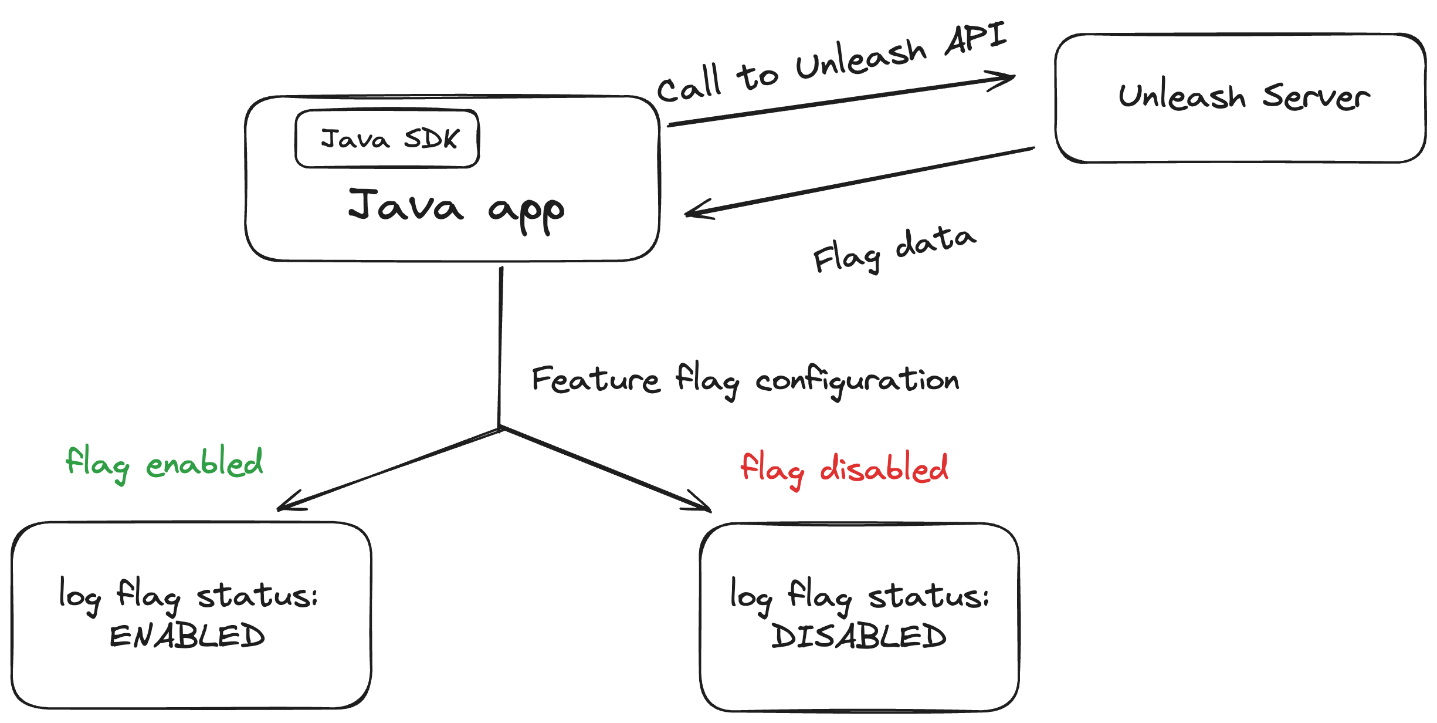
Figura 26 Fluxul ,,Resource Owner Password" în Keycloak

### Unleash

Unleash[38] este o platformă open-source pentru gestionarea feature flag-urilor, folosită pentru a controla activarea și configurarea anumitor funcționalități într-o aplicație, fără a fi necesară modificarea codului sursă și redeploy-ul acesteia. Este o soluție practică în proiectele care presupun dezvoltare și testare continuă, deoarece permite schimbarea rapidă a comportamentului aplicației în funcție de nevoi.

În proiectul meu, Unleash este folosit pentru configurarea dinamică a opțiunilor care definesc ce atribute pot fi cerute de la un credential. Mai exact, prin intermediul feature flag-urilor, pot modifica direct din interfața Unleash opțiunile afișate în aplicația mobilă, fără a fi nevoie să actualizez backend-ul. Acest lucru este util mai ales în etapele de testare, unde pot schimba rapid ce date sunt cerute utilizatorului, validând astfel diverse scenarii de prezentare și verificare a credentialelor.

Arhitectura Unleash include un server care gestionează toate flag-urile definite și clienți care rulează în aplicație și interoghează periodic serverul pentru a afla statusul fiecărui flag. În acest fel, aplicația backend poate decide în timp real ce atribute să solicite utilizatorului, pe baza valorilor configurate în Unleash.

 Un alt avantaj important este interfața sa web, ușor de utilizat, prin care pot fi adăugate, modificate sau șterse flag-uri, precum și definite strategii de activare, fără a fi necesare cunoștințe de programare. De asemenea, integrarea sa într-un container Docker a simplificat mult deploy-ul în infrastructura proiectului.

[[14]](#footnote-14)

Figura 27 Arhitectura Unleash

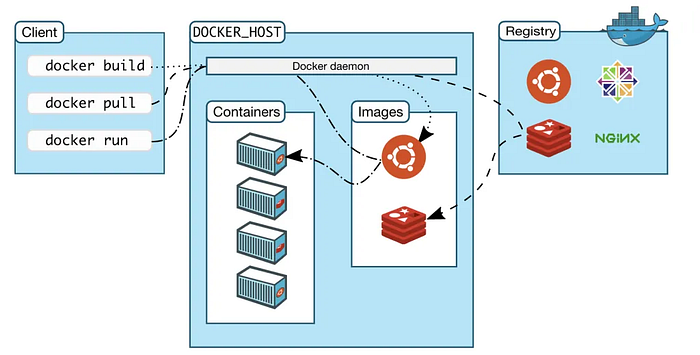
### Docker

Docker[39] este o platformă open-source care permite containerizarea aplicațiilor, adică rularea acestora împreună cu toate dependințele necesare, într-un mediu izolat, reproductibil, indiferent de sistemul de operare pe care rulează. Prin utilizarea containerelor, aplicațiile devin mai ușor de dezvoltat și deployat, eliminând problemele care apar de obicei din cauza diferențelor dintre mediile de dezvoltare și producție.

Arhitectura Docker se bazează pe imagini și containere. O imagine Docker conține tot ce este necesar pentru rularea unei aplicații: codul sursă, librăriile, runtime-ul și configurațiile. Din această imagine se creează containerul, care reprezintă instanța rulată efectiv pe sistemul gazdă.

În proiectul meu, Docker a fost utilizat pentru a containeriza backend-ul Spring Boot, PostgreSQL, Keycloak și Unleash. Astfel, toate serviciile au putut fi pornite rapid, fără configurări suplimentare, prin folosirea fișierelor docker-compose, care definesc modul în care mai multe containere rulează împreună și interacționează în rețea.

Un alt avantaj major al Docker este izolarea aplicațiilor. Fiecare container rulează independent, cu propriile sale librării și procese, fără a interfera cu alte aplicații de pe sistem. În plus, consumul de resurse este redus, deoarece containerele împart kernelul sistemului gazdă, spre deosebire de mașinile virtuale clasice care emulează un sistem de operare complet.

 Docker Hub pune la dispoziție imagini oficiale pentru majoritatea tehnologiilor folosite în proiect (Java, PostgreSQL, Keycloak, Unleash), lucru care a simplificat mult configurarea și integrarea acestora în infrastructura aplicației.

[[15]](#footnote-15)

Figura 28 Arhitectura Docker

### Cloudflare

Cloudflare[40] este o platformă globală de servicii web care oferă funcționalități de tip Content Delivery Network (CDN), securitate și optimizare a performanței aplicațiilor online. Este folosită de milioane de site-uri și aplicații datorită fiabilității, vitezei de răspuns și protecției avansate împotriva atacurilor cibernetice.

În proiectul meu, Cloudflare a fost utilizat în primul rând pentru asigurarea comunicației securizate prin HTTPS. Prin integrarea cu Cloudflare, toate cererile trimise către domeniile aplicației sunt criptate, protejând astfel datele utilizatorilor împotriva interceptării. Platforma oferă certificate SSL gratuite, configurabile rapid, fără a necesita generarea sau instalarea manuală pe server.

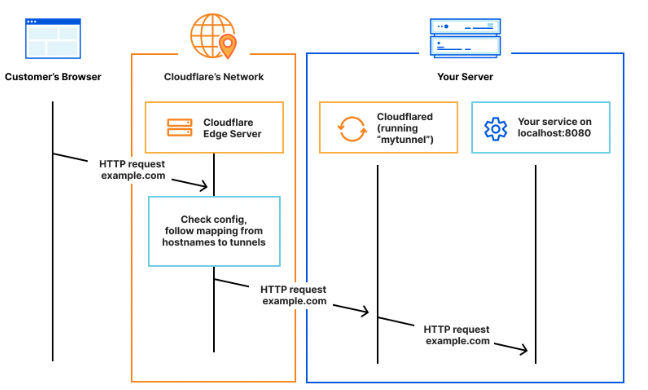
Un alt beneficiu major este funcționalitatea de reverse proxy. Astfel, toate cererile către aplicație trec mai întâi prin rețeaua Cloudflare, care filtrează traficul și blochează cererile malițioase, reducând riscul atacurilor de tip DDoS sau accesului neautorizat.

În proiectul meu, am folosit și Cloudflare Tunnel, care permite expunerea backend-ului fără a deschide porturi publice pe serverul local. Cloudflare Tunnel creează un canal securizat între server și infrastructura Cloudflare, permițând accesul public la backend doar prin rețeaua Cloudflare, crescând astfel semnificativ securitatea aplicației.

Prin Cloudflare am expus public următoarele servicii:

* **backend.credcheck.site** – pentru accesarea serviciilor backend Spring Boot
* **auth.credcheck.site** – pentru serviciul de autentificare Keycloak
* **flags.credcheck.site** – pentru serverul Unleash, folosit la gestionarea feature flag-urilor

Un alt beneficiu important al Cloudflare este rețeaua sa globală de edge servers, care cache-uiesc conținut static și redirecționează traficul prin ruta optimă, reducând astfel latența și timpul de încărcare al aplicației pentru utilizatori, indiferent de locația acestora.

 Cloudflare oferă și o interfață web intuitivă, prin care pot fi configurate rapid DNS-ul, certificatul SSL, politicile de firewall, redirecționările și alți parametri necesari infrastructurii unei aplicații moderne.[40]

[[16]](#footnote-16)

Figura 29 Tunel Cloudflare

## Rezultatele testelor efectuate și interpretarea acestora

Aplicația dezvoltată are o structură complexă, fiind compusă din mai multe module și servicii care interacționează între ele, motiv pentru care asigurarea calității este un aspect esențial în cadrul proiectului. Pentru ca aplicația să fie fiabilă și să funcționeze conform așteptărilor, a fost nevoie să realizez o serie de teste care să confirme corectitudinea implementării.

Testarea este una dintre etapele importante în dezvoltarea oricărei aplicații software, deoarece prin testare pot fi identificate din timp posibile erori sau comportamente neașteptate, care pot afecta experiența utilizatorului final. În același timp, testarea contribuie la creșterea încrederii în aplicație și oferă siguranța că aceasta poate fi utilizată fără probleme în scenariile reale pentru care a fost creată.

În cadrul testelor efectuate, am urmărit aspecte precum funcționalitatea fiecărui modul, modul de interacțiune dintre module (backend, aplicație mobilă, wallet digital), dar și comportamentul sistemului în ansamblu, pentru a mă asigura că toate componentele lucrează împreună într-un mod eficient.

### Testarea unitară

Testarea unitară reprezintă verificarea celor mai mici componente ale unei aplicații, cum ar fi funcțiile, metodele sau clasele, pentru a confirma că funcționează corect independent de restul sistemului. Este un tip de testare white-box, realizată de obicei de dezvoltator, care urmărește să identifice rapid eventualele erori din logica unei componente, înainte ca aceasta să fie integrată în fluxul complet al aplicației.

Prin testarea unitară sunt verificate atât comportamentele așteptate ale funcțiilor, cât și situațiile în care acestea primesc date de intrare invalide sau lipsă, asigurând astfel robustețea implementării.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr. Crt.** | **Cerința** | **Modalitate de testare** | **Rezultate așteptate** | **Erori posibile** |
| TU01 | Generarea unui presentation definition | 1. Se pregătește un fișier json valid în path-ul configurat. 2. Se apelează loadDefinition() din DefinitionService cu numele fișierului. 3. Se verifică dacă obiectul returnat nu este null și are ID generat. | Se returnează un obiect PresentationDefinitionTO cu ID unic generat corect. | 1. Fișierul nu există. 2. Format JSON invalid. 3. UUID nu este generat corect. |
| TU02 | Generarea id-urilor unice | 1. Se apelează generateTransactionId() și generateRequestId() din IdGenerator de mai multe ori.  2. Se compară rezultatele pentru a confirma unicitatea.  3. Se validează formatul UUID. | Fiecare ID generat este unic și respectă formatul UUID. | 1. Generarea ID-ului eșuează.  2. Format incorect generat. |
| TU03 | Inițierea tranzacțiilor | 1. Se creează un obiect InitTransactionTO valid.  2. Se apelează handleInitTransaction() din VerifierService.  3. Se verifică salvarea tranzacției cu status PENDING în repository.  4. Se validează InitTransactionResponse returnat. | Tranzacția este salvată cu status PENDING și răspunsul include ID-ul tranzacției și requestUri-ul generat. | 1. IdGenerator eșuează.  2. Repository indisponibil.  3. PresentationDefinition invalid. |
| TU04 | Generarea request object-ului JWT | 1. Se creează o prezentare Presentation.Requested validă. 2. Se apelează createSignedRequestObject() din CreateJarService. 3. Se decodează JWT-ul și se verifică semnătura și claims-urile incluse. | JWT-ul generat este valid, semnat corect și include toate claims-urile necesare. | 1. Cheia privată lipsește.  2. ECDSASigner aruncă excepție.  3. Date lipsă în prezentare. |
| TU05 | Generarea cheii efemere | 1. Se apelează generateEphemeralEncryptionKey() din EphemeralKeyGenerator. 2. Se verifică tipul obiectului returnat. 3. Se validează parametrii ECKey generați. | Cheia efemeră ECKey este generată corect și are parametrii de criptare setați. | 1. Generatorul de chei eșuează.  2. Algoritmul de criptare nu este disponibil. |
| TU06 | Validarea semnăturii SD-JWT (caz valid și invalid) | 1. Se pregătește un vp\_token valid semnat de un issuer autorizat.  2. Se apelează validate() din WalletResponseValidator cu vp\_token-ul valid și se verifică faptul că metoda finalizează fără excepții și actualizează statusul tranzacției la ACCEPTED.  3. Se pregătește un vp\_token invalid (modificat semnătura).  4. Se apelează validate() cu vp\_token-ul invalid și se verifică faptul că este aruncată o excepție corespunzătoare, iar tranzacția nu este acceptată. | Pentru vp\_token valid: semnătura este validată și credentialul acceptat. Pentru vp\_token invalid: este aruncată excepție și credentialul nu este acceptat. | 1. Certificatul x5c lipsește din header. 2. Format JWT invalid. 3. Cheia publică nu este de tip EC. 4. Semnătura JWT invalidă trece greșit validarea. 5. Excepția nu este tratată corespunzător. |
| TU07 | Verificarea consistenței sd\_hashes dintr-un credential | 1. Se pregătește un vp\_token valid cu disclosures corespunzătoare. 2. Se apelează validateDisclosureAndCheckAge() cu disclosure-ul și lista de sd\_hashes. 3. Se verifică dacă hash-ul calculat corespunde cu lista sd\_hashes. | Hash-ul calculat este găsit în sd\_hashes și disclosure-ul este considerat valid. | 1. Saltul din disclosure este corupt.  2. Calcul hash incorect.  3. Format disclosure invalid. |
| TU08 | Validarea disclosures | 1. Se creează disclosure-uri valide pentru diverse atribute (ex: is\_over\_18, birthdate).  2. Se apelează validateDisclosureAndCheckAge().  3. Se verifică returnarea valorii boolean corespunzătoare în funcție de datele din disclosure. | Disclosure-ul este validat corect și se determină majoratul sau atributul corespunzător. | 1. Disclosure-ul nu conține 3 elemente.  2. Conversia datei de naștere eșuează.  3. Claim necunoscut. |
| TU09 | Verificarea statusului unui credential (valid și revocat) | 1. Se pregătește un index și un status list care conține credentialul ca valid (bit=0). 2. Se apelează isCredentialValid() din StatusListService și se verifică returnarea true. 3. Se modifică status list-ul pentru ca bitul credentialului să fie setat ca revocat (bit=1). 4. Se apelează din nou metoda și se verifică returnarea false. | Pentru status valid: metoda returnează true. Pentru status revocat: metoda returnează false. | 1. Index out of range.  2. Status list invalid sau corupt.  3. Status list expirată.  4. Valoarea bitului interpretată greșit. |
| TU10 | Decompresia status list-ului | 1. Se pregătește un status list comprimat valid în format Base64.  2. Se apelează metoda decompress() din StatusListService.  3. Se verifică faptul că rezultatul decompresiei corespunde structurii originale a listei de biți. | Status list-ul este decompresat corect, fără pierderi de date. | 1. Format invalid al listei comprimate.  2. Date corupte care generează excepții la decompresie.  3. Buffer overflow la alocarea spațiului de decompresie. |
| TU11 | Verificarea fallback-ului blob – IPFS | 1. Se pregătește un pointerHash valid și un index pentru verificarea prin blob.  2. Se apelează isRevokedViaBlob() din CascadeRevocationService și se simulează necesitatea fallback-ului (ex. blob indisponibil).  3. Se verifică apelarea automată a fallback-ului IPFS prin isRevokedViaIpfs() și returnarea rezultatului corect. | Dacă blob-ul eșuează, fallback-ul IPFS este apelat și returnează statusul corect al credentialului. | 1. Pointer hash invalid.  2. IPFS indisponibil după fallback.  3. Rezultat inconsistent între blob și IPFS. |
| TU12 | Verificarea completă a BlockchainRevocationChecker (credential valid și revocat) | 1. Se configurează parametrii (contractAddress, abiJson, issuerAddress, index) pentru un credential valid.  2. Se apelează checkRevocationViaBlockchain() din BlockchainRevocationChecker.  3. Se verifică faptul că metoda returnează false (credential valid, nerevocat).  4. Se configurează parametrii pentru un credential revocat pe blockchain.  5. Se apelează din nou metoda.  6. Se verifică faptul că metoda returnează true (credential revocat). | Pentru credential valid: metoda returnează false. Pentru credential revocat: metoda returnează true. | 1. Contract ABI invalid.  2. Conexiune eșuată la blockchain.  3. Contractul nu conține pointerHash-ul.  4. Index invalid.  5. Fallback IPFS eșuează.  6. Excepții la rularea scriptului Python. |
| TU13 | Actualizarea statusului tranzacției la ACCEPTED în EvidenceService | 1. Se creează o tranzacție de test cu status PENDING.  2. Se apelează metoda care validează credentialul și actualizează statusul la ACCEPTED.  3. Se verifică în baza de date că statusul tranzacției a fost actualizat corect la ACCEPTED. | Statusul tranzacției este actualizat la ACCEPTED și persistat în baza de date. | 1. Tranzacția nu este găsită în baza de date.  2. Statusul nu este actualizat din cauza unei erori ORM.  3. Persistarea tranzacției eșuează. |
| TU14 | Actualizarea statusului tranzacției la DENIED în EvidenceService | 1. Se creează o tranzacție de test cu status PENDING.  2. Se apelează metoda care invalidează credentialul și actualizează statusul la DENIED.  3. Se verifică în baza de date că statusul tranzacției a fost actualizat corect la DENIED. | Statusul tranzacției este actualizat la DENIED și persistat în baza de date. | 1. Tranzacția nu este găsită în baza de date.  2. Statusul nu este actualizat din cauza unei erori ORM.  3. Persistarea tranzacției eșuează. |
| TU15 | Logarea inițierii tranzacției | 1. Se inițiază o tranzacție nouă prin VerifierService.  2. Se apelează metoda logTransactionInitialized() din EvidenceService.  3. Se verifică în baza de date că event-ul a fost adăugat în istoricul tranzacției. | Evenimentul de inițiere este logat corect în tranzacție. | 1. Tranzacția nu este găsită.  2. Event-ul nu este salvat în baza de date.  3. Persistarea tranzacției eșuează. |
| TU16 | Logarea răspunsului wallet-ului | 1. Se creează o tranzacție de test cu presentationDefinitionId cunoscut.  2. Se apelează metoda logWalletResponsePosted() cu un payload simulând un răspuns valid.  3. Se verifică în baza de date că event-ul a fost adăugat la tranzacție. | Evenimentul de răspuns este logat corect în tranzacție. | 1. Tranzacția nu este găsită.  2. Event-ul nu este adăugat în lista de events.  3. Persistarea tranzacției eșuează. |
| TU17 | Ștergerea tranzacțiilor PENDING vechi | 1. Se creează în baza de date tranzacții PENDING cu timestamp mai vechi de 2 minute.  2. Se apelează metoda deleteStalePendingTransactions() din TransactionCleanupService.  3. Se verifică că tranzacțiile vechi au fost șterse din baza de date. | Tranzacțiile PENDING vechi sunt șterse corect. | 1. Query-ul pentru selectarea tranzacțiilor vechi eșuează. 2. Operația de ștergere în masă eșuează. 3. Conexiunea la baza de date este indisponibilă. |

Tabelul 2 Testele unitare

### Testarea de integrare

Testarea de integrare are rolul de a verifica modul în care diferitele componente ale aplicației funcționează împreună, după ce au fost testate individual prin teste unitare. Este un tip de testare white-box, deoarece presupune cunoașterea arhitecturii interne a aplicației și a modului în care sunt organizate modulele sale.

Prin testarea de integrare se urmărește validarea funcționalității fluxurilor care implică mai multe clase, servicii sau module, pentru a asigura că nu apar probleme în momentul în care acestea interacționează. Chiar dacă fiecare componentă funcționează corect individual, integrarea lor poate genera erori neașteptate, motiv pentru care acest tip de testare este esențial.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr. Crt.** | **Cerința** | **Modalitate de testare** | **Rezultate așteptate** | **Erori posibile** |
| TI01 | Verificarea interacțiunii dintre aplicația mobilă și backend | 1. Se pornește backend-ul și aplicația mobile în mediu de test.  2. Se inițiază o cerere din aplicația mobilă pentru generarea unui request URI.  3. Se verifică răspunsul backend-ului și afișarea codului QR în aplicație. | Aplicația primește request URI valid și afișează codul QR. | 1. Backend-ul nu răspunde.  2. Formatul request URI-ului este invalid.  3. Conexiunea HTTPS eșuează. |
| TI02 | Verificarea fluxului complet inițiere tranzacție, scanare QR, wallet, backend | 1. Se inițiază o tranzacție din aplicația mobile. 2. Se scanează codul QR generat cu wallet-ul EUDI. 3. Se trimite credentialul către backend și se verifică răspunsul final. | Fluxul complet se finalizează cu validarea corectă a credentialului și actualizarea statusului tranzacției. | 1. Wallet-ul nu trimite credentialul. 2. Backend-ul nu validează credentialul. 3. Statusul tranzacției nu este actualizat. |
| TI03 | Verificarea integrării backend-ului cu PostgreSQL pentru salvarea și actualizarea tranzacțiilor | 1. Se inițiază o tranzacție nouă prin backend.  2. Se verifică în baza de date că tranzacția este salvată cu status PENDING.  3. Se validează credentialul și se verifică actualizarea statusului în DB. | Tranzacția este salvată și statusul se actualizează corect în PostgreSQL. | 1. Conexiunea la baza de date eșuează.  2. Persistarea tranzacției eșuează. |
| TI04 | Integrarea backend-ului cu Keycloak pentru autentificarea și autorizarea utilizatorilor | 1. Se configurează Keycloak cu un user de test.  2. Se trimite o cerere autorizată către backend cu token JWT generat de Keycloak.  3. Se verifică accesul la endpoint-ul protejat. | Backend-ul acceptă cererea și validează JWT-ul emis de Keycloak. | 1. Token-ul JWT este invalid sau expirat.  2. Configurația Keycloak este greșită. |
| TI05 | Integrarea backend-ului cu Unleash pentru citirea feature flag-urilor | 1. Se configurează Unleash cu un flag de test activ. 2. Se pornește backend-ul și se inițiază o cerere care citește flag-ul. 3. Se verifică răspunsul backend-ului în funcție de valoarea flag-ului. | Backend-ul citește corect flag-ul din Unleash și adaptează comportamentul. | 1. Conexiunea la Unleash eșuează. 2. Flag-ul nu este găsit. 3. Backend-ul nu aplică corect logica pe baza flag-ului. |
| TI06 | Integrarea backend-ului cu Cloudflare Tunnel pentru acces securizat din exterior | 1. Se configurează Cloudflare Tunnel pentru backend.  2. Se accesează un endpoint backend prin URL-ul Cloudflare.  3. Se verifică răspunsul și certificatul SSL. | Backend-ul este accesibil securizat prin Cloudflare Tunnel. | 1. Cloudflare Tunnel nu este configurat corect. 2. Conexiunea HTTPS eșuează. 3. Certificatul SSL nu este valid. |

Tabelul 3 Testete de integrare

### Testare de performanță

Testarea de performanță are rolul de a evalua rapiditatea și stabilitatea aplicației în diferite scenarii de utilizare. Prin această testare am urmărit să determin cât timp durează procesarea unei validări de credential în funcție de metodele de verificare folosite și de modul în care status list-ul este gestionat (cache-uit sau nu). Astfel, am putut identifica eventualele limitări de performanță și optimiza fluxurile critice pentru utilizatori.

Pentru proiectul meu, am realizat cronometrarea timpului de răspuns al backend-ului în următoarele scenarii:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. Crt.** | **Cerința** | **Modalitate de testare** | **Erori posibile** |
| TP01 | Validare folosind doar Status List cu lista cache-uită | 1. Se pregătește un credential valid cu status list deja salvată în cache.  2. Se trimite cererea de validare.  3. Se măsoară timpul total de răspuns. | 1. Cache-ul nu este accesat corect. 2. Status list-ul din cache este expirat. |
| TP02 | Validare folosind doar Status List cu lista necachuită | 1. Se șterge status list-ul din cache.  2. Se trimite cererea de validare.  3. Se măsoară timpul total de răspuns. | 1. Download-ul status list-ului eșuează.  2. Timeout la validare. |
| TP03 | Validare folosind doar metoda blob cu status list cache-uită | 1. Se pregătește credentialul valid, status list cache-uită.  2. Se trimite cererea de validare folosind metoda blob.  3. Se cronometrează răspunsul. | 1. Conexiunea la blob storage eșuează.  2. Verificarea blob returnează eroare. |
| TP04 | Validare folosind doar metoda blob cu status list necachuită | 1. Se șterge status list-ul din cache.  2. Se trimite cererea de validare folosind metoda blob.  3. Se cronometrează timpul total de răspuns. | 1. Fetch-ul status list-ului eșuează.  2. Conexiunea la blob storage eșuează. |
| TP05 | Validare cu fallback pe IPFS cu status list cache-uită | 1. Se pregătește credentialul valid, status list cache-uită.  2. Se simulează eroare la blob storage.  3. Se testează fallback-ul pe IPFS și se cronometrează răspunsul. | 1. IPFS este indisponibil.  2. Fallback-ul nu este activat corect. |
| TP06 | Validare cu fallback pe IPFS cu status list necachuită | 1. Se șterge status list-ul din cache.  2. Se simulează eroare la blob storage.  3. Se trimite cererea de validare și se cronometrează răspunsul. | 1. Fetch-ul status list-ului eșuează.  2. IPFS nu returnează blob-ul corect. |
| TP07 | Validare combinată blob + Status List cu lista cache-uită | 1. Se pregătește credentialul valid, status list cache-uită.  2. Se trimite cererea de validare care verifică ambele metode.  3. Se cronometrează timpul total. | 1. Conexiunea la blob storage eșuează.  2. Validarea status list-ului eșuează. |
| TP08 | Validare combinată blob + Status List cu lista necachuită | 1. Se șterge status list-ul din cache.  2. Se trimite cererea de validare combinată.  3. Se cronometrează răspunsul. | 1. Download-ul status list-ului eșuează.  2. Conexiunea la blob storage eșuează. |
| TP09 | Validare combinată IPFS + Status List cu lista cache-uită | 1. Se pregătește credentialul valid, status list cache-uită.  2. Se simulează eroare la blob pentru a folosi IPFS.  3. Se trimite cererea de validare și se cronometrează răspunsul. | 1. IPFS nu returnează datele corect.  2. Fallback-ul pe IPFS nu funcționează. |
| TP10 | Validare combinată IPFS + Status List cu lista necachuită | 1. Se șterge status list-ul din cache.  2. Se simulează eroare la blob pentru a forța IPFS.  3. Se trimite cererea de validare și se cronometrează timpul. | 1. Download status list eșuat.  2. IPFS nu răspunde în timp util. |

Tabelul 4 Testele de performanță

### Testare de acceptanță

Testarea de acceptanță are ca scop confirmarea faptului că aplicația implementată respectă toate cerințele inițiale și poate fi utilizată efectiv în scenariile reale pentru care a fost proiectată. Aceasta este ultima etapă de testare înainte de lansarea produsului și validează funcționalitatea din perspectiva utilizatorului final.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr. Crt.** | **Cerința** | **Modalitate de testare** | **Rezultate așteptate** | **Erori posibile** |
| TA01 | Inițierea tranzacției din aplicația mobile și generarea codului QR | 1. Se accesează aplicația mobile.  2. Se apasă butonul pentru inițiere tranzacție.  3. Se verifică generarea și afișarea codului QR. | Cod QR generat și afișat corect pe ecran. | 1. Conexiunea la backend eșuează.  2. Codul QR nu este generat din cauza lipsei datelor de request URI. |
| TA02 | Scanarea codului QR cu wallet-ul digital și trimiterea credentialului către backend | 1. Se scanează codul QR generat în TA01 cu EUDI Wallet. 2. Se selectează credentialul necesar. 3. Se trimite credentialul către backend. | Credentialul este trimis și primit de backend fără erori. | 1. Wallet-ul nu trimite credentialul. 2. Cod QR invalid sau expirat. 3. Backend-ul returnează eroare la recepție. |
| TA03 | Validarea credentialului de către backend pentru un utilizator major | 1. Se pregătește un credential valid cu atributul „is\_over\_18=true”.  2. Se trimite credentialul prin wallet.  3. Se verifică statusul tranzacției în backend. | Statusul tranzacției este actualizat la ACCEPTED. | 1. Verificarea semnăturii SD-JWT eșuează.  2. Validarea atributului returnează false din eroare de parsare.  3. Probleme la actualizarea statusului în DB. |
| TA04 | Validarea credentialului de către backend pentru un utilizator minor | 1. Se pregătește un credential valid cu atributul „is\_over\_18=false”.  2. Se trimite credentialul prin wallet.  3. Se verifică statusul tranzacției în backend. | Statusul tranzacției este actualizat la DENIED. | 1. Validarea atributului e interpretată greșit.  2. Statusul tranzacției nu este actualizat corect.  3. Backend-ul returnează eroare la salvare. |
| TA05 | Afișarea mesajelor corespunzătoare în aplicația mobile | 1. Se parcurg testele TA03 și TA04.  2. Se observă mesajul afișat în app pentru ACCEPTED.  3. Se observă mesajul afișat pentru DENIED. | Mesajele de succes și respingere sunt afișate corect și rapid. | 1. Mesajele nu se afișează din cauza erorii de parsing răspuns. 2. Layout-ul aplicației nu actualizează ecranul. |

Tabelul 5 Testele de acceptanță

### Raport de testare

Raportul de testare are rolul de a demonstra calitatea soluției dezvoltate, confirmând că aceasta respectă cerințele funcționale și tehnice definite în etapa de proiectare. Prin testele unitare, de integrare, de performanță și de acceptanță, au fost verificate atât corectitudinea implementării fiecărui modul în parte, cât și comportamentul întregului sistem în scenarii reale de utilizare.

1. **Testarea unitară**

Pentru testarea unitară, fiecare metodă implementată a fost verificată individual, izolându-se dependențele externe prin utilizarea de mock-uri acolo unde a fost necesar. Astfel, logica fiecărui modul a fost evaluată strict în funcție de valorile de intrare și rezultatele așteptate, confirmând corectitudinea implementării înainte de integrarea în fluxurile complete ale aplicației.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. Crt.** | **Status** | **Rezultat** | **Observații** |
| TU01 | PASSED | PresentationDefinitionTO este generat cu ID unic corect. | Fișierul JSON a fost încărcat fără erori, iar ID-ul UUID a fost generat automat la încărcare. |
| TU02 | PASSED | ID-urile generate sunt unice și conforme cu formatul UUID. | S-au generat mai multe ID-uri și toate au fost unice și în format valid UUID. |
| TU03 | PASSED | Tranzacția este salvată cu status PENDING și răspunsul include ID-ul și requestUri-ul. | S-a verificat salvarea tranzacției în repository și structura răspunsului InitTransactionResponse. |
| TU04 | PASSED | JWT-ul generat este valid, semnat corect și conține claims-urile necesare. | S-a verificat semnătura ECDSA și payload-ul JWT generat de CreateJarService. |
| TU05 | PASSED | Cheia efemeră ECKey este generată corect cu parametrii de criptare setați. | Obiectul returnat a fost de tip ECKey și a inclus crv, x, y și kid corespunzătoare. |
| TU06 | PASSED | Semnătura SD-JWT este validată corect (valid) și excepția este aruncată corespunzător (invalid). | Pentru vp\_token valid credentialul a fost acceptat, iar pentru vp\_token invalid s-a aruncat excepția de semnătură invalidă. |
| TU07 | PASSED | Hash-ul calculat din disclosure este găsit în sd\_hashes și validarea trece. | S-a verificat consistența hash-urilor calculate și lipsa diferențelor între disclosure și sd\_hashes. |
| TU08 | PASSED | Disclosure-ul este validat corect și atributul (ex. is\_over\_18) este determinat corect. | Funcția validateDisclosureAndCheckAge() a returnat rezultatele corecte pentru disclosure-urile testate. |
| TU09 | PASSED | Pentru credential valid returnează true, iar pentru revocat returnează false. | StatusListService a interpretat corect bitul asociat indexului testat. |
| TU10 | PASSED | Status list-ul comprimat este decompresat corect, fără pierderi de date. | Rezultatul decompress() a fost un array de biți valid, egal cu lista originală. |
| TU11 | PASSED | Dacă verificarea prin blob eșuează, fallback-ul IPFS este apelat și returnează statusul corect. | Testul a simulat indisponibilitatea blob-ului și a confirmat că metoda isRevokedViaIpfs() este apelată automat. |
| TU12 | PASSED | Pentru credential valid metoda returnează false (nerevocat), iar pentru credential revocat returnează true. | BlockchainRevocationChecker a confirmat corect statusul în ambele scenarii testate. |
| TU13 | PASSED | Statusul tranzacției este actualizat la ACCEPTED și persistat în baza de date. | EvidenceService a setat corect statusul ACCEPTED și modificarea a fost salvată în repository. |
| TU14 | PASSED | Statusul tranzacției este actualizat la DENIED și persistat în baza de date. | Similar TU13, dar cu status DENIED, confirmând logica de actualizare în funcție de rezultat. |
| TU15 | PASSED | Evenimentul de inițiere a tranzacției este logat corect. | Logarea inițierii prin logTransactionInitialized() a adăugat event-ul în lista tranzacției. |
| TU16 | PASSED | Evenimentul de răspuns al wallet-ului este logat corect în tranzacție. | logWalletResponsePosted() a adăugat payload-ul primit în lista de events și a actualizat timestamp-ul tranzacției. |
| TU17 | PASSED | Tranzacțiile PENDING mai vechi de 2 minute sunt șterse corect din baza de date. | TransactionCleanupService a selectat și șters tranzacțiile expirate, confirmând funcționalitatea de clean-up programat. |

Tabelul 6 Rezultatele testelor unitare

1. **Testarea de integrare**

Testarea de integrare a verificat modul în care modulele individuale ale aplicației interacționează între ele atunci când sunt combinate într-un flux complet. Această etapă de testare a urmărit să identifice problemele apărute la integrarea componentelor dezvoltate separat.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. Crt.** | **Status** | **Rezultat** | **Observații** |
| TI01 | PASSED | Aplicația mobile primește un request URI valid de la backend și afișează codul QR. | A fost verificată conexiunea HTTPS, formatul request URI și afișarea codului QR fără erori în aplicație. |
| TI02 | PASSED | Fluxul complet inițiere tranzacție – scanare QR – wallet – backend s-a finalizat cu succes, statusul tranzacției fiind actualizat corespunzător. | Toate componentele (mobile app, backend, EUDI Wallet) au comunicat conform așteptărilor, fără întreruperi în flux. |
| TI03 | PASSED | Tranzacția a fost salvată în PostgreSQL cu status PENDING și actualizată la ACCEPTED după validare. | Conexiunea la DB a fost stabilă, persistarea și actualizarea tranzacției s-au efectuat corect. |
| TI04 | PASSED | Backend-ul a acceptat token-ul JWT emis de Keycloak, permițând accesul la endpoint-ul protejat. | Configurația Keycloak a fost verificată, autentificarea și autorizarea funcționând fără erori. |
| TI05 | PASSED | Backend-ul a citit corect feature flag-ul din Unleash și a aplicat logica corespunzătoare în răspuns. | Flag-ul de test a fost detectat corect; backend-ul și-a schimbat comportamentul conform configurației Unleash. |
| TI06 | PASSED | Backend-ul a fost accesat securizat prin Cloudflare Tunnel, certificatul SSL fiind validat cu succes. | URL-ul Cloudflare a funcționat corespunzător, conexiunea HTTPS fiind stabilă și sigură. |

Tabelul 7 Rezultatele testelor de integrare

1. **Testarea de performanță**

Testarea de performanță a avut ca scop măsurarea timpilor de răspuns pentru fluxurile de validare, în diferite scenarii de configurare, pentru a evalua rapiditatea, stabilitatea și scalabilitatea soluției implementate.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. Crt.** | **Status** | **Timp mediu măsurat** | **Observații** |
| TP01 | PASSED | 35 ms | Timp excelent, cache-ul reduce semnificativ latența. |
| TP02 | PASSED | 247 ms | Creștere a timpului din cauza fetch-ului inițial. |
| TP03 | PASSED | 1844 ms | Timp ridicat, specific verificării blob. |
| TP04 | PASSED | 3550 ms | Fetch necachuit + verificare blob cresc latența. |
| TP05 | PASSED | 339 ms | IPFS a răspuns rapid în scenariul cache-uit. |
| TP06 | PASSED | 4200 ms | IPFS necachuit implică latență mare. |
| TP07 | PASSED | 1885 ms | Similar TP03, combină verificările. |
| TP08 | PASSED | 3908 ms | Timp ridicat, dar sub fallback-ul complet IPFS. |
| TP09 | PASSED | 356 ms | IPFS cache-uit rapid, similar cu TP05. |
| TP10 | PASSED | 5410 ms | Cel mai mare timp măsurat, include fetch complet IPFS + status list necachuită. |

Tabelul 8 Rezultatele testelor de performanță

1. **Testare de acceptanță**

Testarea de acceptanță a avut rolul de a confirma că soluția implementată respectă cerințele funcționale din perspectiva utilizatorului final și oferă o experiență completă, stabilă și corectă în scenariile reale de utilizare. Prin aceste teste s-a verificat că aplicația răspunde așteptărilor și poate fi acceptată pentru utilizare în producție.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. Crt.** | **Status** | **Rezultat** | **Observații** |
| TA01 | PASSED | Cod QR generat și afișat corect în aplicația mobile. | Aplicația a comunicat cu backend-ul și a afișat codul QR fără întârzieri. |
| TA02 | PASSED | Credentialul a fost trimis de wallet și primit de backend fără erori. | Wallet-ul a scanat QR-ul generat și a trimis credentialul, backend-ul l-a recepționat corect. |
| TA03 | PASSED | Statusul tranzacției actualizat la ACCEPTED pentru utilizator major. | Credential valid, backend-ul a confirmat majoratul și a salvat statusul corect. |
| TA04 | PASSED | Statusul tranzacției actualizat la DENIED pentru utilizator minor. | Backend-ul a validat corect atributul is\_over\_18=false și a respins credentialul. |
| TA05 | PASSED | Mesajele de succes și respingere afișate corect în aplicația mobilă. | Layout-ul a actualizat ecranele corespunzător, fără erori de parsing sau afișare. |

Tabelul 9 Rezultatele testelor de acceptanță

# Concluzii

## Sinteza principalelor idei din lucrare

Scopul lucrării a fost dezvoltarea unei soluții concrete care să faciliteze validarea atestărilor electronice de atribute, oferind un echilibru între securitate, eficiență și respectarea standardelor europene actuale. Soluția propusă integrează un backend robust, implementat în Java cu Spring Boot, alături de o aplicație mobile Android intuitivă, rezultând într-un sistem complet, capabil să fie utilizat imediat în scenarii reale.

Implementarea prezentată demonstrează cum procesele de validare a credențialelor, care din perspectiva utilizatorului se rezumă la scanarea unui cod QR și partajarea câtorva date, presupun în spate mecanisme complexe care asigură autentificarea emitentului, verificarea integrității și validității datelor, precum și protejarea confidențialității utilizatorului în fiecare etapă. Prin aceste funcționalități, soluția contribuie direct la creșterea încrederii în serviciile digitale și la reducerea birocrației în interacțiunile cu instituțiile publice și private.

Avantajul major al acestei lucrări constă în faptul că pune la dispoziție o implementare completă, modulară și testată, care poate fi preluată rapid de către dezvoltatori și organizații pentru a integra validarea atestărilor electronice în propriile sisteme, fără a mai fi nevoie ca aceștia să studieze în detaliu fiecare standard tehnic avansat implicat. Astfel, timpul de implementare este redus considerabil, riscurile asociate proiectelor complexe de identitate digitală scad, iar echipele pot adopta rapid soluții conforme cu reglementările europene, fără a investi resurse semnificative în înțelegerea profundă a protocoalelor.

Totodată, soluția prezentată dovedește aplicabilitatea practică a conceptelor teoretice studiate, prin integrarea directă cu portofelul digital EUDI Wallet, validat la nivel european, și implementarea unei metode suplimentare de verificare a revocării credentialelor, pe lângă mecanismele standardizate deja existente.

## Direcții pentru continuarea cercetării

Lucrarea realizată deschide mai multe direcții pentru extinderea și îmbunătățirea soluției propuse. O primă direcție importantă o reprezintă implementarea completă a mecanismului de verificare a trust list-urilor conform standardelor finale care vor fi publicate la nivel european. În prezent, implementarea este orientată pe structuri de test, însă integrarea directă cu trust list-urile oficiale EUDI va fi esențială pentru interoperabilitatea sistemului.

Totodată, un aspect ce poate fi îmbunătățit este optimizarea interogărilor efectuate către metoda blob și fallback IPFS. În timpul testelor au fost întâmpinate situații în care API-ul de blob a returnat erori, afectând astfel timpul total de răspuns. Optimizarea acestui flux ar putea crește preformanțele soluției.

O altă direcție importantă de cercetare ar putea fi explorarea unor metode noi de verificare a revocării credențialelor, care să elimine complexitatea fallback-urilor multiple și să reducă timpul total de validare.

Nu în ultimul rând, o direcție importantă pentru continuarea cercetării ar fi extinderea funcționalității aplicației mobile, astfel încât aceasta să poată stoca și gestiona credențialele utilizatorului, transformându-se practic într-un wallet digital complet, nu doar într-o aplicație care generează și afișează coduri QR. O astfel de dezvoltare ar permite utilizatorului să își administreze direct credențialele, să le prezinte în diverse contexte și să beneficieze de un flux complet de identitate digitală direct pe telefon.

# Bibliografie

[1] European Commission, *eIDAS Regulation*, available at https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation, accessed at 24.06.2025

[2] European Commission, *EU Digital Identity Wallet Home*, available at https://ec.europa.eu/digital-building-blocks/sites/display/EUDIGITALIDENTITYWALLET/, accessed at 24.06.2025

[3] eudi-wallet, *EU Digital Identity Wallet Reference Implementation*, available at https://github.com/eu-digital-identity-wallet/eudi-wallet-reference, accessed at 21.06.2025

[4] IETF, *Selective Disclosure for JWTs (SD-JWT)*, available at https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt/, accessed at 24.06.2025

[5] Digital Identity Wallet, *What are the 3 types of Electronic Attestations of Attributes (EAA)?*, available at https://www.digital-identity-wallet.eu/news/what-are-the-3-types-of-electronic-attestations-of-attributes-eaa/, accessed at 16.06.2025

[6] Talao, *EUDI Wallet: Understanding Credentials in eIDAS 2 – EAA, QEAA and PID*, available at https://www.talao.io/blog/eudi-wallet-understanding-credentials-in-eidas-2-eaa-qeaa-and-pid/, accessed at 16.06.2025

[7] European Commission, *The many use cases of the EU Digital Identity Wallet*, https://ec.europa.eu/digital-building-blocks/sites/display/EUDIGITALIDENTITYWALLET/The+many+use+cases+of+the+EU+Digital+Identity+Wallet, accessed at 16.06.2025

[8] AppViewX, *What Happens When a Certificate Chain of Trust Breaks?*, available at https://www.appviewx.com/blogs/what-happens-when-a-certificate-chain-of-trust-breaks/, accessed at 16.06.2025

[9] ThreatPost, *Google Drops Trust in Chinese Certificate Authority CNNIC*, available at https://threatpost.com/google-drops-trust-in-chinese-certificate-authority-cnnic/111974/, accessed at 17.06.2025

[10] Wikipedia, *Online Certificate Status Protocol*, available at https://en.wikipedia.org/wiki/Online\_Certificate\_Status\_Protocol, accessed at 17.06.2025

[11] Google Security Blog, *Announcing the first SHA1 collision*, available at https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html, accessed at 18.06.2025

[12] Cryptomathic, *Quantum Computing and Its Impact on Cryptography*, available at https://www.cryptomathic.com/blog/quantum-computing-and-its-impact-on-cryptography, accessed at 18.06.2025

[13] Adrian-Tudor Dumitrescu, *Failures of Public Key Infrastructure: 53 Year Survey, Delft University of Technology*, supervised by Johan Pouwelse

[14] QuSecure, *The Troubles with PKI: What Every Business Needs to Know*, available at https://www.qusecure.com/the-troubles-with-pki-what-every-business-needs-to-know, accessed at 18.06.2025

[15] Outspoken Media, *OCSP: The Pros and Cons*, available at https://outspokenmedia.com/https/ocsp/, accessed at 18.06.2025

[16] 1Kosmos, *Decentralizing Identity for Enhanced User Control*, available at https://www.1kosmos.com/identity-management/decentralizing-identity-for-enhanced-user-control/, accessed at 19.06.2025

[17] Identity Management Institute, *Digital Identity Wallet Benefits and Risks. Available* at https://identitymanagementinstitute.org/digital-identity-wallet-benefits-and-risks/, accessed at 19.06.2025

[18] Epicenter.works, *EU Digital Identity Reform: The Good, Bad & Ugly in the eIDAS Regulation*, available at https://epicenter.works/en/content/eu-digital-identity-reform-the-good-bad-ugly-in-the-eidas-regulation, accessed at 19.06.2025

[19] Wikipedia, *Privacy by Design*, available at https://en.wikipedia.org/wiki/Privacy\_by\_design, accessed at 20.06.2025

[20] W3C, *Verifiable Credentials Data Model v2.0*, available at https://www.w3.org/TR/vc-data-model/, accessed at 20.06.2025

[21] OpenID Foundation, *OpenID for Verifiable Presentations (OpenID4VP)*, available at https://openid.net/specs/openid-4-verifiable-presentations-1\_0.html, accessed at 20.06.2025

[22] European Commission, *European Digital Identity Wallet Architecture and Reference Framework*, Version 1.0.0, January 2023, available at https://digital-strategy.ec.europa.eu/en/library/european-digital-identity-wallet-architecture-and-reference-framework

[23] OpenID Foundation, *OpenID for Verifiable Credential Issuance (OpenID4VCI)*, available at https://openid.net/specs/openid-4-verifiable-credential-issuance-1\_0.html, accessed at 20.06.2025

[24] W3C, *Verifiable Credentials Bitstring Status List 2021*, available at https://www.w3.org/TR/vc-bitstring-status-list/, accessed at 22.06.2025

[25] Felix Hoops, Jonas Gebele, Florian Matthes, *CRSet: Private Non-Interactive Verifiable Credential Revocation*, Technical University of Munich, Munich, Germany

[26] iDAKTO, *European Digital Wallet: Understanding the Trust Model*, available at https://www.idakto.com/blog/eudi-wallet-what-are-the-mechanisms-and-guarantees-of-the-trust-model/, accessed at 23.06.2025

[27] European Commission, *EU Trusted Lists*, available at https://digital-strategy.ec.europa.eu/en/policies/eu-trusted-lists, accessed at 23.06.2025

[28] Talao, *Talao Identity Wallet and SSI Solutions*, available at https://www.talao.io/, accessed at 21.06.2025

[29] Walt.id, *Digital identity and wallet infrastructure*, available at https://walt.id/, accessed at 21.06.2025

[30] Dock, *Dock Wallet – Verifiable Credentials and Digital Identity Wallet*, available at https://www.dock.io/dock-wallet, accessed at 22.06.2025

[31] Trinsic, *Trinsic – APIs for Verifiable Credentials and Digital Identity*, available at https://trinsic.id/, accessed at 22.06.2025

[32] ZealiD, *ZealiD Wallet – Qualified Electronic Signatures and eID*, available at https://www.zealid.com/, accessed at 22.06.2025

[33] Google, *Android Developers – Build apps for Android*, available at https://developer.android.com/, accessed at 25.06.2025

[34] Oracle, *Java Platform, Standard Edition Documentation*, available at https://docs.oracle.com/javase/8/docs/, accessed at 25.06.2025

[35] Spring, *Spring Boot Reference Documentation*, available at https://spring.io/projects/spring-boot, accessed at 26.06.2025

[36] PostgreSQL Global Development Group, *PostgreSQL Documentation*, available at https://www.postgresql.org/docs/, accessed at 26.06.2025

[37] Keycloak, *Keycloak Documentation*, available at https://www.keycloak.org/documentation, accessed at 26.06.2025

[38] Unleash, *Unleash Open Source Feature Flag Management*, available at https://www.getunleash.io/, accessed at 27.06.2025

[39] Docker, *Docker Documentation*, available at https://docs.docker.com/, accessed at 27.06.2025

[40] Cloudflare, *Cloudflare Documentation*, available at https://developers.cloudflare.com/, accessed at 27.06.2025

1. https://medium.com/datafrens-sg/unerstanding-the-eidas-2-0-and-its-implication-for-individuals-privacy-and-data-protection-rights-df0ae62eaafa [↑](#footnote-ref-1)
2. https://www.criipto.com/blog/sd-jwt-based-verifiable-credentials [↑](#footnote-ref-2)
3. https://tsmatz.wordpress.com/2020/06/25/what-is-verifiable-credentials/ [↑](#footnote-ref-3)
4. https://www.intesigroup.com/en/news/eudi-arf/ [↑](#footnote-ref-4)
5. https://www.w3.org/TR/vc-bitstring-status-list/ [↑](#footnote-ref-5)
6. https://www.talao.io/talao-wallet/ [↑](#footnote-ref-6)
7. https://walt.id/white-paper/decentralized-identity-playbook [↑](#footnote-ref-7)
8. https://medium.com/beyond-the-crypto-horizon/time-to-research-dock-aa94158d2bd2 [↑](#footnote-ref-8)
9. https://trinsic.id/platform/ [↑](#footnote-ref-9)
10. https://www.zealid.com/en/identity-wallet [↑](#footnote-ref-10)
11. https://www.geeksforgeeks.org/introduction-to-spring-boot/ [↑](#footnote-ref-11)
12. https://www.scaler.com/topics/postgres-architecture/ [↑](#footnote-ref-12)
13. https://www.hcltech.com/blogs/keycloak-multitenancy [↑](#footnote-ref-13)
14. https://docs.getunleash.io/feature-flag-tutorials/java [↑](#footnote-ref-14)
15. https://medium.com/@ravipatel.it/understanding-docker-architecture-a-comprehensive-guide-5ce9129df1a4 [↑](#footnote-ref-15)
16. https://developers.cloudflare.com/cloudflare-one/connections/connect-networks/ [↑](#footnote-ref-16)