Aegis

Aplicație pentru păstrarea sigură a documentelor



Aplicație dezvoltată de echipa Cretzu Cafe:

* Student caporal CORDUNEANU Arian-Andrei
* Student caporal IVAN Florentin-Marian

Cuprins

[Capitolul 1. Introducere 3](#_Toc193639971)

[1.1. Scopul proiectului 3](#_Toc193639972)

[1.2. Lista definițiilor 3](#_Toc193639973)

[1.3. Lista referințelor 3](#_Toc193639974)

[1.4. Structura DCS 3](#_Toc193639975)

[Capitolul 2. Descrierea generală a produsului software 3](#_Toc193639976)

[2.1. Descrierea generală 3](#_Toc193639977)

[2.2. Platforma HW/SW 3](#_Toc193639978)

[2.3. Constrângeri 3](#_Toc193639979)

[Capitolul 3. Cerințe specifice 3](#_Toc193639980)

[3.1. Cerințe funcționale 3](#_Toc193639981)

[3.2. Cerințe ne-funcționale 3](#_Toc193639982)

[Capitolul 4. Anexe 3](#_Toc193639983)

[Descriere model logic 3](#_Toc193639984)

[Diagrama claselor 3](#_Toc193639985)

# Capitolul 1. Introducere

## Scopul proiectului

Această aplicație este dezvoltată pentru a stoca într-un mod sigur documentele utilizatorilor prin utilizarea unor metode de criptare și partajare securizată. Aplicația își propune să asigure un nivel de confidențialitate, integritate și accesibilitate pentru utilizatorii săi.

Conexiunea dintre client și server va fi privată, iar transferul de date se va realiza prin intermediul unor canale de comunicație securizate utilizând protocoale proprii de criptare cu cifru XOR. Acest lucru protejează datele utilizatorilor împotriva interceptării și asigură confidențialitatea.

Documentele vor fi criptate local pe dispozitivul utilizatorului înainte de a fi încărcate pe server. Criptarea va fi realizată printr-o cheie individual pentru fiecare user, garantând astfel o protecție personalizată și fiabilă.

Aplicația permite partajarea securizată a fișierelor, oferind utilizatorilor posibilitatea de a selecta contactele dorite pentru partajare. Sistemul gestionează cu ușurință procesul de partajare, oferind o experiență simplă și eficientă.

Autentificarea utilizatorilor se va realiza pe baza unui sistem de nume de utilizator și parolă, unde parola si numele de utilizator va fi stocată local sub formă de hash pentru a preveni accesul neautorizat.

De asemenea, aplicația include funcționalități de jurnalizare, înregistrând toate acțiunile efectuate la nivelul clientului și serverului pentru a asigura trasabilitatea și identificarea eventualelor probleme sau erori.

## Lista definițiilor

Cheie simetrică: tipul de cheie utilizat atât pentru criptare, cât și pentru decriptare. Această cheie este împărțită între expeditor și destinatar

Cheie publică/privată: sunt folosite în sistemele de criptate asimetrice unde cheia publica este utilizată pentru criptare, iar cheia privată pentru decriptare (cheia privată fiind strict confidențială)

Hash: funcție matematică care transformă un set de date într-un șir unic de caractere, utilizat pentur verificarea integrității sau ascunderea datelor

## Lista referințelor

Work in progress

## Structura DCS

Capitolul 1 – Introducere: oferă o imagine de ansamblu asupra scopului aplicației, termenilor tehnici și referințelor relevante utilizate în dezvoltarea proiectului. Oferă contextul și terminologia necesară

Capitolul 2 – Descrierea generală a produsului software: se concentrează pe o descriere detaliată a aplicației, a platformei hardware/software necesare și a eventualelor constrângeri de proiectare. În plus aici se găsesc detalii despre mediul operațional și specificațiile platformei

Capitolul 3 – Cerințe specifice: este cel mai detaliat capitol, incluzând cerințele funcționale și cerințele nefuncționale.

Capitolul 4 – Anexe: oferă suport grafic și documentație suplimentară pentru cerințele definite. Include modele logice, diagrame ale claselor, fluxuri de date și alte elemente grafice relevante pentru implementare

# Capitolul 2. Descrierea generală a produsului software

## 2.1. Descrierea generală

Aplicația Aegis este o soluție software concepută pentru stocarea și partajarea sigură a documentelor utilizatorilor. Scopul său principal este de a asigura protecția datelor personale și confidențialitatea informațiilor prin utilizarea unor tehnologii avansate de criptare, autentificare și comunicare securizată.

Sistemul este bazat pe un model arhitectural client-server și include următoarele funcționalități principale:

**Client:**

* Utilizează interfața grafică pentru interacțiunea directă cu utilizatorii.
* Realizează criptarea fișierelor local prin generarea unei chei simetrice individuale pentru fiecare fișier.
* Stochează local cheile simetrice, utilizând hash-uri pentru anonimizare (opțional).
* Trimite fișierele criptate către server prin conexiuni securizate HTTPS/TLS.
* Primește de la server cheile publice ale altor utilizatori pentru partajare.

**Server:**

* Găzduiește fișierele criptate primite de la client.
* Stochează cheile simetrice, criptate folosind cheia publică a utilizatorului.
* Menține o listă centralizată cu utilizatorii înregistrați pentru facilitarea partajării fișierelor.
* Notifică destinatarii atunci când un fișier a fost partajat.
* Asigură jurnalizarea tuturor acțiunilor efectuate de utilizatori.

## 2.2. Platforma HW/SW

Proiectul se desfășoară pe două laptopuri conectate la aceeași rețea locală (LAN), fiecare rulând câte o componentă a aplicației: serverul și clientul. Sistemele de operare utilizate sunt Windows 10/11, iar limbajul de programare este C++, împreună cu biblioteca Qt pentru interfața grafică a clientului. Serverul utilizează biblioteca ODBC pentru a comunica cu o bază de date SQL Server.

Componentele software includ:

* **Server**: aplicație C++ nativă, cu logare, gestionare fișiere și utilizatori, criptare și comunicație socket.
* **Client**: aplicație Qt cu interfață grafică, ce permite logarea, încărcarea și descărcarea de fișiere.

## 2.3. Constrângeri

* Aplicația funcționează doar în cadrul aceleiași rețele locale. Conectarea de la distanță este posibilă doar folosind soluții terțe (ex. Hamachi).
* Serverul trebuie pornit înaintea clientului pentru a stabili conexiunea.
* Porturile de comunicație trebuie să fie libere și neblocate de firewall sau antivirus.
* Limitări de dimensiune a fișierelor impuse de bufferul de transmisie socket și de limita bazei de date.
* Lipsa criptării avansate pentru transmisie (opțional de extins cu TLS/SSL).

# Capitolul 3. Cerințe specifice

## 3.1. Cerințe funcționale

* Autentificarea utilizatorilor pe baza unor credențiale (nume utilizator + parolă).
* Adăugarea, ștergerea și modificarea conturilor de utilizator.
* Încărcarea fișierelor în baza de date.
* Listarea fișierelor încărcate de utilizator.
* Descărcarea fișierelor de către utilizatorul autenticat.
* Logarea acțiunilor în fișier de jurnal (log).

## 3.2. Cerințe ne-funcționale

* Sistemul trebuie să fie robust la erori de conexiune și să afișeze mesaje informative.
* Timpul de răspuns între cererea clientului și răspunsul serverului nu trebuie să depășească 1 secundă pentru fișiere de dimensiuni mici.
* Datele utilizatorului trebuie protejate în baza de date (parolele sunt salvate criptat).
* Arhitectura modulară permite extinderea ușoară a aplicației (ex. adăugare funcționalitate de căutare).

# Capitolul 4. Anexe

## Descriere model logic

Baza de date conține tabelele Utilizatori și UserFiles. Relația este de tip 1:N, un utilizator poate avea mai multe fișiere. Cheile primare și externe sunt configurate pentru a asigura integritatea datelor.

Tabelul Utilizatori conține:

* ID – identificator unic;
* Username – username-ul hash-uit;
* Password – parola hash-uita;

Tabelul UserFiles conține:

* FileID – identificator unic;
* UserID – referință către utilizator;
* FileName – numele fișierului;

## Diagrama claselor

**Server:**

* DatabaseManagerAccounts – gestionează conexiunea la baza de date și operațiile SQL (login, adăugare, ștergere fișiere etc.);
* FileManager – gestionează salvarea/încărcarea/stergerea fișierelor local pe disc;
* Logger – scrie acțiunile importante într-un fișier de log;
* Retea – implementează protocolul de comunicație socket cu clientul, precum si handle-ul clientului;
* UserManager – gestionează parsarea protocolului trimis de client pentru cererea pe care o are.

**Client:**

* Network.h – oferă funcționalități de trimitere/recepționare prin socket TCP;
* mainwindow.ui – interfața grafică, cu funcții pentru autentificare și operații pe fișiere.