



**UNIVERSITATEA
TEHNICĂ
DIN CLUJ-NAPOCA**

Blood Donation System

Inginerie Software

Autor: Mirisan Octavian

Grupa: 30231

FACULTATEA DE AUTOMATICA
SI CALCULATOARE

Ianuarie 2025

Cuprins

1	Enuntul problemei	2
1.1	Descriere	2
1.2	Donator	2
1.3	Asistent	2
1.4	Administrator	2
1.5	Spectator	2
2	Instrumentele utilizate	3
2.1	Frontend (client)	3
2.2	Backend (server)	3
2.3	Baza de date	3
3	Diagramele UML	4
3.1	Diagrama cazurilor de utilizare (Use Case)	4
3.2	Diagrama entitate-relatie	5
3.3	Diagrame de activitati	5
3.4	Diagrame de pachete	15
3.5	Diagrame de clase	16
3.6	Diagrama de componente	17
3.7	Diagrama de dezvoltare	18
4	Descrierea aplicatiei	19
5	Concluzie si posibilitati de dezvoltare ulterioara	22
5.1	Justificarea alegerii: website sau desktop app	22
5.2	Concluzie	22
5.3	Posibilitati de dezvoltare ulterioara a aplicatiei client-server	22

1 Enuntul problemei

1.1 Descriere

Proiectul presupune dezvoltarea unei aplicatii destinate gestionarii donatiilor de sange intr-un spital. Aplicatia va fi utilizata de catre patru tipuri de utilizatori: Donator, Asistent Medical, Administrator de sistem si Spectator. Fiecare tip de utilizator va avea acces la un set specific de functionalitati, descrise mai jos. Aplicatia va gestiona intregul proces de donare, de la programarea donatiei, verificarea eligibilitatii, procesarea colectiilor de sange si gestionarea stocurilor de sange.

1.2 Donator

Donatorul este responsabil pentru donarea de sange. Principalele functionalitati ale donatorului sunt:

- Vizualizare date personale (nume, email, CNP etc.)
- Programare donatie de sange
- Completare formular de eligibilitate (autoevaluare a starii de sanatate si grupa sanguina)
- Vizualizare istoricul donatiilor
- Anularea sau reprogramarea unei donatii

1.3 Asistent

Asistentul medical este responsabil pentru gestionarea donatiilor in sistem. Acesta are urmatoarele functionalitati:

- Procesarea donatiilor (inregistrarea si gestionarea donatiilor de sange)
- Validarea formularelor completate de catre potentialii donatori
- Generarea si exportarea rapoartelor despre donatiile efectuate
- Vizualizarea istoricului donatiilor precum si a statisticilor legate de acestea (ce cantitate din fiecare grupa de sange exista)

1.4 Administrator

Administratorul de sistem este responsabil pentru gestionarea platformei si monitorizarea intregii activitati. Functionalitatile administratorului includ:

- Gestionarea donatorilor si a personalului medical (operatiuni CRUD pentru asistenti si donatori)
- Monitorizarea activitatilor (vizualizarea donatiilor, a rapoartelor generate, a activitatilor din cadrul platformei)
- Vizualizarea stocurilor de sange

1.5 Spectator

Spectatorul are acces la aplicatie fara a necesita crearea unui cont sau autentificare. Functionalitatile acestuia includ:

- Vizualizarea informatiilor despre procesul de donare (FAQ, beneficii ale donarii de sange, criterii de eligibilitate)
- Accesarea materialelor educative despre donarea de sange si impactul acesteia asupra comunitatii

2 Instrumentele utilizate

2.1 Frontend (client)

Partea de frontend a aplicatiei este responsabila pentru interfata utilizatorului si pentru interactiunea acestuia cu aplicatia. Instrumentele utilizate pentru partea de frontend includ:

- **HTML:** Limbajul de marcare utilizat pentru crearea structurii paginilor web. HTML definește elementele de baza ale unei pagini web, precum titluri, paragrafe, imagini, butoane, formulare etc. Este esential pentru crearea continutului vizibil al aplicatiei.
- **CSS:** Limbajul folosit pentru a stiliza paginile HTML. CSS permite personalizarea aspectului paginii, inclusiv culori, fonturi, dimensiuni si pozitionare. Acesta asigura ca aplicatia sa fie atractiva si usor de utilizat.
- **JavaScript:** Limbajul de programare utilizat pentru a adauga interactiuni dinamice pe paginile web. JavaScript este folosit pentru a manipula elementele HTML si CSS, pentru a valida formularele sau pentru a trimite cereri asincrone catre server fara a reîncărca pagina.
- **Bootstrap:** Un framework CSS open-source care ofera un set de instrumente predefinite pentru crearea de pagini web responsive. Bootstrap faciliteaza implementarea unui design modern si adaptabil la diferite dispozitive (desktop, tablete, telefoane mobile). Acesta include componente precum butoane, formulare, bare de navigare si multe altele, reducand semnificativ timpul necesar pentru dezvoltarea interfetei.

2.2 Backend (server)

Partea de backend a aplicatiei este responsabila pentru procesarea cererilor utilizatorului si manipularea datelor. Instrumentele utilizate pentru backend sunt:

- **Python:** Limbajul de programare folosit pentru dezvoltarea aplicatiei. Python este apreciat pentru sintaxa sa clara si usor de inteles, ceea ce il face potrivit atat pentru incepatori, cat si pentru dezvoltatori experimentati. In acest proiect, Python este folosit pentru a implementa logica de server si pentru a interactiona cu baza de date.
- **Flask:** Un framework web micro pentru Python, care permite dezvoltarea rapida si usoara a aplicatiilor web. Flask este flexibil si minimalistic, oferind doar instrumentele esentiale pentru a crea aplicatii web. Acesta permite gestionarea rutelor, manipularea cererilor HTTP, procesarea datelor si interogarea bazei de date, toate fiind realizate intr-un mod eficient si usor de inteles.

2.3 Baza de date

Pentru stocarea si gestionarea datelor aplicatiei, am utilizat urmatoarele tehnologii:

- **SQLite:** Un sistem de gestionare a bazelor de date relationale (RDBMS) care functioneaza pe baza unui singur fisier. SQLite este usor de utilizat, nu necesita un server de baze de date separat si este ideal pentru aplicatii de dimensiuni mici si medii. In acest proiect, SQLite este folosit pentru a stoca informatii despre donatori, donatii, rapoarte, utilizatori si alte date esentiale ale aplicatiei. Integrarea cu Flask este realizata simplu prin intermediul extensiei Flask-SQLAlchemy.

3 Diagramele UML

3.1 Diagrama cazurilor de utilizare (Use Case)

Diagrama cazurilor de utilizare descrie interacțiunile dintre utilizatori (actori) și sistem, evidențiind funcționalitățile principale ale aplicației și modul în care utilizatorii pot interacționa cu aceasta.

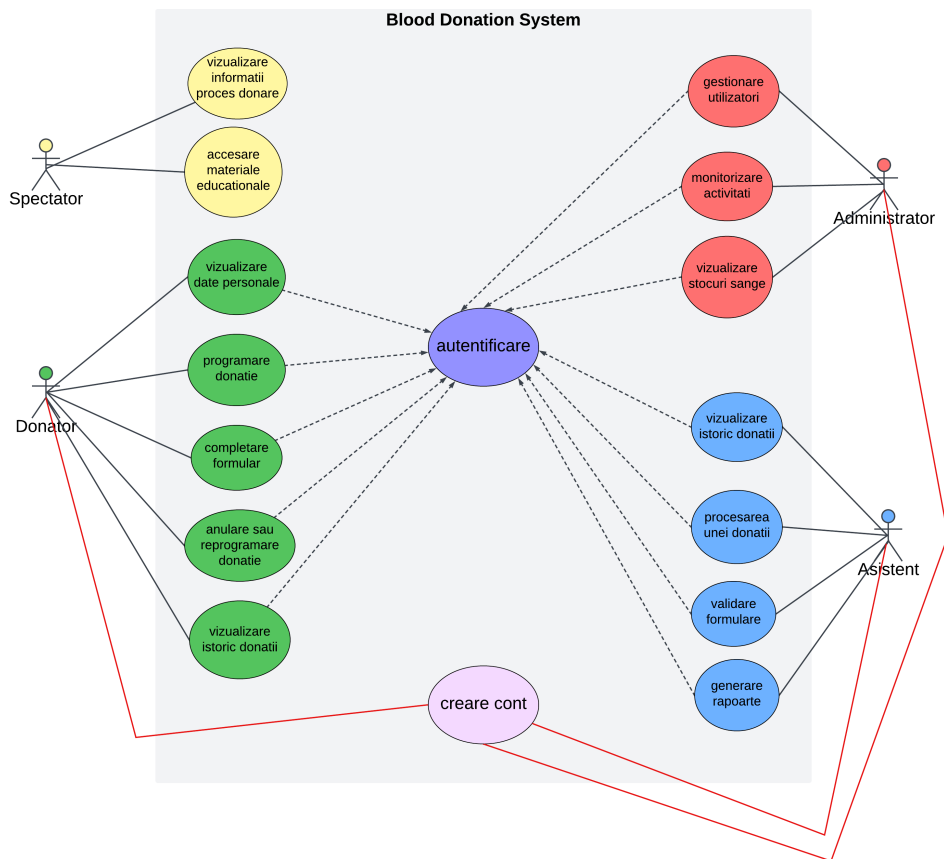


Figura 1: Diagrama cazurilor de utilizare a aplicației.

3.2 Diagrama entitate-relatie

Diagrama entitate-relatie (ERD) ilustreaza structura bazei de date, indicand entitatile (de exemplu, Donor, Asistent, Report) si relatiile dintre acestea, ajutand la intelegerea organizarii si legaturilor dintre datele sistemului.

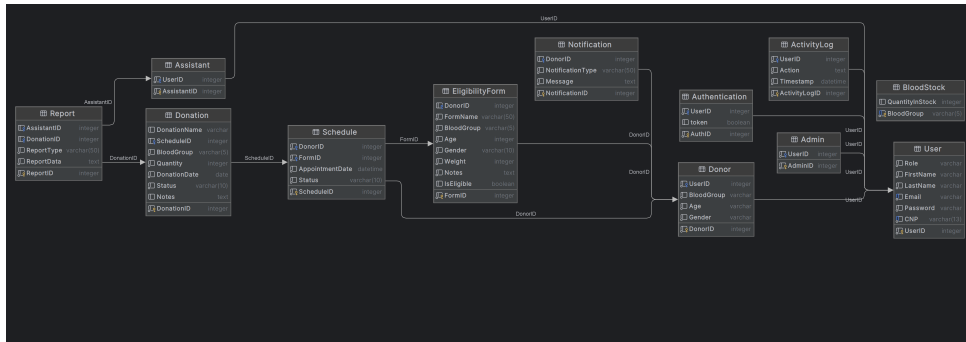


Figura 2: Diagrama entitate-relatie a aplicatiei.

3.3 Diagrame de activitati

Diagramele de activitati descriu fluxurile de lucru (workflows) sau procesele care au loc in cadrul aplicatiei, ilustrand pasii si actiunile care sunt realizate intr-o secventa specifica.

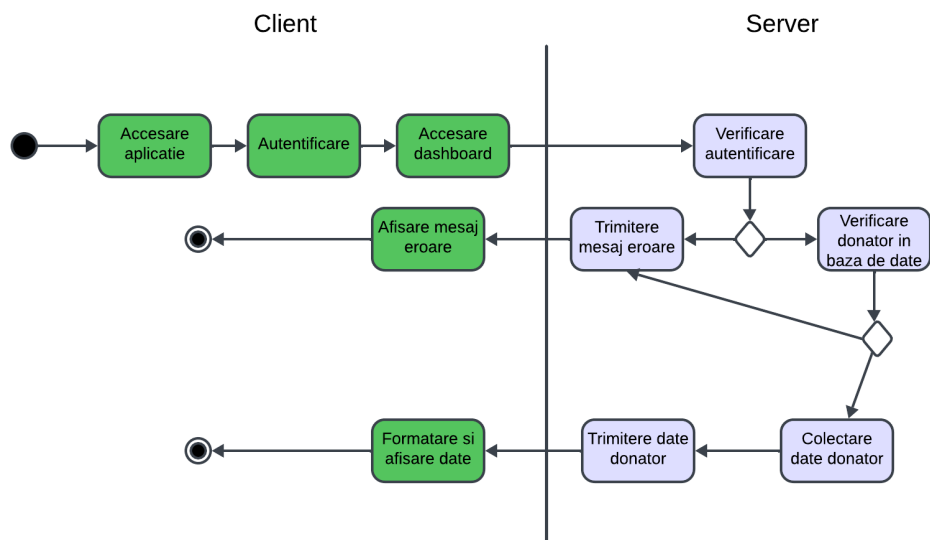


Figura 3: Donator - vizualizare date personale.

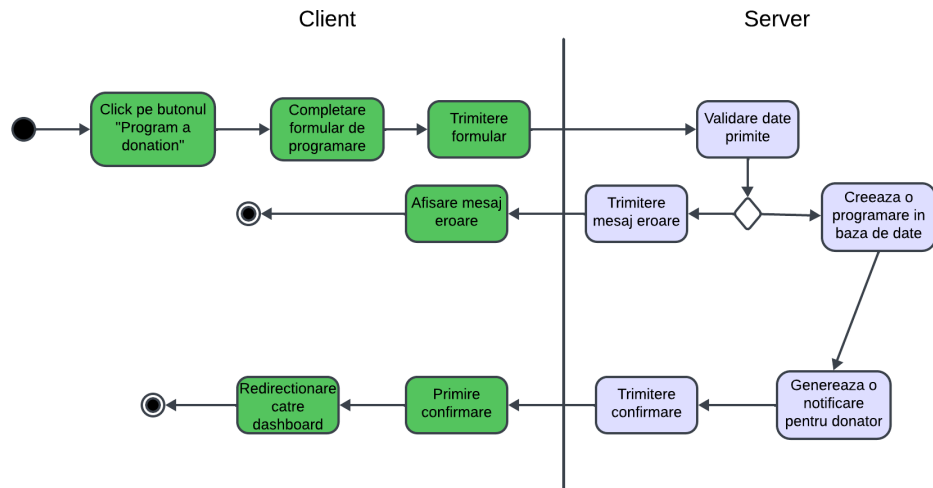


Figura 4: Donator - programare donatie.

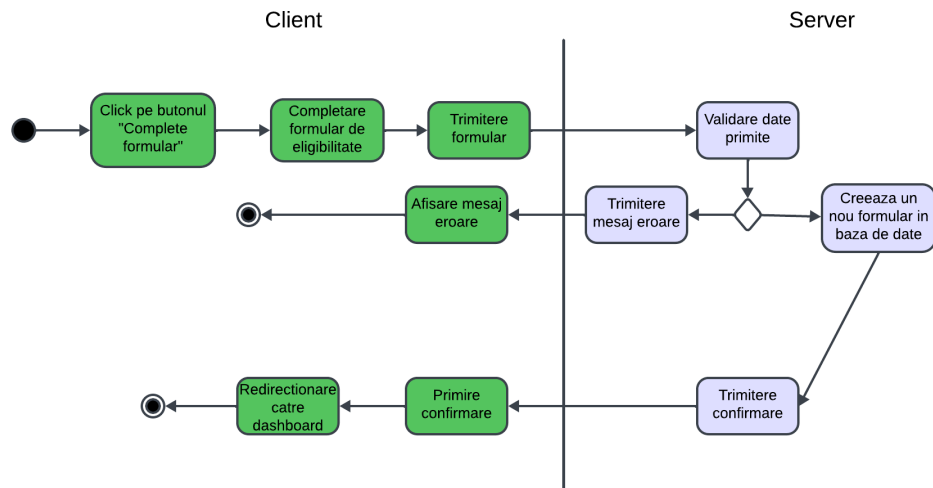


Figura 5: Donator - completare formular.

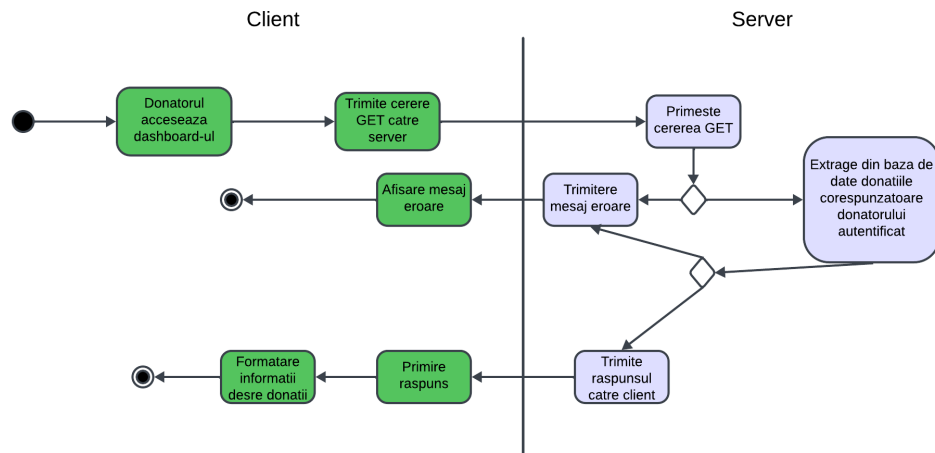


Figura 6: Donator - vizualizare istoric donatii.

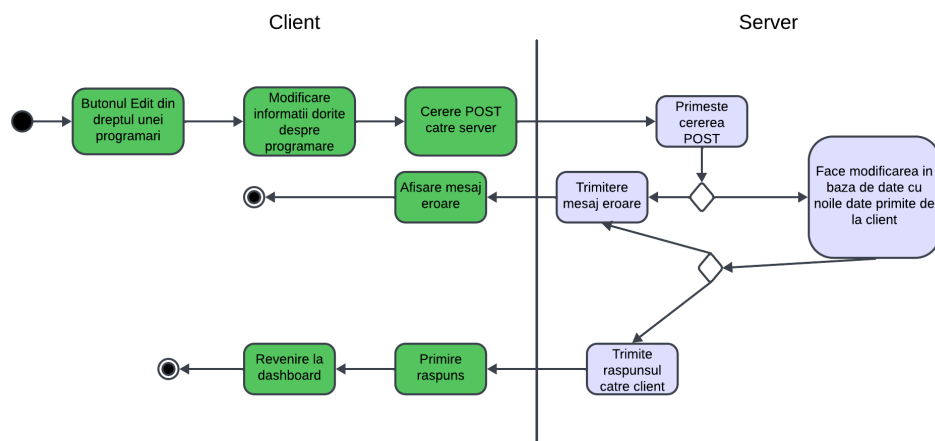


Figura 7: Donator - reprogramarea unei donatii.

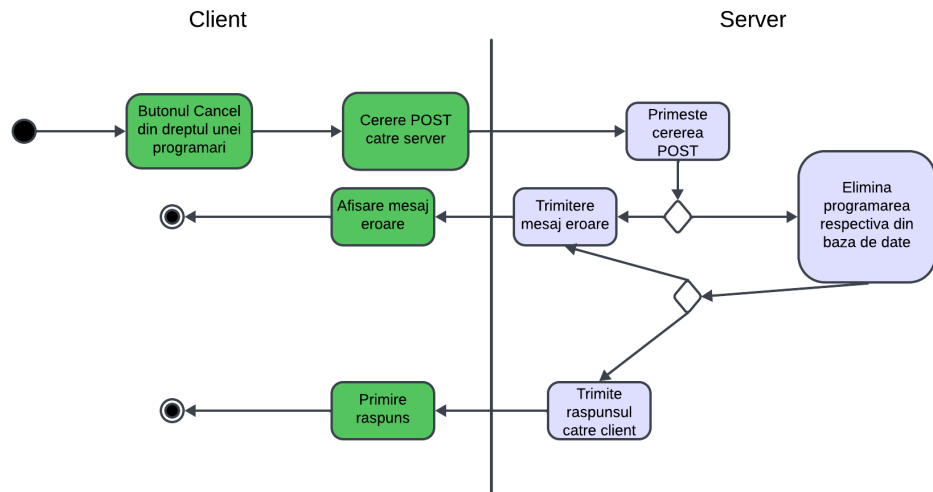


Figura 8: Donator - anularea unei programari.

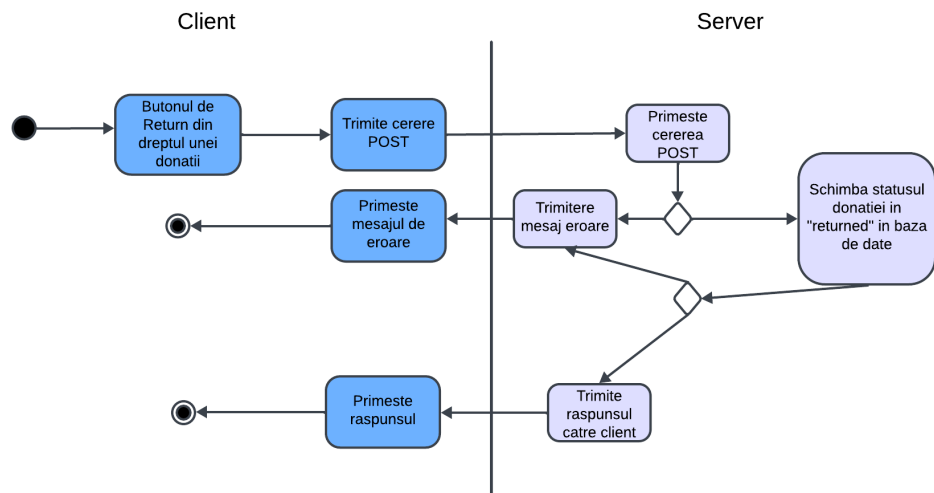


Figura 9: Asistent - returnarea unei donari.

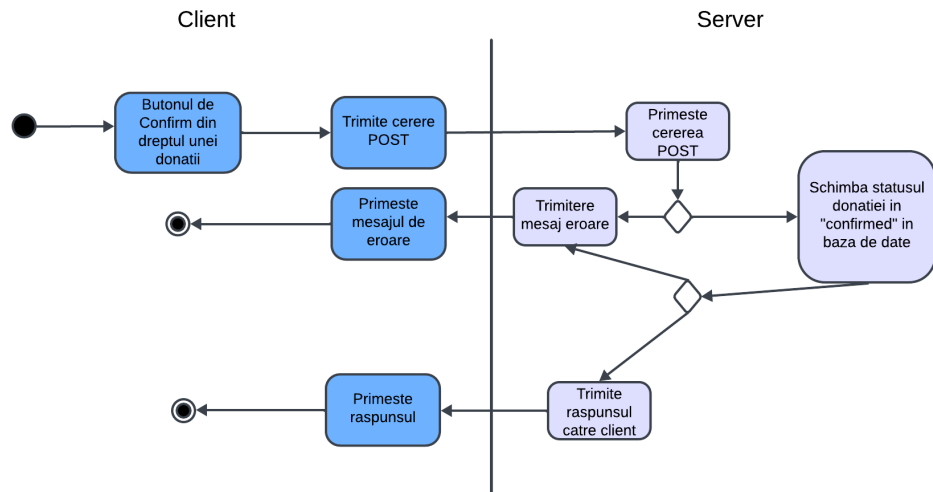


Figura 10: Asistent - confirmarea unei donari.

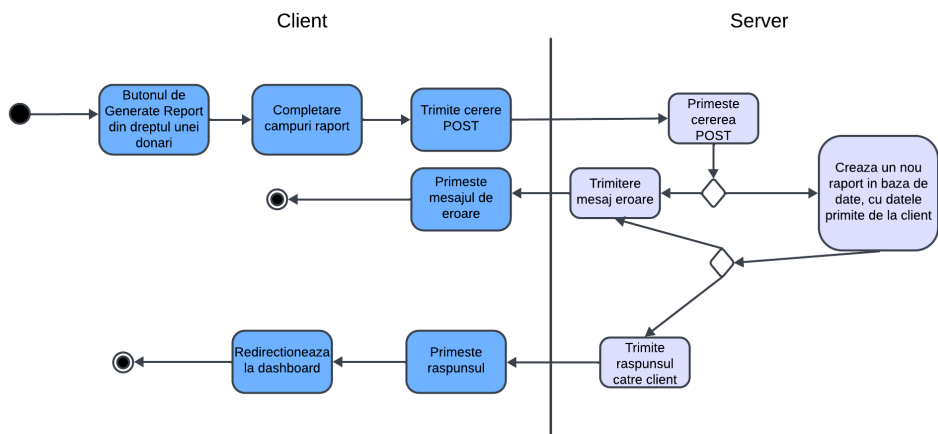


Figura 11: Asistent - generarea unui raport pentru o donare.

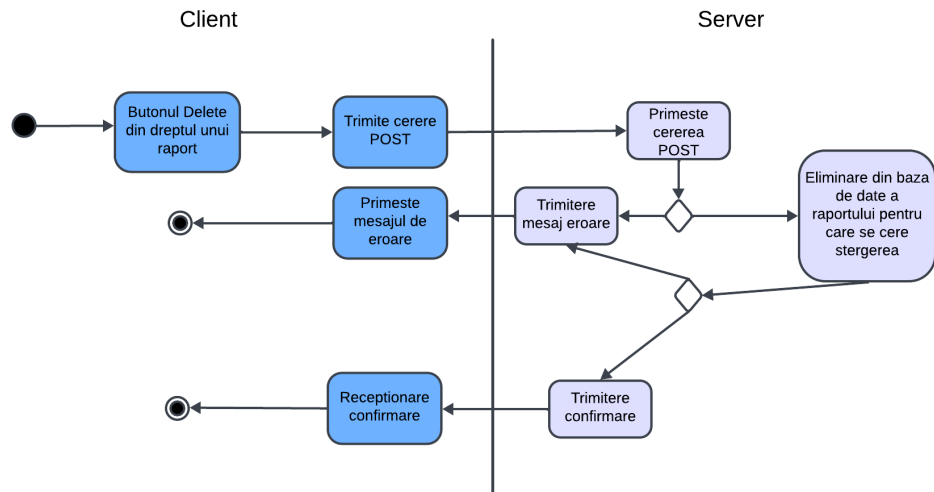


Figura 12: Asistent - stergerea unui raport existent.

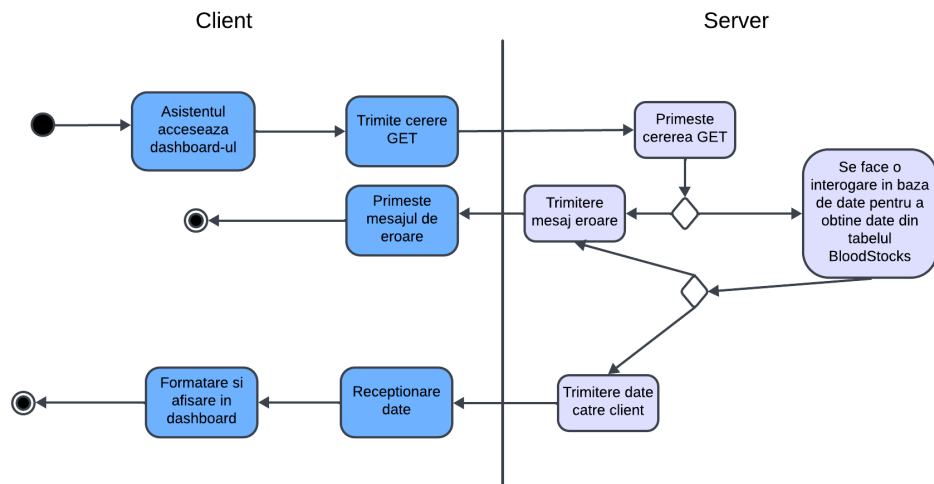
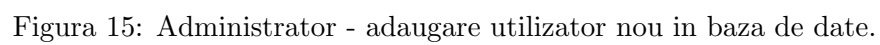


Figura 13: Asistent - vizualizare stocuri si statistici legate de donari.



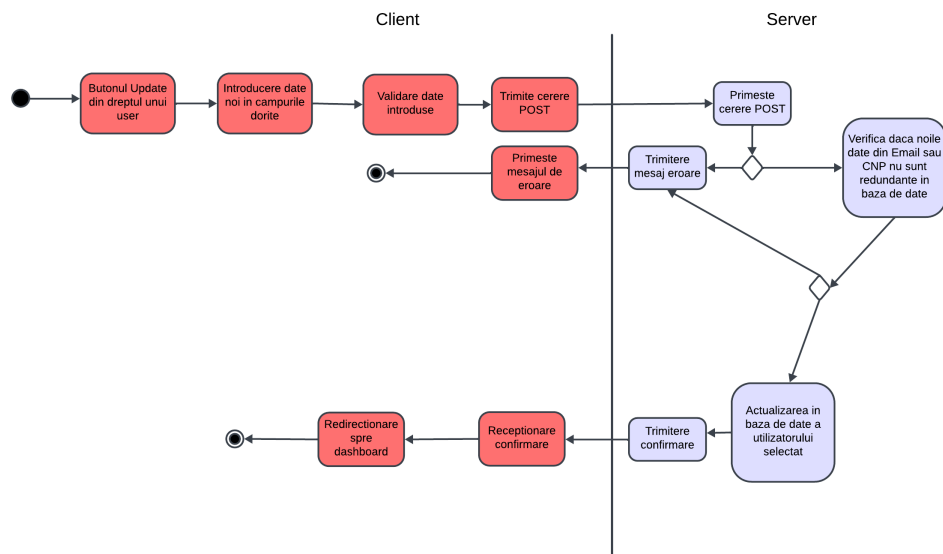


Figura 16: Administrator - actualizare utilizator in baza de date.

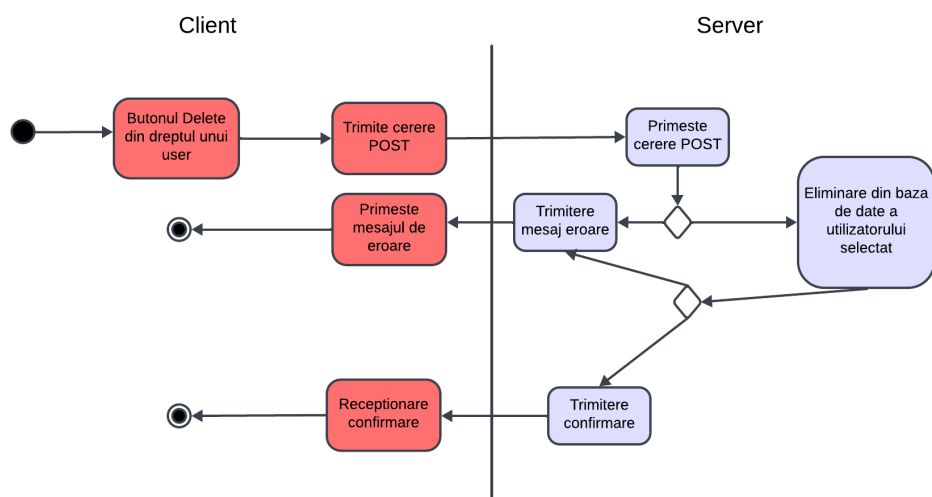


Figura 17: Administrator - eliminare utilizator din baza de date.

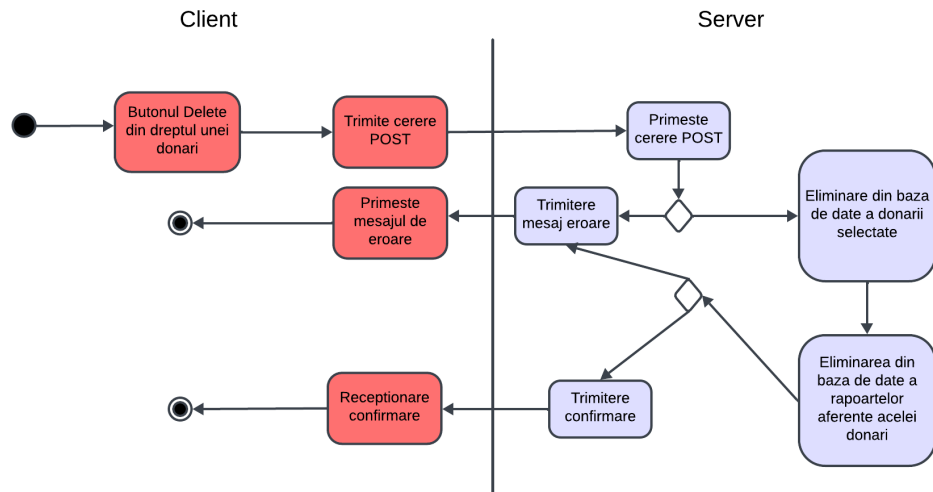


Figura 18: Administrator - eliminare donatie din baza de date.

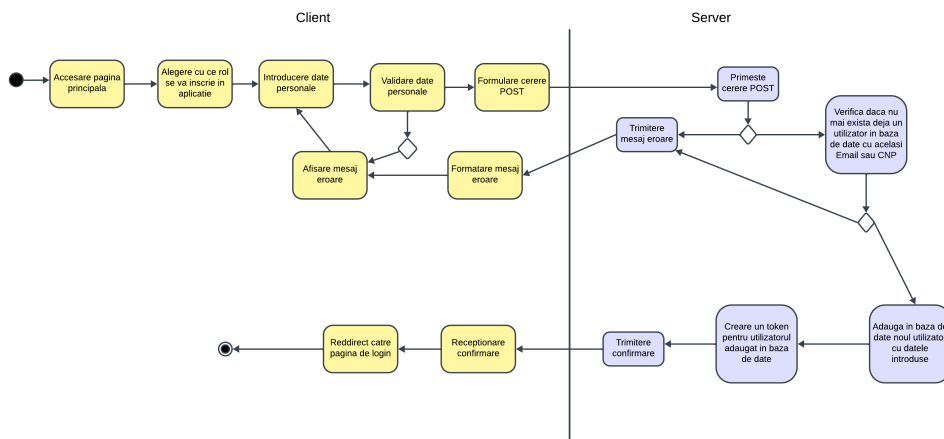


Figura 19: Utilizator - operatia de signup.

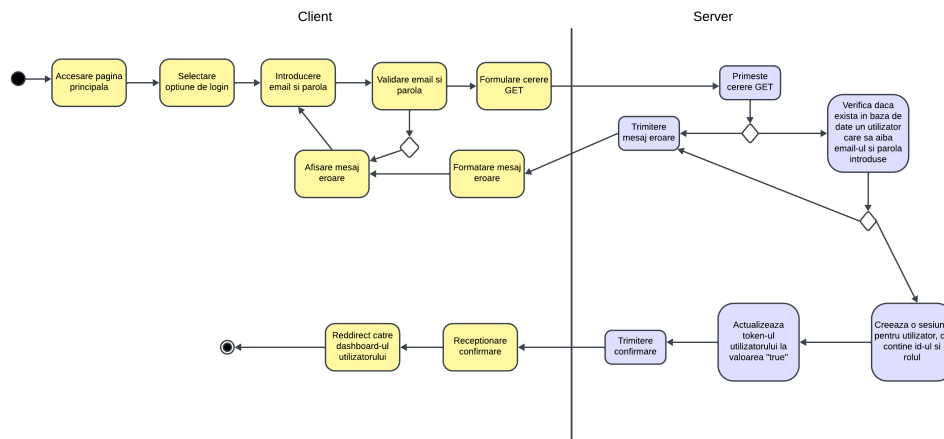


Figura 20: Utilizator - operatia de login (autentificare).

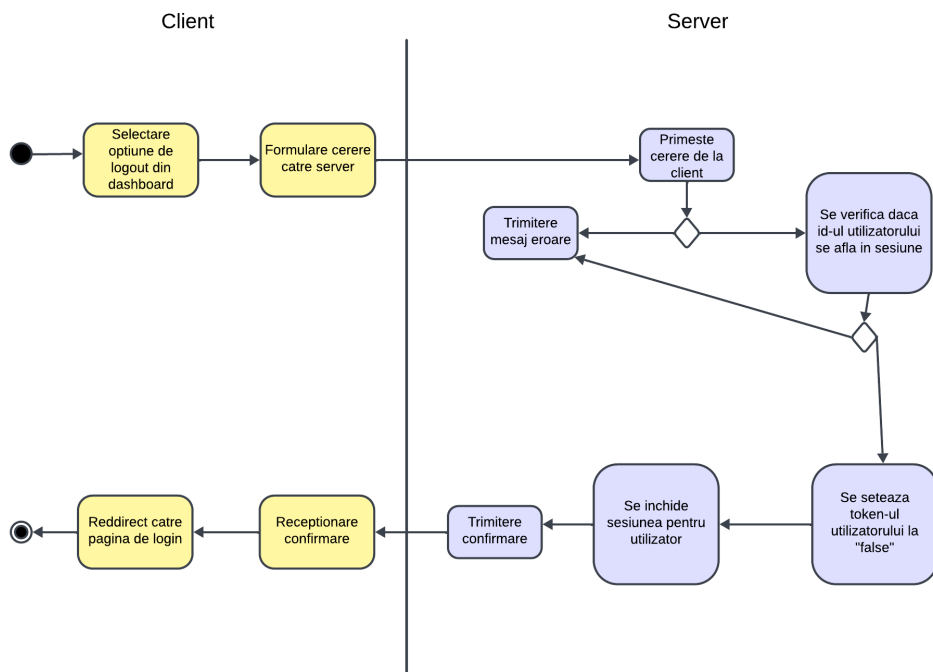


Figura 21: Utilizator - operatia de logout.

3.4 Diagrame de pachete

Diagramele de pachete arata modul in care componentele sau modulele sistemului sunt grupate intr-o structura ierarhica, facilitand organizarea si intelegerea dependintelor dintre diferitele parti ale aplicatiei.

Legat de partea de Server, exista 3 pachete: Controllers care contine rutele aplicatiei web(logica backend-ului se afla aici). Models unde sunt definite tabelele asa cum exista in baza de date a aplicatiei. Main este pachetul in care sunt definite fisierele ce stau la baza aplicatiei Flask: gestionarea rutelor, crearea bazei de date, crearea serverului.

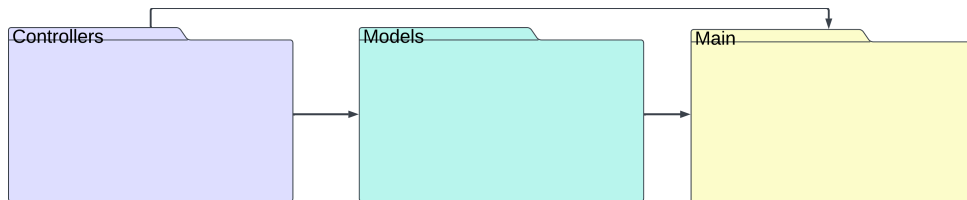


Figura 22: Diagrama de pachete pentru Server.

In ceea ce priveste clientul, aici pachetele constau din fisiere .html, care reprezinta scheletul paginilor web, precum si fisiere .css si .js care definesc aspectul, respectiv comportamentul si functionarea frontend-ului.

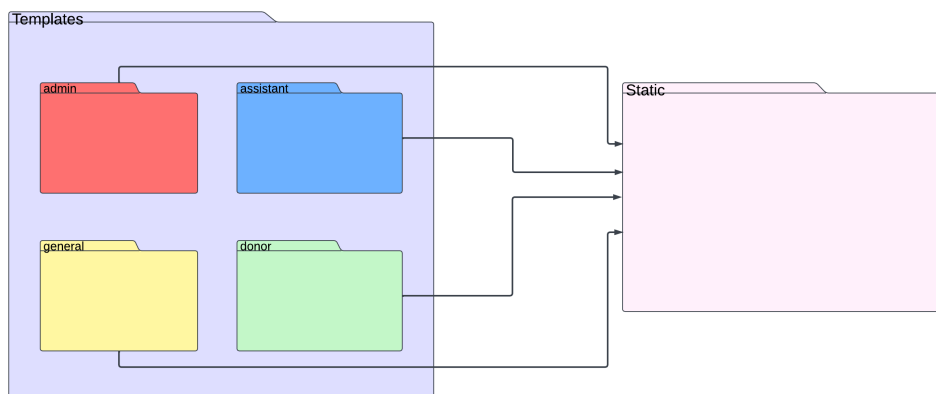


Figura 23: Diagrama de pachete pentru Client.

3.5 Diagrame de clase

Diagramele de clase descriu structura statica a unui sistem, reprezentand clasele din aplicatie si relatiile dintre ele (mostenire, asociere, compunere), precum si atributele si metodele acestora.

**Unde am scris cu culoarea rosie am precizat ce clase din pachetul Models sunt folosite in rutele din pachetul Controllers deoarece desenul arata prea incarcat daca trageam sageti, si nu se mai intelegea nimic clar.

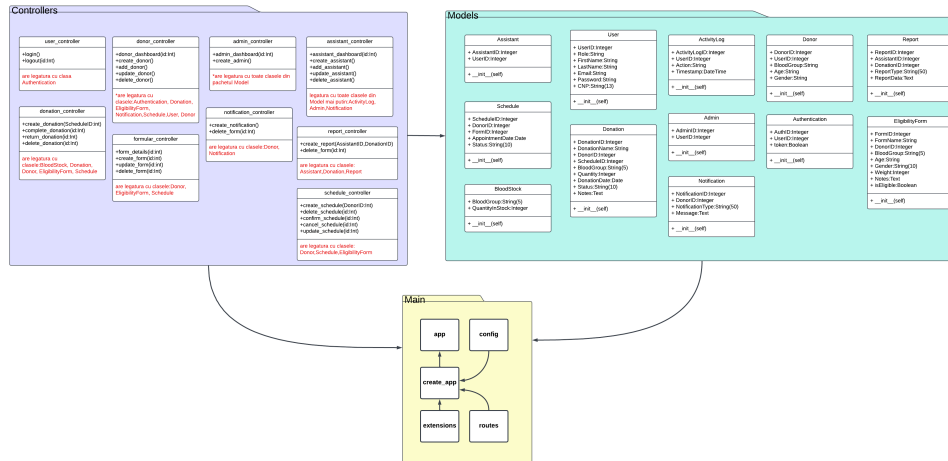


Figura 24: Diagrama de clase pentru Server.

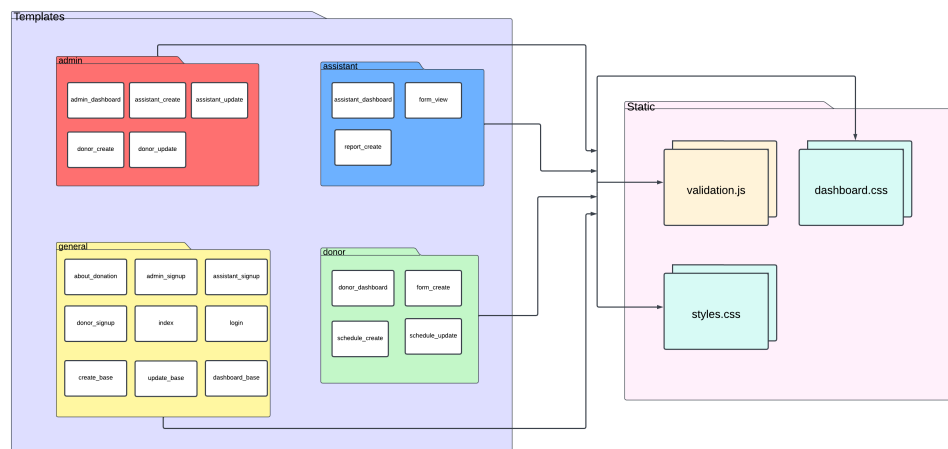


Figura 25: Diagrama de clase pentru Client.

3.6 Diagrama de componente

Diagrama de componente descrie arhitectura modulara a sistemului, indicand componentele software (de exemplu, baze de date, module de procesare) si modul in care acestea interactioneaza intre ele pentru a indeplini cerintele aplicatiei.

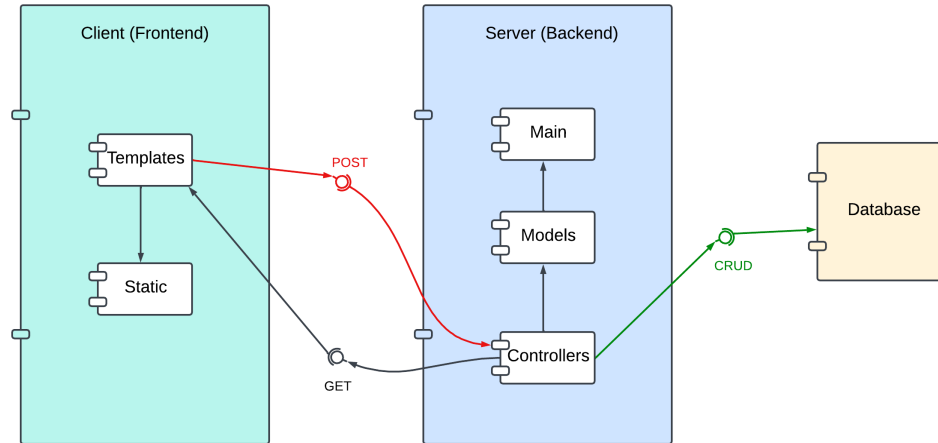


Figura 26: Diagrama de componente.

Componentele care fac parte din diagrama sunt:

- **Client:** in directorul Templates se afla fisierele .html care reprezinta scheletul frontend-ului. Folderul Static reprezinta resursele statice, precum fisiere JavaScript, CSS sau imagini.
- **Server:** Main e componenta principala a aplicatiei Backend care orchestreaza toate celelalte module. Models reprezinta logica legata de date si structurile acestora. Controllers gestioneaza cererile HTTP venite de la client.
- **Database:** Stocheaza datele aplicatiei, fiind accesata de catre server prin operatii CRUD(Create, Read, Update, Delete)

3.7 Diagrama de dezvoltare

Diagrama de dezvoltare prezinta infrastructura hardware si software necesara pentru a rula aplicatia, incluzand serverele, baze de date si alte resurse utilizate, precum si legaturile dintre acestea in timpul procesului de dezvoltare.

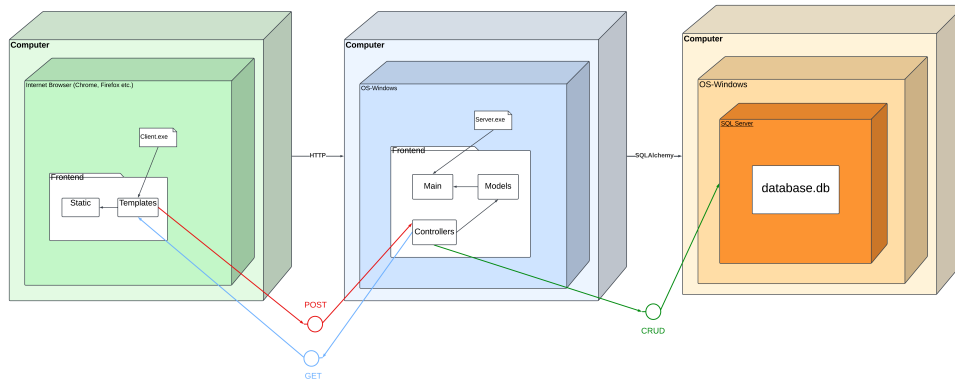


Figura 27: Diagrama de dezvoltare.

Descrierea diagramei:

- **Computer(Frontend):** Rularea aplicatiei intr-un browser. Componenta Frontend lucreaza cu Templates si Static.
- **Computer(Backend):** Ruleaza aplicatia Flask pe un server care include modulele Main, Models si Controllers. Serverul gestioneaza cererile HTTP si comunica cu baza de date.
- **Computer(Database):** Este utilizata pentru persistenta datelor din aplicatie. Este manipulata de catre backend prin intermediul operatiilor CRUD.

Linia rosie(POST) si linia albastra(GET) ilustreaza comunicarea dintre Client si Server.

- Cererea **POST** este utilizata pentru a trimite date noi catre Server. Se foloseste pentru actiuni care modifica sau creeaza resurse pe server(ex: crearea unui cont nou, completarea unui formular)
- Cererea **GET** e utilizata pentru a cere informatii de la Server, fara a trimite date suplimentare. E folosita pentru a obtine pagini web sau date din baza de date.

Linia verde(CRUD) reprezinta conexiunea dintre backend si baza de date.

4 Descrierea aplicatiei

Pagina principala a aplicatiei(index.html) aflata la adresa <http://127.0.0.1:5000>.

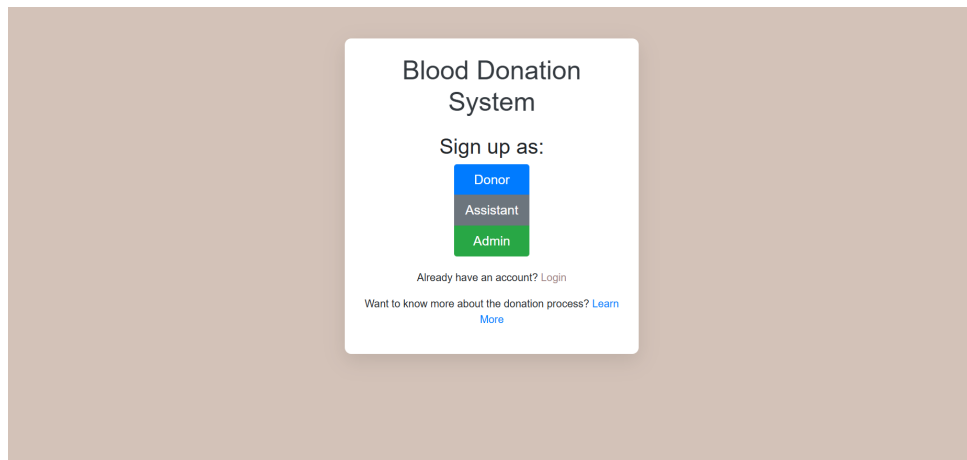


Figura 28: Pagina principala a aplicatiei - index.html.

Pagina de Log In (Autentificare) in aplicatie.

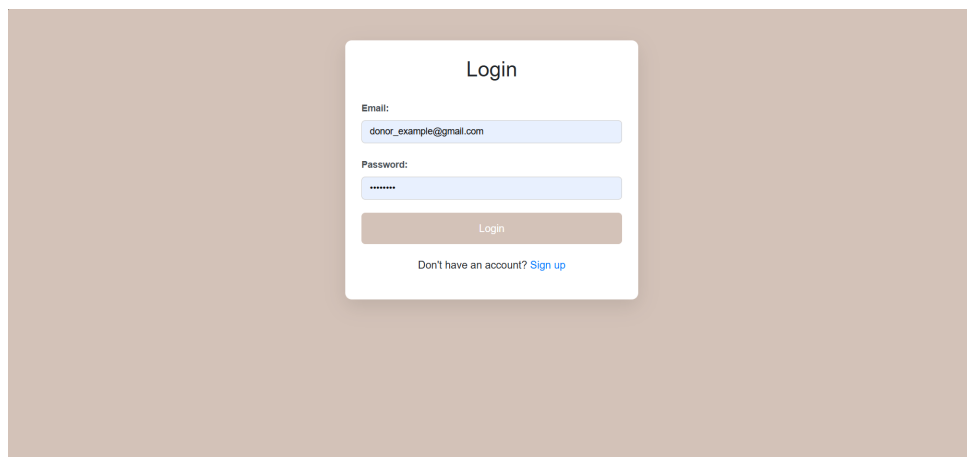
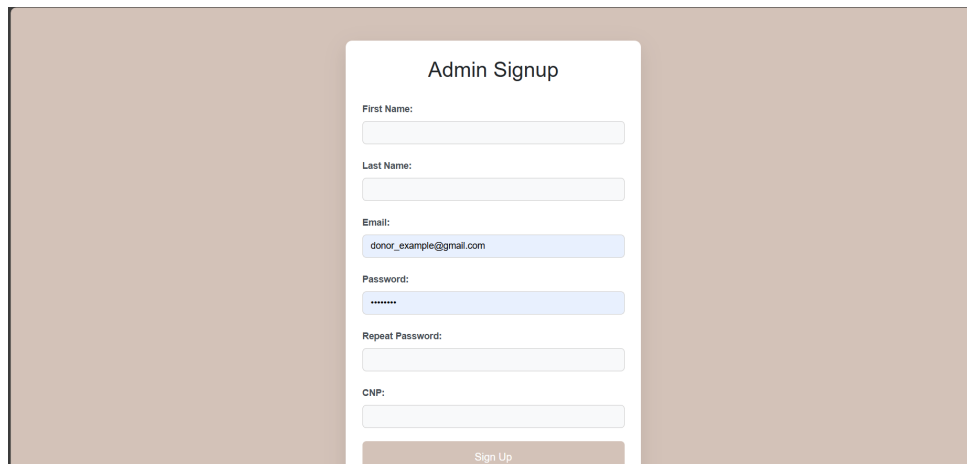


Figura 29: Pagina de Autentificare in aplicatie.

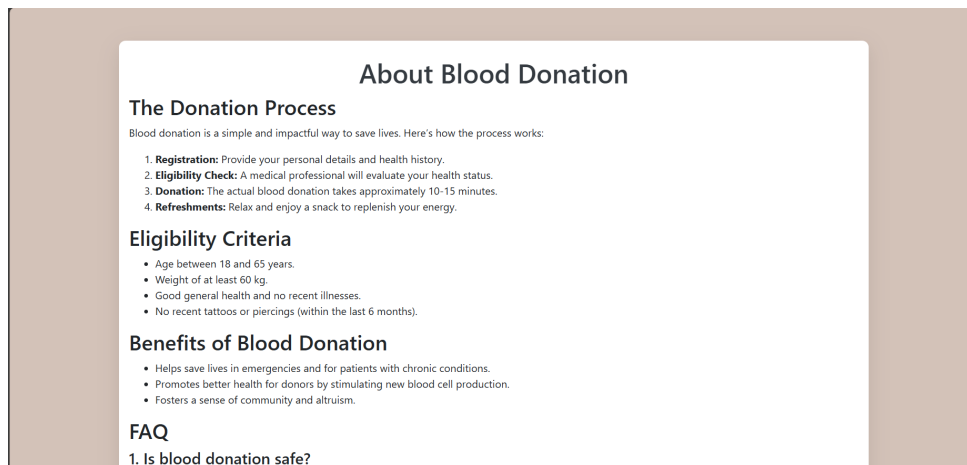
Pagina de Sign Up in aplicatie.



The image shows a web form titled "Admin Signup" centered on a light brown background. The form is white with a thin border and contains the following fields: "First Name:" with a text input; "Last Name:" with a text input; "Email:" with a text input containing the placeholder "donor_example@gmail.com"; "Password:" with a password input showing six asterisks; "Repeat Password:" with a text input; and "CNP:" with a text input. At the bottom of the form is a "Sign Up" button.

Figura 30: Pagina de Inscriere in aplicatie.

Pagina destinata utilizatorilor de tip spectator, ce contine informatii relevante legate de procesul donarii de sange.



The image shows a web page titled "About Blood Donation" with a light brown background. The content is organized into several sections: "The Donation Process" with a sub-header and a paragraph explaining the process; a numbered list of four steps: 1. **Registration:** Provide your personal details and health history. 2. **Eligibility Check:** A medical professional will evaluate your health status. 3. **Donation:** The actual blood donation takes approximately 10-15 minutes. 4. **Refreshments:** Relax and enjoy a snack to replenish your energy. "Eligibility Criteria" with a bulleted list: • Age between 18 and 65 years. • Weight of at least 60 kg. • Good general health and no recent illnesses. • No recent tattoos or piercings (within the last 6 months). "Benefits of Blood Donation" with a bulleted list: • Helps save lives in emergencies and for patients with chronic conditions. • Promotes better health for donors by stimulating new blood cell production. • Fosters a sense of community and altruism. "FAQ" with a single question: "1. Is blood donation safe?"

Figura 31: Pagina cu informatii legate de procesul donarii de sange.

Paginile web ce reprezinta dashboard-urile pentru diferite tipuri de utilizatori

Welcome!

First Name

Donor

Last Name

Example

Email

donor_example@gmail.com

CNP

8030326303943

Log Out

Notifications

Notification Type	Message	Actions
-------------------	---------	---------

Schedules

Program a Donation

Appointment Date	Eligibility Form	Status	Actions
------------------	------------------	--------	---------

Formulars

Complete Formular

Form name	Blood Group	Age	Gender	Weight	Notes	Eligibility	Actions
No forms found							

Donations History

Figura 32: Donor Dashboard.

Welcome!

First Name

Admin

Last Name

Unu

Email

admin1@gmail.com

CNP

1030326303943

Log Out

Donors

Add Donor

First Name	Last Name	Email	CNP	Age	Gender	Blood Group	Actions
Donator	Don	donor1@gmail.com	5030326303942	64	Male	AB+	Delete Update

Assistants

Add Assistant

First Name	Last Name	Email	CNP	Actions
------------	-----------	-------	-----	---------

Donations

Donation Name	Blood Group	Quantity	Donation Date	Status	Notes	Actions
---------------	-------------	----------	---------------	--------	-------	---------

Figura 33: Admin Dashboard.

Welcome!

First Name

Assistant

Last Name

Example

Email

assistant1@gmail.com

CNP

2230326303943

Log Out

Donations

Donation Name	Blood Group	Quantity	Donation Date	Status	Notes	Actions
---------------	-------------	----------	---------------	--------	-------	---------

Donor Schedules

Donor Name	Appointment Date	Eligibility Form	Status	Actions
------------	------------------	------------------	--------	---------

Reports

No reports available.

Blood Stock Chart

Quantity in Stock

Figura 34: Assistant Dashboard.

5 Concluzie si posibilitati de dezvoltare ulterioara

5.1 Justificarea alegerii: website sau desktop app

Am ales ca aceasta aplicatie sa fie o aplicatie web din urmatoarele motive:

- Accesibilitate crescuta: un website poate fi accesat de pe orice dispozitiv(PC, tableta, smartphone, laptop), fara a necesita instalare
- Suport pentru diferite sisteme de operare: permite utilizatorilor din diverse locatii sa acceseze site-ul, indiferent de OS(Linux, Windows, MacOS)
- Scalabilitate: este mai usor de scalat pentru a sustine o baza de utilizatori in crestere
- Securizare centralizata: website-ul poate avea securitatea gestionata in mod centralizat de catre server (criptarea datelor, back-up automat), ceea ce reduce riscurile asociate cu aplicatiile instalate local.

5.2 Concluzie

In urma realizarii acestui proiect la disciplina Ingineria Sistemelor, am dobandit urmatoarele cunostiinte

- Logica de business din spatele unei aplicatii web client-server
- Lucru cu instrumente din sfera dezvoltarii frontend a unui website (HTML, CSS si JavaScript)
- Lucru cu instrumente din sfera dezvoltarii sistemelor de backend a unei aplicatii web (framework-ul Flask al limbajului Python) precum si sisteme legate de gestiunea bazelor de date (SQLAlchemy, SQLite)
- Proiectarea si crearea unor schite si diagrame pentru a exemplifica grafic functionalitatea aplicatiei (diagrame de UseCase, de activitate, de componente, de dezvoltare etc.)

5.3 Posibilitati de dezvoltare ulterioara a aplicatiei client-server

- Frontend: folosirea unor framework-uri mai moderne, precum Angular, React.js, Vue.js, pentru a conferi o mai buna performanta si o experienta mai fluida pentru utilizatori.
- Backend: utilizarea unor mecanisme mai puternice pentru a asigura siguranta datelor utilizatorilor.
- Backend: folosirea unor structuri de date care sa permita o complexitate mai buna a aplicatiei precum si un timp de executie mai mic