Chat App
-Spring Boot and React-

\

Autori: Ciu Adrian-Valentin
Atitienei Stefan Costin
Profesor coordonator: Mitrea Delia

Cuprins

Specificații şi analiza sistemului 3

Precizarea limbajului de programare ales (plus motivație), a sistemului de operare sub care se face implementarea, a cerințelor hardware 3

Proiectarea aplicației 3

Arhitectura aplicatiei: 3

Functionarea aplicatiei 5

Cazuri de utilizare: 5

Diagrama de cazuri de utilizare: 6

..... 6

Diagrama de clase (UML): 7

Diagrame de secventa: 8

Diagrama bazei de date: 9

Cazuri de testare: 10

Testarea propriu-zisa a plicatiei 11

Manual de utilizare: 11

Concluzii si dezvoltari ulterioare: 13

Bibliografie 13

Specificații şi analiza sistemului

Sistemul pe care vrem sa-l implementam are ca obiectiv principal comunicarea dintre 2 persoane sa fie cat mai usoara, indifirecnt de zona geografica in care se afla.

Problema rezolvata de aceasta aplicatia este comunicarea la lunga distanta dintre 2 persoane. Solutia este implementarea unei aplicatii ce transmite si primeste instantaneu mesaje de la alti utiliazatori ai aplicatiei. Toate persoanele cu varsta mai mare de 14 ani vor putea sa aiba access la aplicatie.

Precizarea limbajului de programare ales (plus motivație), a sistemului de operare sub care se face implementarea, a cerințelor hardware

Pentru a implementa aceasta aplicatia vom utiliza urmatoarele tehnologii:

a. Java + frameworkul Spring → pentru a putea crea backend-ul, adica REST Api-ul la care frontendul se va putea conecta (folosind fisiere de tip JASON)
b. SQL Server → baza de date relationala folosita pentru a stoca datele despre utilizatori si mesajele trimise
c. React → pentru a creea frontendul (HTML, CSS, JavaScript)

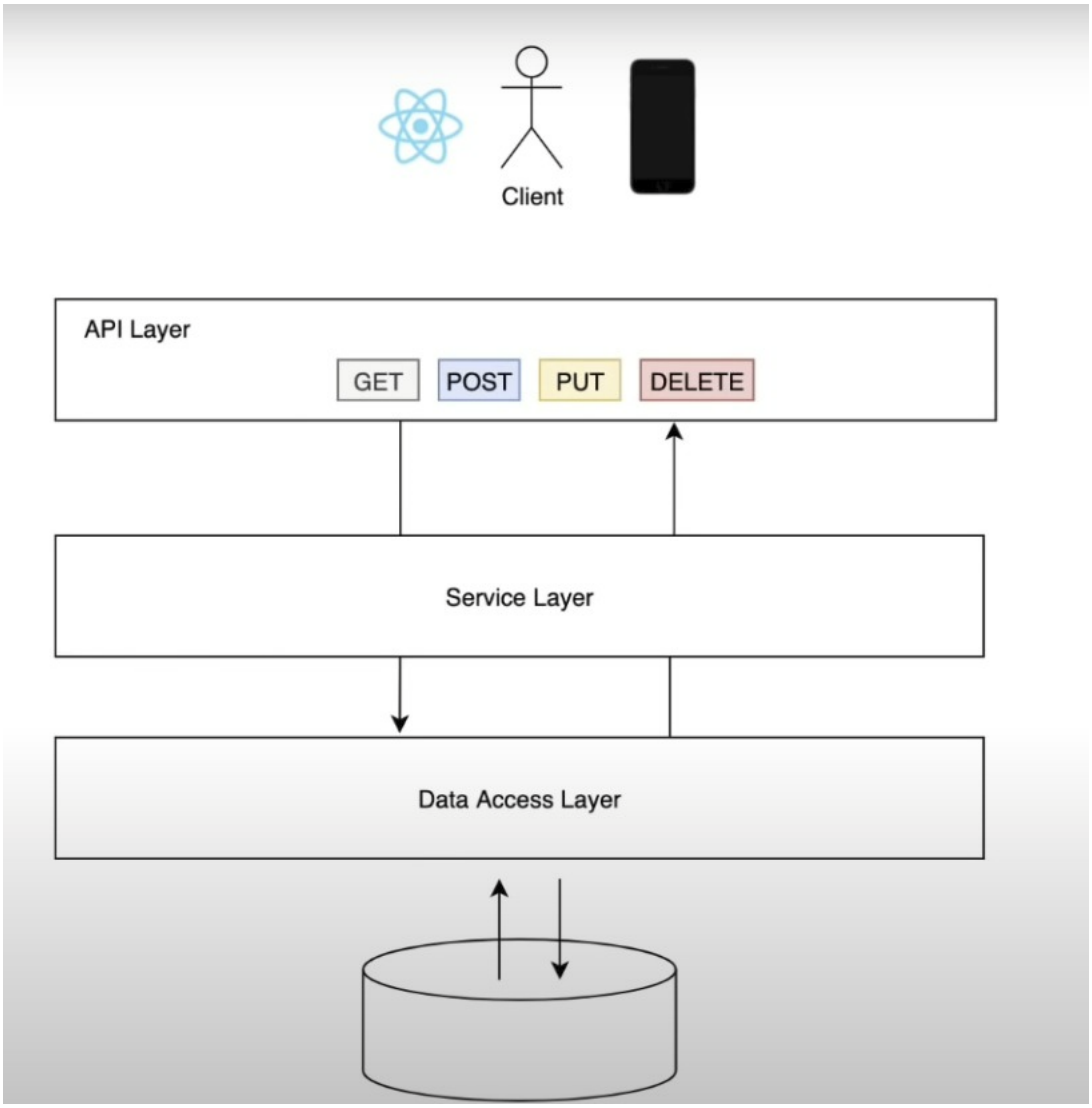
Sistemul de operare sub care se face implementarea este Windows 10/11.

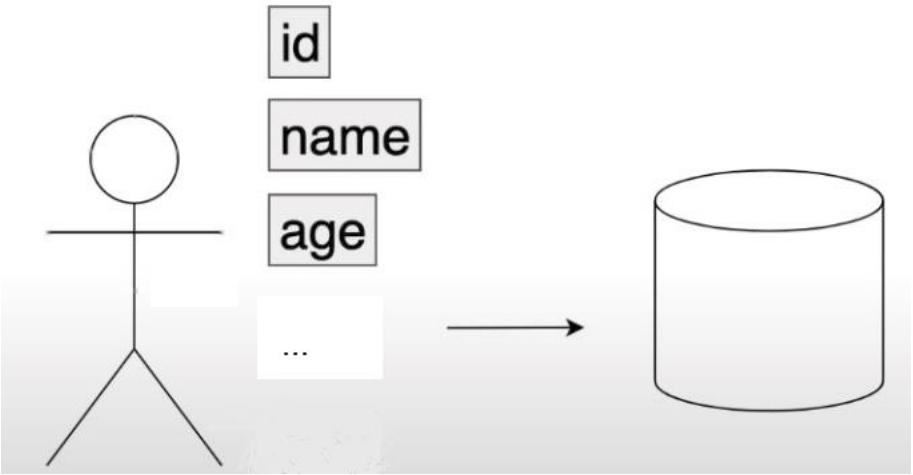
Cerintele hardware sunt minime: processor doual core si 4 GB Ram.

Proiectarea aplicației

Arhitectura aplicatiei:

- **BackEnd**
  - Pentru a stoca datele despre fiecare cont al utlizatorilor si a mesajelor transmise si primate se va utilca baza de date relationala de tip SQL Server
  - Pentru a genera fisierile det ip JASON (transimise si primate de frontend) si pentru conectarea la baza de date se va utiliza frameworkul Spring in Java. De asemenea, tot in java, utilizand frameworkul Spring, se vor implementa meotdele de securitate a contului unui utilizator si majoritatae algorimilor si logicii de functionare corecta a aplicatiei
- **FrontEnd**
  - Se va utilza aplicatia React pentru a manipula conexiunea dintre fiserele .html, .css, .js. De asemenea, utilizand acest program, se va implementa logica si algoritmii de procesare a informatiilor transimise prin fisiere de tip JASON primate de la backend. La randul sau, frontend-ul va transmite prin fisere JASON informatii de valizare si de stocare catre backend.


Client
API Layer
GET
POST
PUT
DELETE
Service Layer
Data Access Layer


id
name
age
...

Functionarea aplicatiei

- Programul permite unui utlizator sa creeze un cont nou
- Progarmul cripteaza parola unui utilizator folosind SHA-512 hashing
- Username-ul si parola unui utilizator vor fi unice
- Programul permite unui utilizator sa-si schimbe parola
- Programul trimite pe mail un link de validare al contului
- Link-ul de validare expira in 72h
- Programul permite unui utilizator sa primieasca un nou link de validare
- Conturile ce nu au fost validate vor fi sterse dupa o saptamana
- Programul permite unui utlizator sa se logheze
- Progamul permite uni utilizator sa se delogheze
- Programul va afisa daca un utilizator este conectat sau nu
- Programl permite unui utilizator sa caute un nou contact caruia sa-i trimita measje
- Programul permite unui utilizator sa selecteze un contact pentru a trimite mesaje
- Programul transmite mesajele in timp real
- Programul va afisa toate persoanle la care un utilizator a transmis mesaje

Cazuri de utilizare:

- Un utilizator poate sa-si creeze un cont nou pe aplicatie
- Un utilizator poate sa intre si sa iasa de aplicatie (log in/ log off)
- Fiecare utilizator poate sa vada daca alt utilizator este online sau nu
- Un utilizator poate sa caute alti utilizatori
- Un utilizator poate sa transmita mesaje la alti utilizatori
- Un utilizator poate sa-si vada conversatiile cu alti utilizator instantaneu
- Un utilizator poate sa vada istoria contactelor cu care a comunicat pana in momentul de fata
- Un utilizator poate sa selecteze un contact din istoricul contactelor pentru a-i transimte noi mesaje

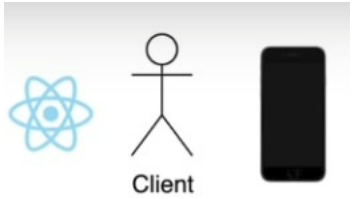


Diagrama de cazuri de utilizare:

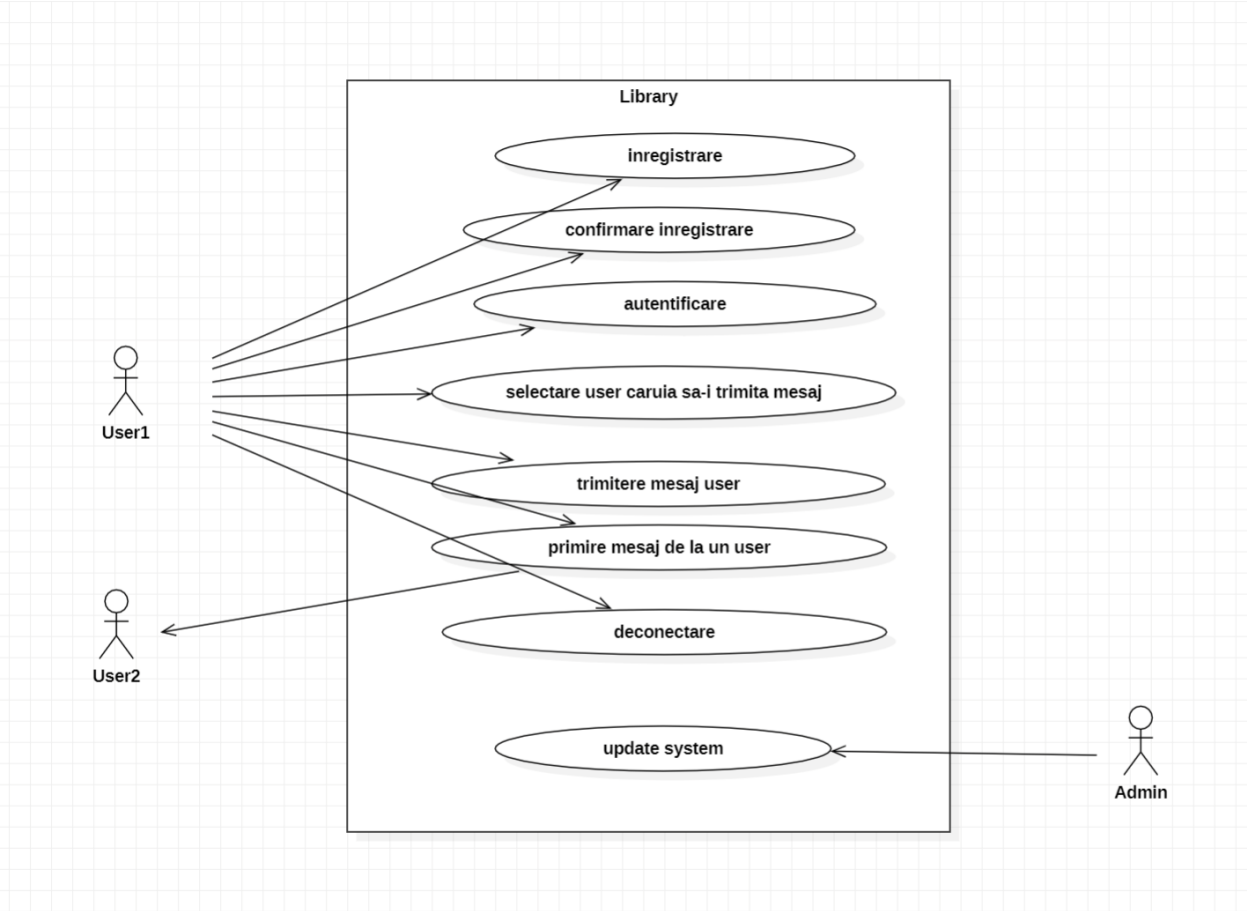
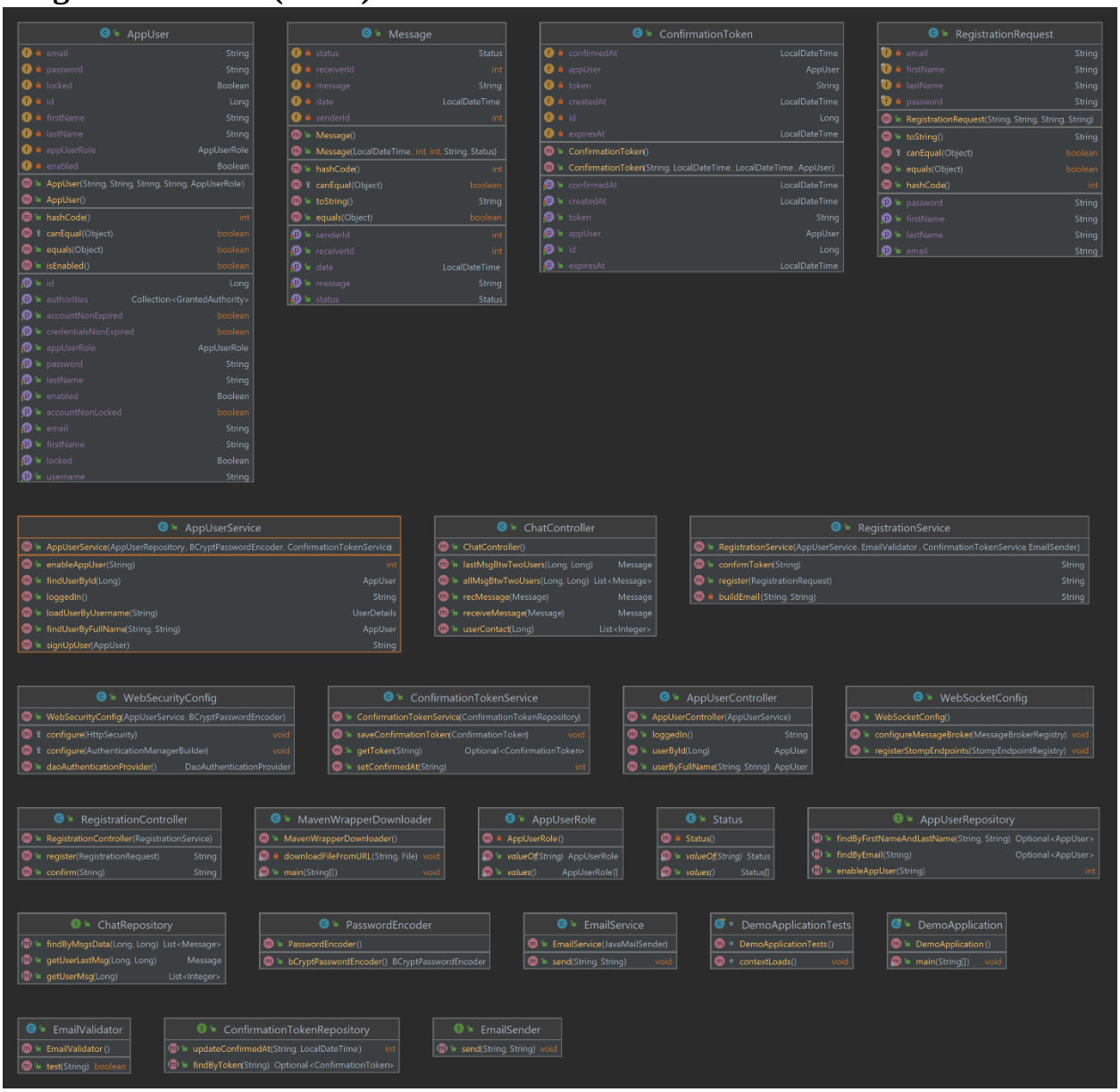


Diagrama de clase (UML):



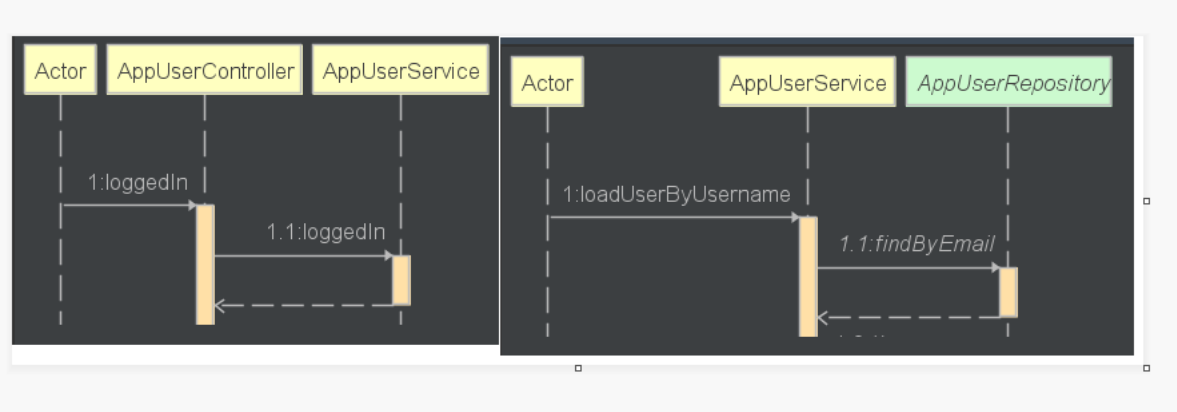
Pentru a putea sa cream API-ul in java, am utilizat mai multe pachete:

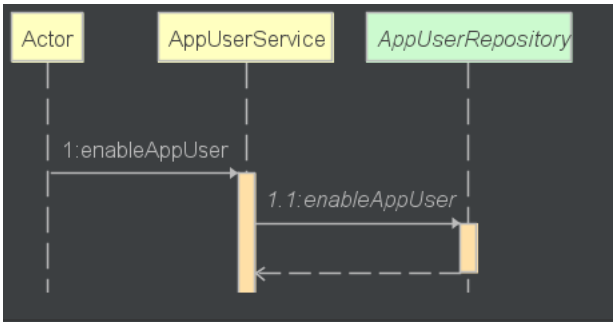
- **Authentication** → in acest pachet am implementat toate functionalitatea de logare sin in registrar a unui user. De asemenea, tot aici a fost implementata partea de securitate implementata prin framework-ul spring. Protocolul de ocmunicare folosit este cel de http, utilizand metodele de GET, POST sin PUT.
    - **Appuser** → fucntionalitatea unui user
    - **Email** → fucntiile de validare a uni email
    - **Registration → funcionalitatea de a inregistra un nou user**
    - **Security → implementarea securitatii contului.** Aici este si criptata parola unui utilzator inainte sa fie trimisa la baza de data. Chiar daca o persoana neautorizata

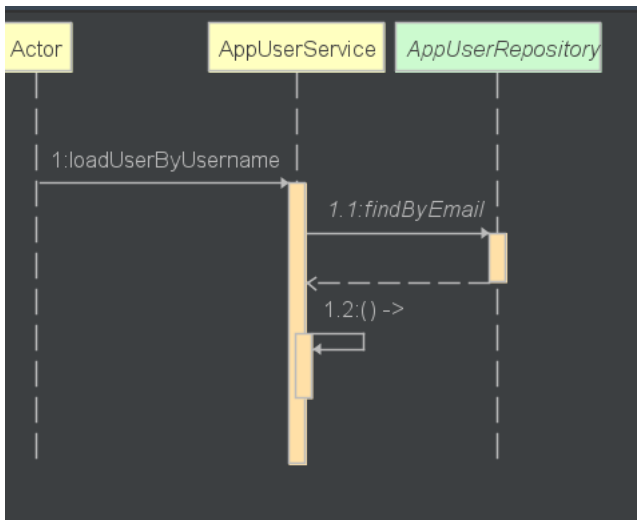
ar avea access la bazaa de date, nu ar avea cum sa acceseze contul din cauza ca parola este criptata intr-un singur sens.

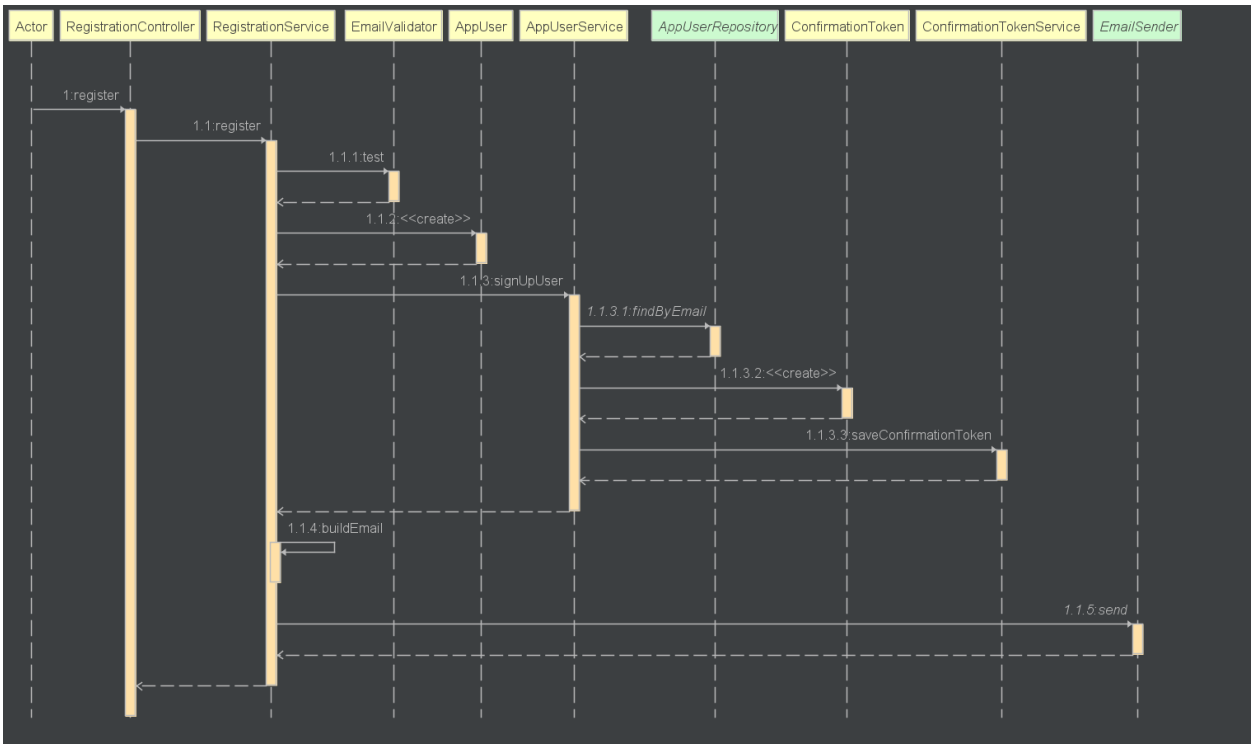
- **Message** → in aceast pachet a fost implementat protocolul de comunicare Websocket pentru a putea stoca in baza de date si trimite mesaje intre doi useri
    - Config → in acest pachet se configureaza path-urile si moduld e functionare al protocolului de comunicare Websocket
    - Model → in acest pachet se implementeaza comunicarea prin protocolul de htpp, utilizand metodele de GET, POST, PUT pentru a putea sa luam date despre user: mesajele dintre cei 2, ultimul mesaj dintre cei 2, informatii depre fiecare.

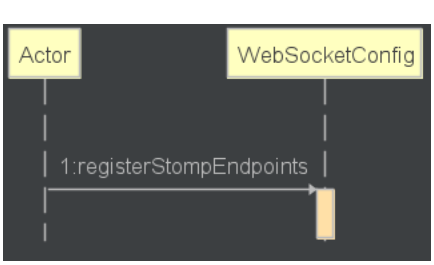
Diagrame de secventa:

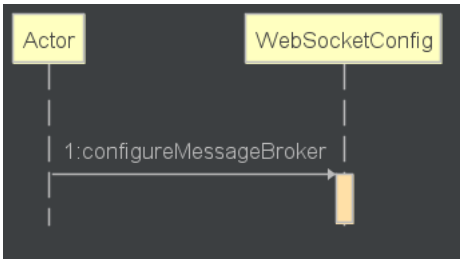












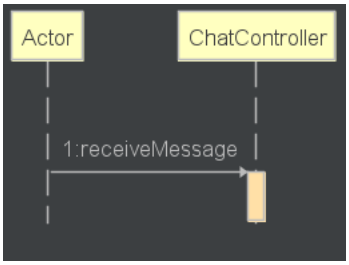
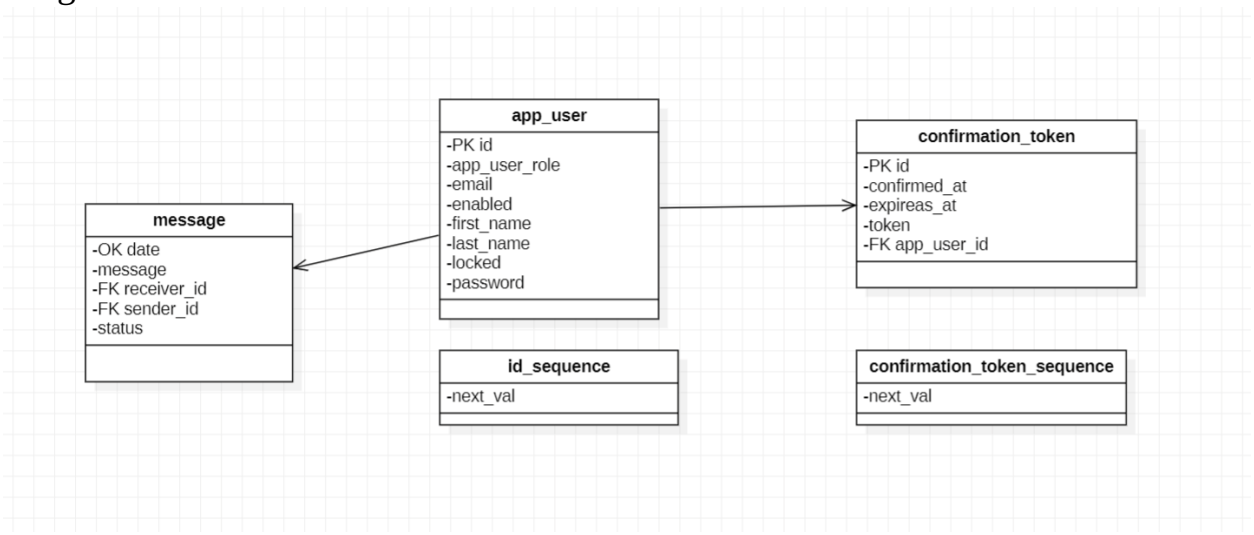


Diagrama bazei de date:



Baza de date utilizata este una da dimensiuni reduce. Scopul principal al bazei de date este de a stoca datele contului unui utlizator, ca de exmplu numele, prenumele, adresa de email si parola. Pentru a putea sa avem disponibile toate mesajele dintre 2 utilizatori, s-a utilizat tabela pentru mesaje in care sunt stocate data trimieterii mesajului si id-urile celor 2 persoane . Tabela pentru confirmarea token-ului este utilizata in proceul de inregistrare pentru a ne putea sa ne asiguram ca un utlizator si-a validat contul folosind-usi propria adresa de email. In caz contrar acel link respira in 15 minute.

Cazuri de testare:

Pentru a ne testa aplicatia am utilizat pentru aprtea de backend unde a fost implementat API-ul aplicatia Postamn pentru a putea sa trimitem http requesturi si as vedem rezultatele acestora. Astfel ami jos sunt enumerate cazurile pe care le-am testat cu aceasta aplicatie:

GET:

http://localhost:8080/api/v1/user/frd/{id} → returneaza lista de personae cu care a vorbit un utilizator

http://localhost:8080/api/v1/msg/{id1}/{id2} → returneaza mesajele transmise dintre id-urile a 2 useri; nu conteaza ordinea celor 2 id-uri in path

http://localhost:8080/api/v1/lastmsg/4/3 --> returneaza ultimul mesaj dintre id-urile a 2 useri; nu conteaza oridnea id-urilor in path

http://localhost:8080/api/v1/user/{first_name}/{last_name} → find by full name

http://localhost:8080/api/v1/user/{id} → find by id

http://localhost:8080/loggedin → se afiseaza ca user-ul este logat sau nu

http://localhost:8080/logout → delogheaza un user

POST:

http://localhost:8080/api/v1/registration → inregistreaza un nou user; returneaza tokenul ce va trimis pe mailul utilizatorului pentru confirmarea contului

JSON body example:

```
{
    "firstName": "Adrian",
    "lastName": "Ciu",
    "email": "adrianciu@yahoo.com",
    "password": "123"
}
```

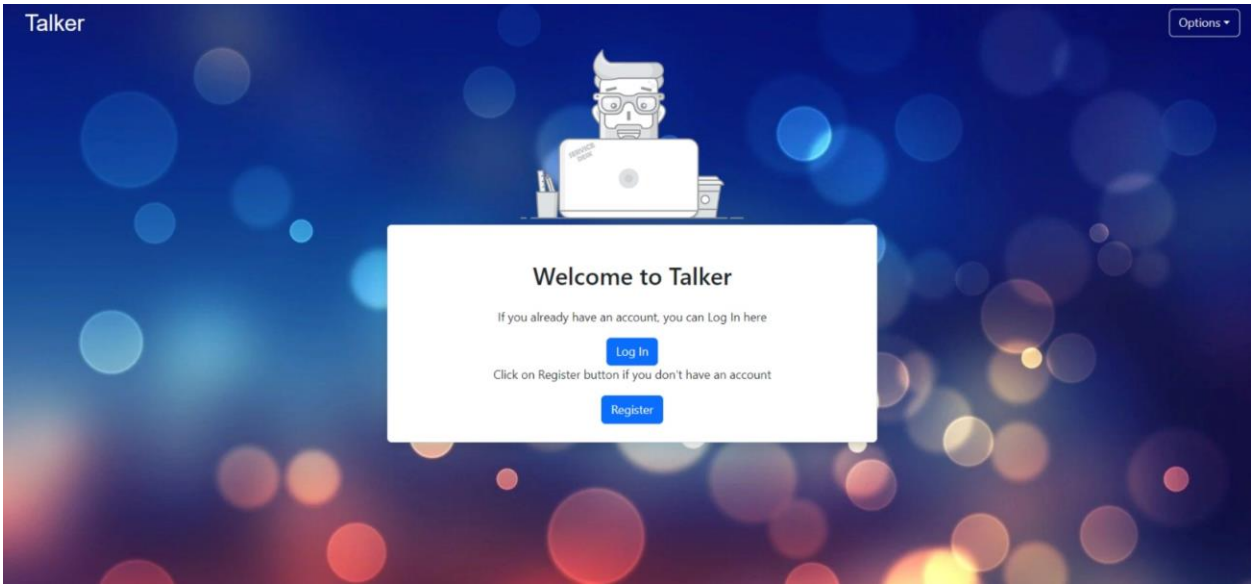
Testarea propriu-zisa a plicatiei

Pentru a testa aplicatia in mod corespunzator, am utilizat partea de frontend pentru a ne asigura ca functionalitatile pe care ni le-am propus sa le implementam functioneaza asa cum ne-am fi asteptat, mai ales ca fost si utlizat Postman inainte pentru a ne face o idee despre ce rezultate am putea sa obtinem.

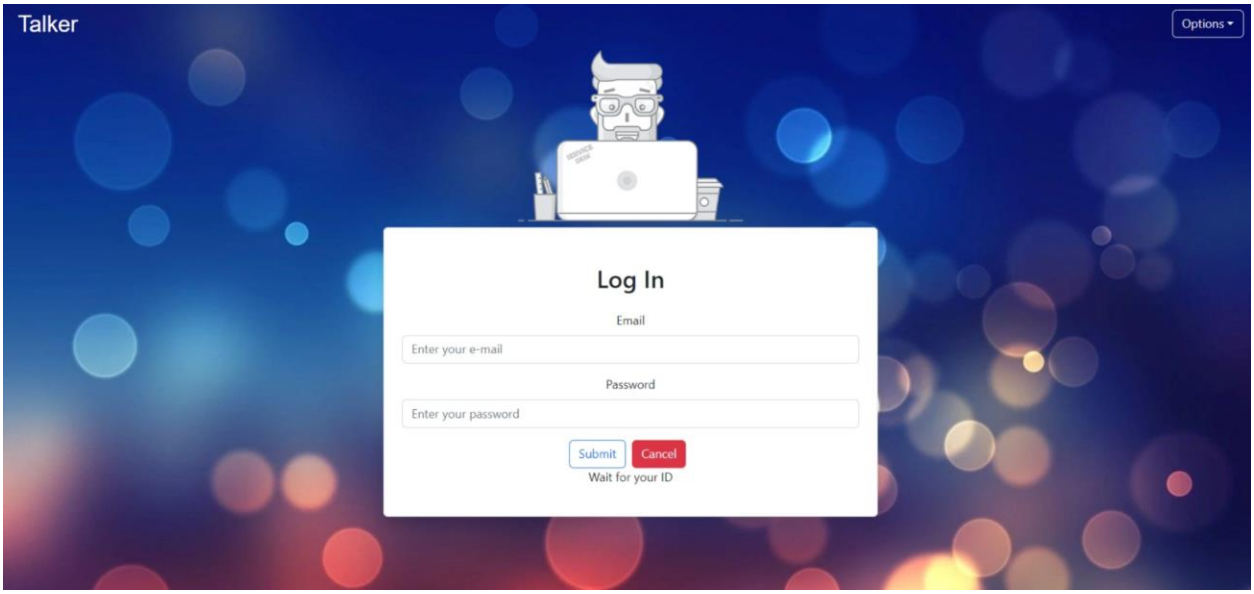
Manual de utilizare:

Pentru a putea un utilizator sa aiba access la aceasta aplicatie, nu are nevoie decat de o conexiune buna la internat si sa intre pe site-ul aplicatiei.

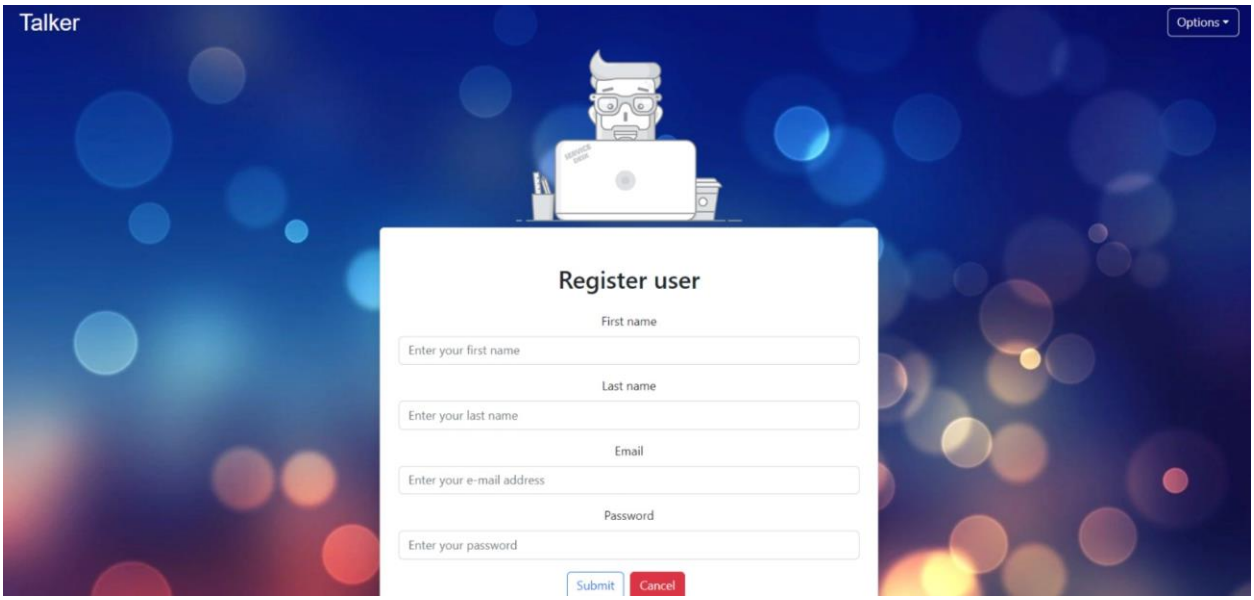
Pagina de start a aplicatiei:



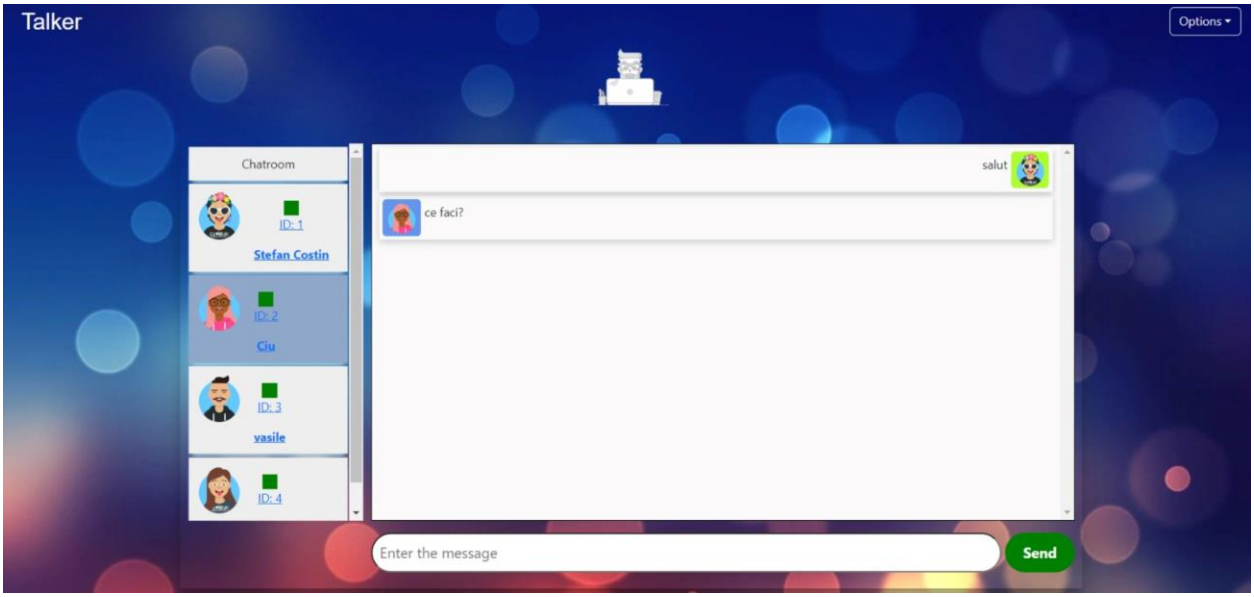
Autentificare in sistem:



Inregistrarea unui nou utilizator in sistem:



Trimitere mesaje catre un utilizator:



Concluzii si dezvoltari ulterioare:

Acesta a fost unul dintre cele mai complexe proiecte la care am lucart in echipa, dar rezultatul obtinut a fost unul multumitor si putem spune ca ne-am atins obiectivele cu acest proiect.

Ca dezvoltari ulterioare, messageria pe care am implementat-o poate sa fie imbunatatira: se poate adauga o functionalitate ca un utilizator sa poate sa trimita fisiere media ca filmulete sau poza. De asemenea, se poate implementa functia de grup in care sa poate sa vorbeasca mai multe personae. Un buton de adaugare si acceptare prietenie cu o persoana ar putea sa fie adaugat in cazul in care un user nu vrea sa primeasca mesaje de la un anumit utilizator.

Bibliografie

[1]https://stackoverflow.com/questions/66329964/spring-jpa-procedure-not-working-correctly-with-multiple-out-clause

[2] https://spring.io/guides/tutorials/rest/

[3]https://kb.supremainc.com/knowledge/doku.php?id=en:1xfaq_how_to_create_a_sql_server_authentication_login_id

[4]https://www.jetbrains.com/help/idea/db-tutorial-connecting-to-ms-sql-server.html#connect-by-using-windows-domain-authentication-macos-and-linux

[5] https://www.callicoder.com/spring-boot-websocket-chat-example/

[6] https://www.callicoder.com/spring-boot-websocket-chat-example/

[7] https://github.com/callicoder/spring-boot-websocket-chat-demo

[8]https://o7planning.org/10719/create-a-simple-chat-application-with-spring-boot-and-websocket

[9] https://www.youtube.com/watch?v=_696SMBLqRA&ab_channel=CodeForgeYT

[10]https://stackoverflow.com/questions/33407079/spring-security-custom-authentication-provider-always-redirect-to-login-page

[11] https://www.baeldung.com/spring-request-response-body

[12] https://reactjs.org/tutorial/tutorial.html

[13] https://reactjs.org/docs/faq-ajax.html

[14] https://www.npmjs.com/package/react-messaging

[15] https://reactjsexample.com/tag/message/