Universitatea Tehnica din Cluj-Napoca
Facultatea de Automatica si Calculatoare
Sectia de Automatica si Informatica Aplicata

Proiect Sisteme de Control Distribuit

Student : Ghiran Lorena Roxana
Grupa: 30144

Îndrumător: Balea Ștefan

Anul universitar: 2020-2021
Data: 13.01.2021

Cuprins

1. Sistem de monitorizare prin GPS – cerințe

Se va proiecta si se va implementa un sistem distribuit pentru monitorizare poziției prin GPS.
Sistemul va fi compus din urmatoarele aplicații:

SERVER

Aplicație Java Enterprise (J2EE) oferă diferite funcționalitati precum: salvarea unei noi pozitii in baza de date, operatii asupra unei pozitii salvate in baza de date precum: delete, update, get. Totodata, aplicatia furnizeaza toate pozitiile unui utilizator intr-un anumit interval de timp.

Aplicatia va rula într-un servlet container(ex: Tomcat, JBoss). Va exista o singura astfel de aplicatie in sistem (**1 instanta**).

CLIENT

Se va implementa in J2ME, Android sau IOS. Aplicatia permite logarea in aplicatie a unui utilizator existent precum si crearea unui nou utilizator in aplicatie. Aplicatia va citi pozitia curenta in mod automat si o va trimite la server (periodic – ex. La fiecare 3 minute). Odata ce acest buton este activat, apare automat un buton de stop prin care se poate anula trimiterea.

Aplicatia va avea si un buton prin apasarea caruia se va trimite pozitia in mod manual. Vor putea exista mai multe astfel de aplicatie in sistem (**n instante**).

MONITOR

Este o aplicatie **WEB,** folosind orice suita de tehnologii (Jquery / Angular / React /Vue), ce permite vizualizarea pozitiilor istorice ale unui utilizator. Va avea un mecanism simplu de login (pentru un utilizator cu rol de administrator).

Administratorul are la dispozitie o lista cu toti utilizatorii existenti dintre care poate sa isi aleaga un utilizator. Totodata, are la dispozitie doua noi setari precum: **startDate** si **endDate**. Aplicatia server va returna toate pozitiile salvate de catre userul curent in intervalul specificat. Aceste pozitii vor fi afisate pe o harta folosind Google API.

Aceasta aplicație va rula în cadrul aceluiasi servlet container sau folosind reverse-proxy-uri precum Nginx, Apache Web server.

Comunicarea intre cele trei componente se realizeaza prin intermediul protocolului HTTP/HTTPS folosind servicii REST iar mesajele având format JSON.

PROIECTARE

Sistemul va fi proiectat prin intermediul urmatoarelor tipuri de diagrame UML:

- Diagrama cazurilor de utilizare (use-case);
- Diagrama claselor;
- Diagrame secventiale si diagrame ale masinilor de stare specifice celor 3 operatii esentiale: login, salvare pozitie si afisarea pozitiilor pe harta.

DOCUMENTAȚIE

Proiectul va insotit de o documentatie ce va contine:

- Ceritele (acest document);
- Specificatii;
- Cele 4 tipuri de diagrame UML mentionate anterior.

2. Specificații

Sistemul de monitorizare a locației prin GPS este alcătuit din 3 părți :

- Server = Backend
- Frontend = Web app (neimplementat)
- Aplicație mobilă (neimplementat)

Partea de **SERVER** a fost realizată utilizând limbajul de programare Java, folosind mediul de dezvoltare IntelliJ IDEA, împreună cu frameworkul Spring.
Partea de backend este de tip **3 TIER ARCHITECTURE** :

- Controller
- Service
- Repository + Model = DAL (Data Access Layer)

Partea de **CONTROLLER** conține endpoint-urile pentru funcționalitățile de register, login, add location, update location, delete location, find location by user id, filter location by userId/start/end date. Am realizat 2 tipuri de Controller :
-User Controller- conține endpoint-urile care pot fi apelate de către basic user + admin.
-Admin Controller- conține endpoint-urile care pot fi apelate doar de către rolul de admin.
Aceste endpoint-uri le-am testat ulterior cu Postman.

Partea de **SERVICE** este partea internă a serverului. Conține metodele, acțiunile de bază atât generic cât și implementate și partea de securitate pentru obținerea detaliilor despre utilizatorii din baza de date.

Partea de **REPOSITORY** conține partea de interogări și reprezintă interacțiunea cu baza de date.

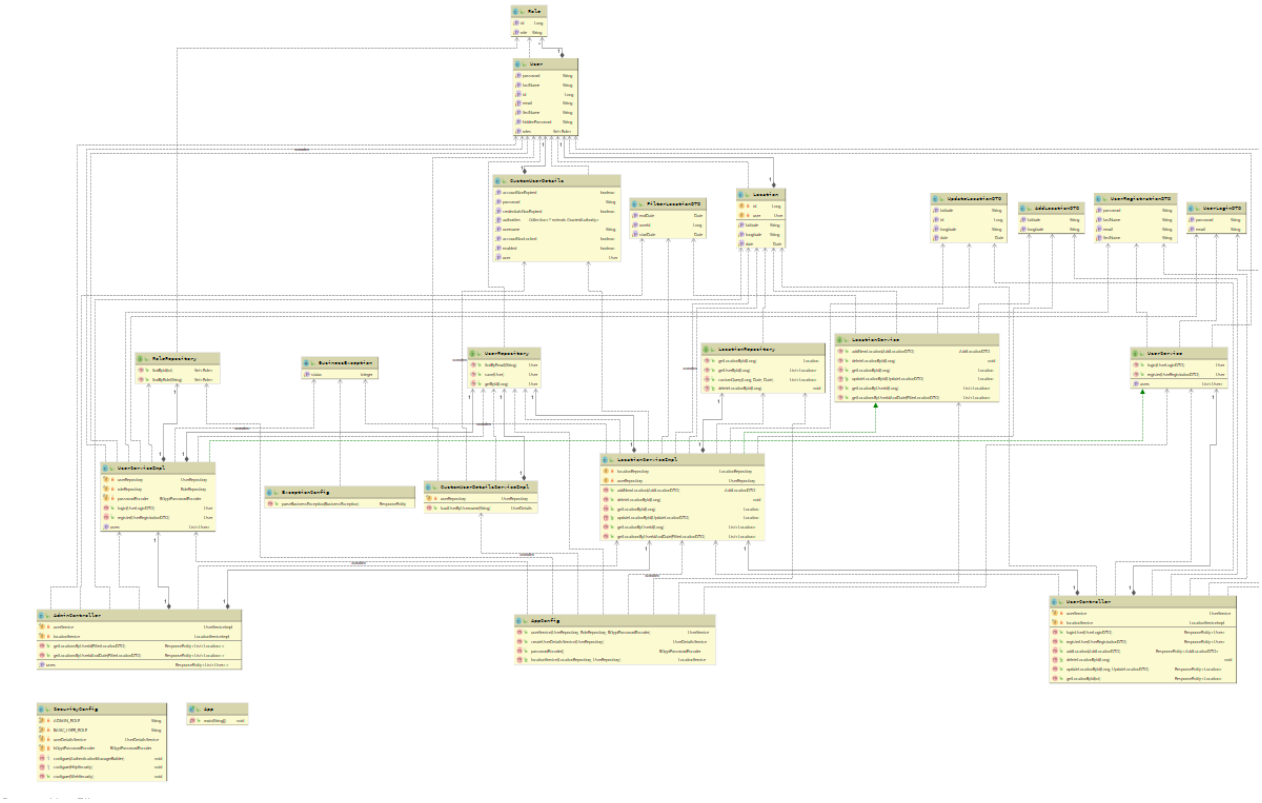
Partea de **MODEL** reprezită modelul de obiecte și conține partea de DTO (Data Transfer Object), Security și partea de entități pentru baza de date.

Baza de date este creată în MySQL Workbench, tabelele fiind create/updatate/salvate direct prin startarea serverului, prin intermediul Hibernate-ului care se ocupă de tot ceea ce ține de baza de date.

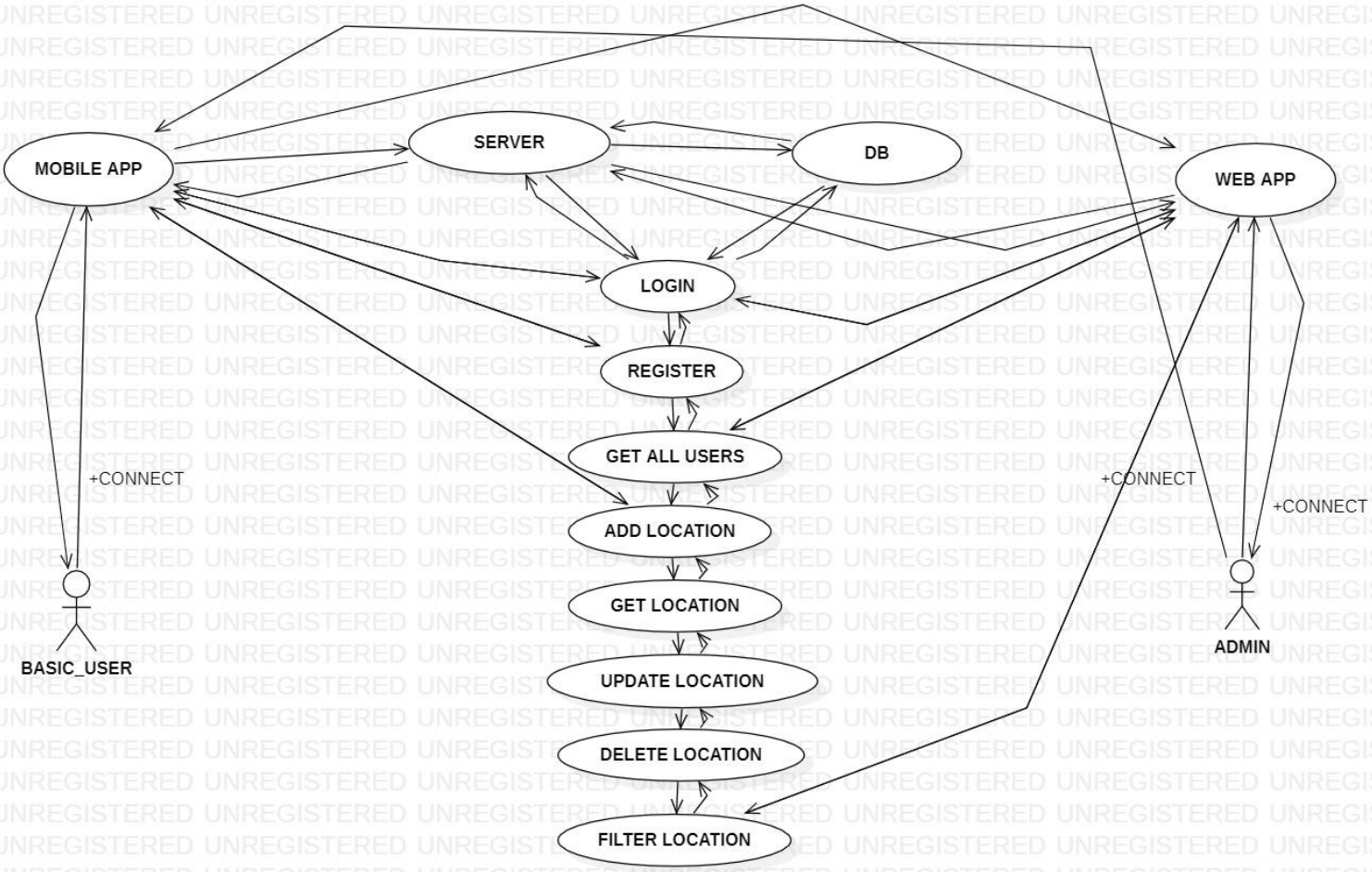
Bazat pe principiul CRUD (Create, Read, Update, Delete) am realizat requesturile pe care ulterior le-am testat în Postman.
Acestea sunt : login (anonymous), register (anonymous), get all users (admin), create new location (basic user/ admin), update location (basic user/ admin), delete location (basic user/ admin), get location by id (basic user/ admin), filter location by user id/start date/end date (admin).

3. Proiectare

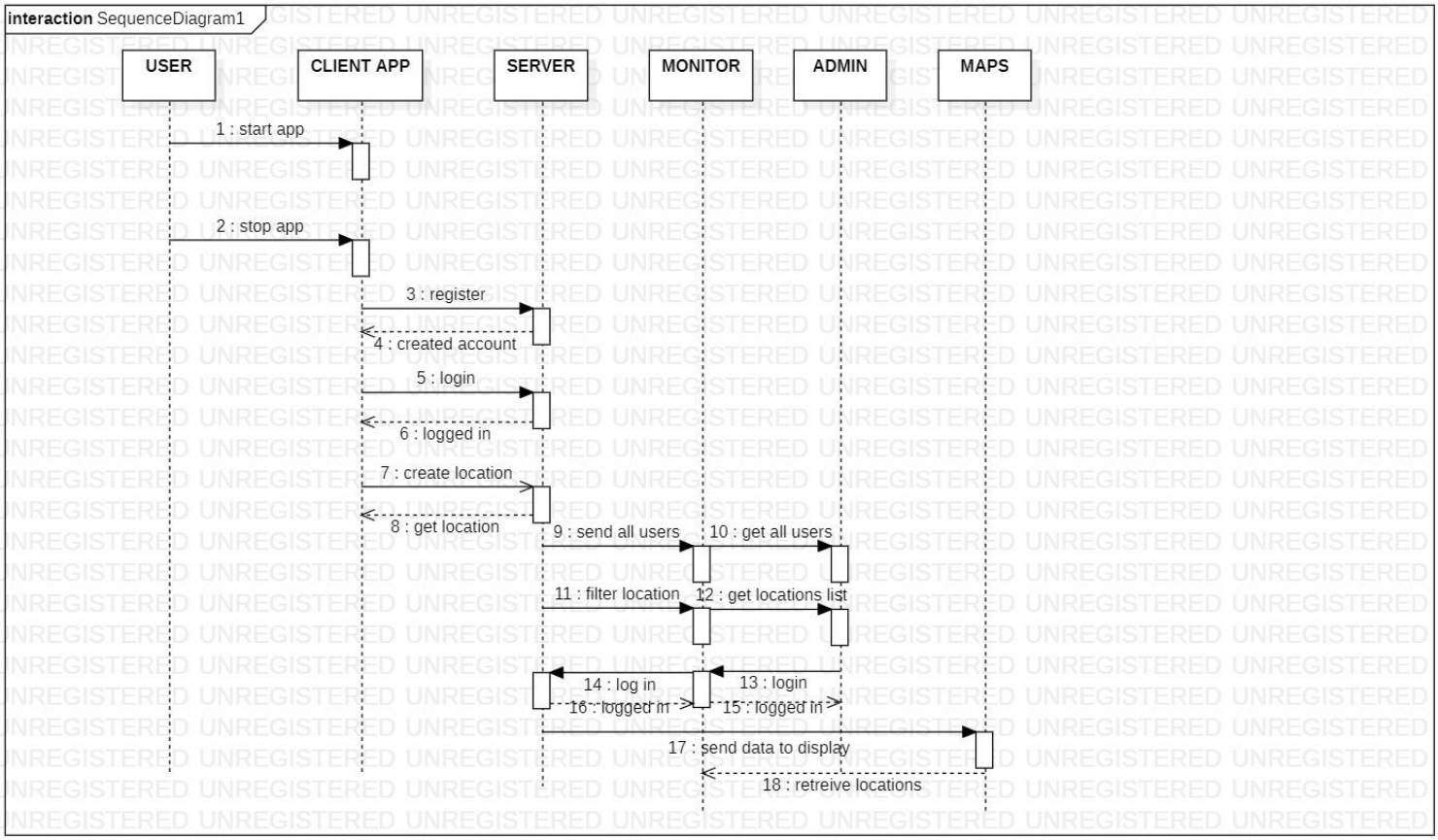
3.1 Diagrama Claselor



3.2 Diagrama use-case



3.3 Diagrame secvențiale



3.4 Diagrame mașini de stare