

**Formula 1 Fantasy Draft**

**Proiect realizat de**

Popirtan Vlad-Mihai

**Profesor indrumator**

Radu Gabriel Tufisi

18/03/2021

**0) Contents**

I.Specificatia proiectului......................................................................................................

1.1 Diagrama Domain Model...................................................................................

II.Modelul Use-Case............................................................................................................

2.1 Utilizatori si stakeholderi...................................................................................

2.2 Identificare Use Case-uri...................................................................................

2.3 Diagrama UML Use Case.................................................................................

III.Design arhitectural..........................................................................................................

3.1 Arhitectura conceptuala.....................................................................................

3.2 Diagrama de pachete..........................................................................................

3.3 Diagrama de clase..............................................................................................

3.4 Diagrama bazei de date......................................................................................

3.5 Diagrame de secventa.........................................................................................

3.6 Diagrame de activitati.........................................................................................

IV Specificatii suplimentare.................................................................................................

4.1 Specificatii non-functionale...............................................................................

4.2 Constrangeri de design......................................................................................

V Testare.............................................................................................................................

5.1 Testarea functionalitatii aplicatiei.....................................................................

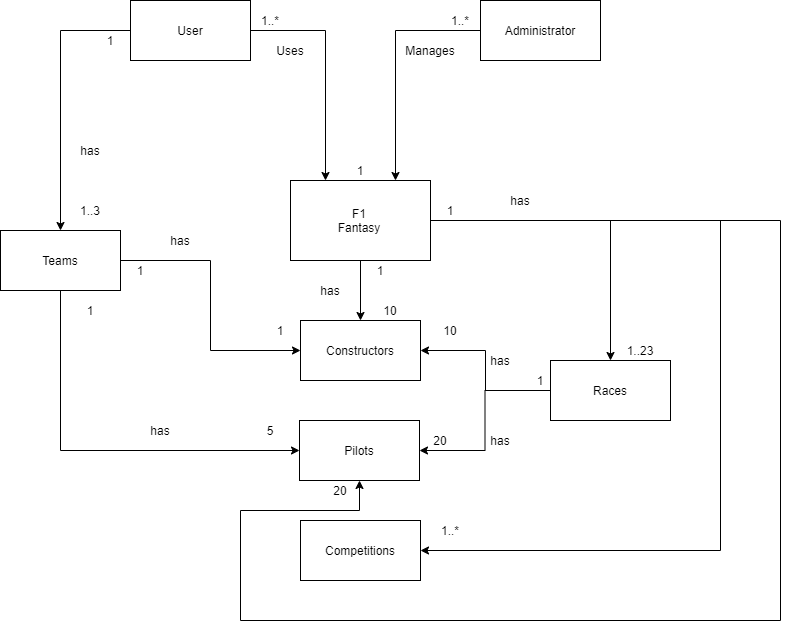
5.2 Dezvoltari ulterioare.........................................................................................

VI.Bibliografie......................................................................................................................

**1) Specificatia proiectului**

Se doreste implementarea unei aplicatii in care utilizatorul isi creeaza o echipa de cinci piloti si un constructor, in cadrul unui buget prestabilit pentru fiecare sezon. Dupa ce acesta isi creeaza echipa, utilizatorul poate decide sa se inscrie cu ea in competitii alaturi de alti utilizatori pentru a concura intre echipe. Dupa fiecare cursa, pretul pilotilor/constructorilor poate varia, iar utilizatorii pot alege sa isi modifice echipa. In acelasi timp, aplicatia va functiona ca un tracker pentru rezultatele din Formula 1. User-ul va putea vizualiza rezultatele curselor sau ale unui anumit pilot, sa creeze echipe, sa stearga echipe, sa modifice echipele sale si sa se inscrie in competitii. Administratorul poate crea sau sterge piloti, constructori sau competitii, poate crea, sterge sau edita informatiile despre curse/piloti/constructori in functie de rezultatele din realitate..

* 1. **Diagrama Domain Model**



**2) Modelul Use-Case**

**2.1) Utilizatori si stakeholder**

In aplicatie avem doua tipuri de utilizatori:

1. Administrator
2. User
3. Echipele de Formula 1

Stakeholderii principali ai acestei aplicatii sunt: administratorii echipei media Formula 1, utilizatorii pasionati de acest sport si echipele de Formula 1 ce beneficiaza de publicitate.

**2.2) Identificarea Use Case-uri**

Nume Use case: Crearea unei echipe noi

Nivel: User-Goal

Actor principal: User

Scenariul principal de success: Dupa ce user-ul realizeaza operatia de login, acesta acceseaza pagina “Teams”, unde isi va gasi echipele existente. Acesta va alege unul din slot-urile disponibile pentru a isi crea noua echipa si va apasa butonul “Create Team”. Utilizatorului ii va aparea o noua fereastara unde isi poate completa echipa, apsand butoanele “Add Driver” sau “Add Constructor”, unde ii vor aparea pilotii sau constructorii nealesi, dar si bugetul valabil. La final, user-ul va apasa butonul “Finish” pentru a salva echipa.

Extensie: Daca utilizatorul are deja 3 echipe, acesta nu va avea niciun slot disponibil pentru a crea o noua echipa, si va fi obligat sa editeze una existenta. De asemenea daca la apasarea butonului “Finish”, bugetul utilizatorului este negativ, acestuia ii va aparea o fereasta de eroare.

Nume Use case: Inregistrare intr-o competitie

Nivel: User-Goal

Actor principal: User

Scenariul principal de success: Dupa ce user-ul realizeaza operatia de login, acesta acceseaza pagina “Competitions”, unde ii vor aparea toate competitiile disponibile. De aici va selecta competitia in care vrea sa se inscrie sau va introduce id-ul uneia, ceea ce il va duce catre o fereastra in care isi poate alege echipa cu care vrea sa se inscrie. Inregistrarea se va finaliza odata cu apaseara butonului “Done”.

Extensie: Daca utilzatorul este deja inscris in competitie, acestuia ii va aparea o fereastra de eroare.

Nume Use case: Modificarea datelor unui pilot

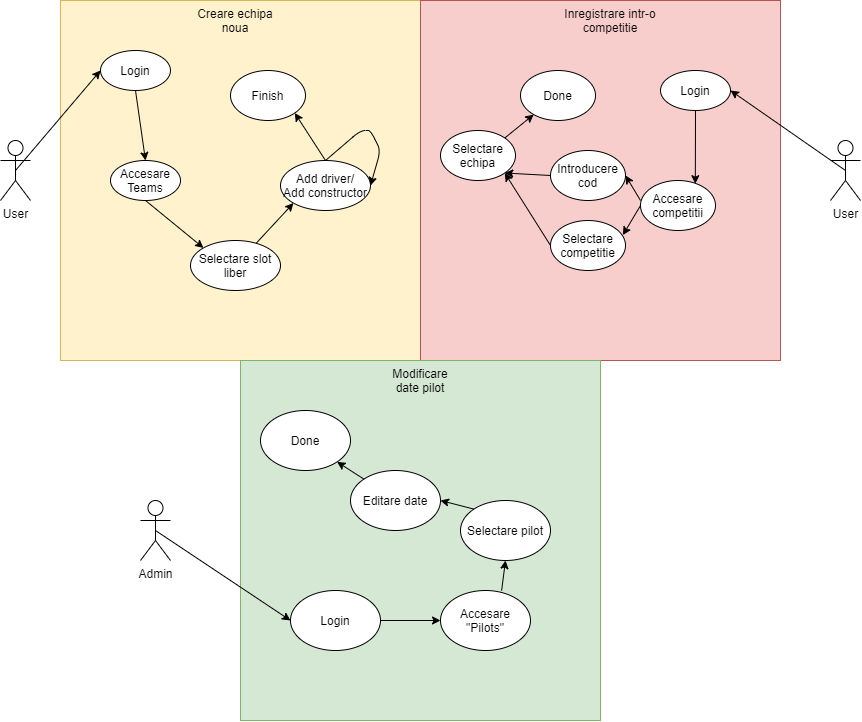
Nivel: User-Goal

Actor principal: Admin

Scenariul principal de success: Dupa ce administratorul realizeaza operatia de login, acesta acceseaza pagina “Pilots”, specifica acestui tip de user. Din aceasta fereasta, administratorul alege un pilot si ii poate modifica numele, nationalitatea, varsta, echipa, punctajul in campionat, numarul de victorii/podiumuri sau punctele in candrul fantasy draft-ului.

Extensie: Daca administratorul introduce date invalide precum varsta negativa sau numarul podiumurilor negative, acestuia ii va aparea o fereastra de eroare.

**2.3) Diagrama UML Use Case**



**3) Design arhitectural**

**3.1) Arhitectura conceptuala**

Pentru realizarea proiectului s-a ales o arhitectura bazat pe nivele de tip Layered Architecture. Pachetele de clase sunt impartite in nivele ce au rolul de a indeplini aceeasi functionalitate in program. Nivelele prezente in program sunt: Presentation Layer, Business Layer, Persistence Layer.

*Persistence Layer:* Acesta este nivelul in care se afla pachete ce tin de preluarea si salvarea datelor. Pachetele ce fac parte din Persistence Layer sunt:

• Repository – Acest pachet contine clasele cu rolul de a executa querriuri ce acceseaza baza de date. Prin aceste clase preluam date din baza de date si le trimitem catre clasele din Business Logic Layer pentru a fi salvate.

• Querries – Acest pachet contine clasele ce salveaza querriurile ce urmeaza sa fie apelate de clasele din pachetul Repository.

• Entity – Acest pachet contine clasele in care stocam datele primite in urma apelarii querriurilor.. Ele reprezinta niste replici ale tabelelor din baza de date.

*Business Laye*r: Acesta este nivelul in care se afla majoritatea functionalitatilor din interiorul proiectului. Pachetele ce fac parte din Business Layer sunt:

• Dto – Pachetul dto, Data Transfer Object, contine clase ce fac posibil transferul de date intre nivele. Folosind aceasta clasa in interiorul programului, nu trebuie sa recurgem la clasa entity, al carui scop este de a salva datele primite din baza de date.

• Exception – Pachetul exception contine o serie de mesaje ce apar atunci cand integritatea datelor este incalcata de catre utilizator.

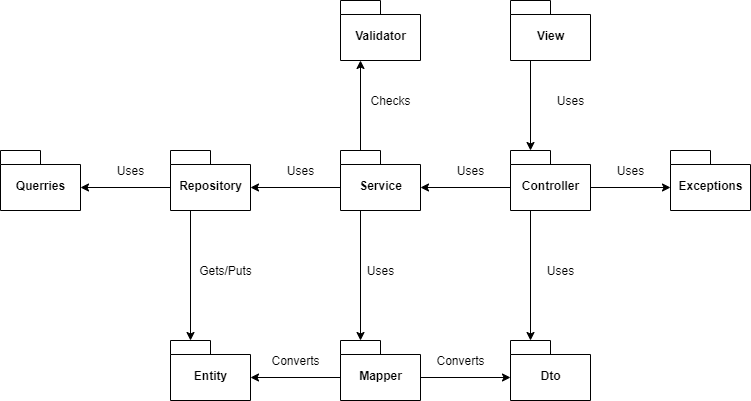
• Validator – Pachetul validator contine clase ce rezolva aceste incalcarii a integritatii datelor. Ele apoi afiseaza mesajele specifice erorii.

• Mappers – Pachetul mappers se ocupa de trecerea de la clasele din pachetul dto in clasele din pachetul entity, facand astfel un transfer intre baza de date si proiect (Valabil si invers).

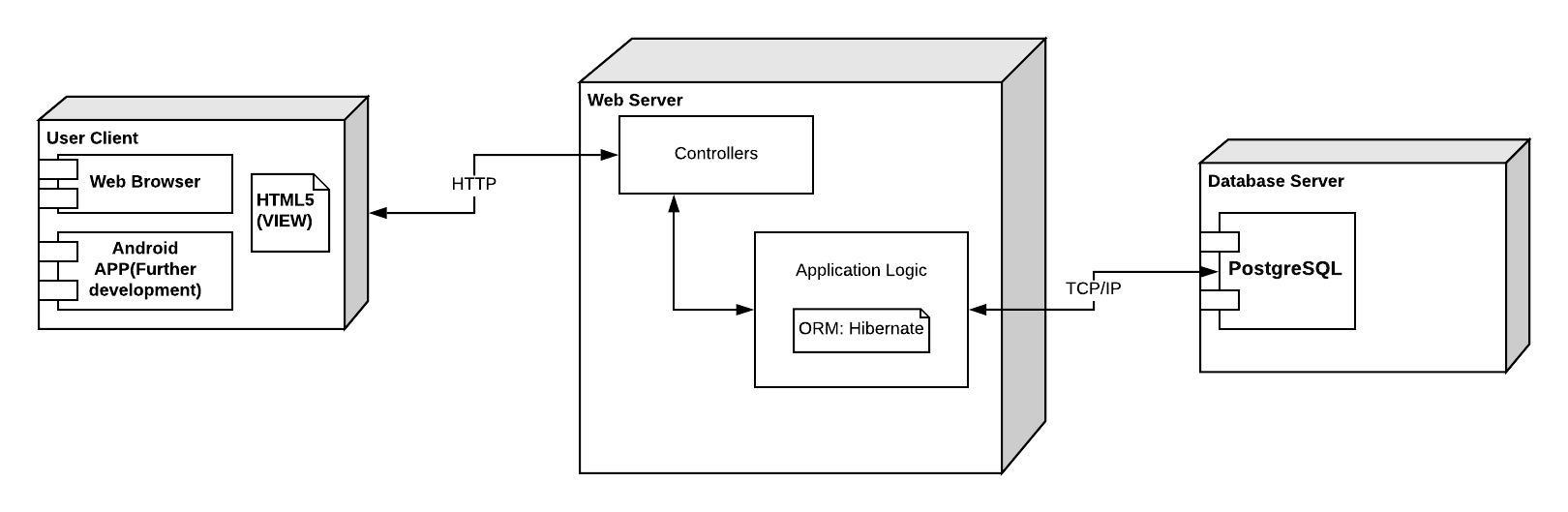
• Service – Acest pachet contine majoritatea claselor ce manipuleaza datele programului. Acest pachet face legatura intre datele salvate din pachetul dto si controllerul interfetei grafice specifica tipului de date. In acest pachet avem metode precum “getAllMatches()”, getLiveMatches(), getAllPlayers() etc;

*Presentation Layer:* Acest nivel este accesibil de catre utilizatorii aplicatiei prin intermediul unui browser, si acesta este compus din componente de interfata si componente ce suporta interactiunea cu sistemul. Acesta va fi realizat din doua componente: React Native pentru a edita pagina, iar pentru a controla pagina web vom folosi Spring.

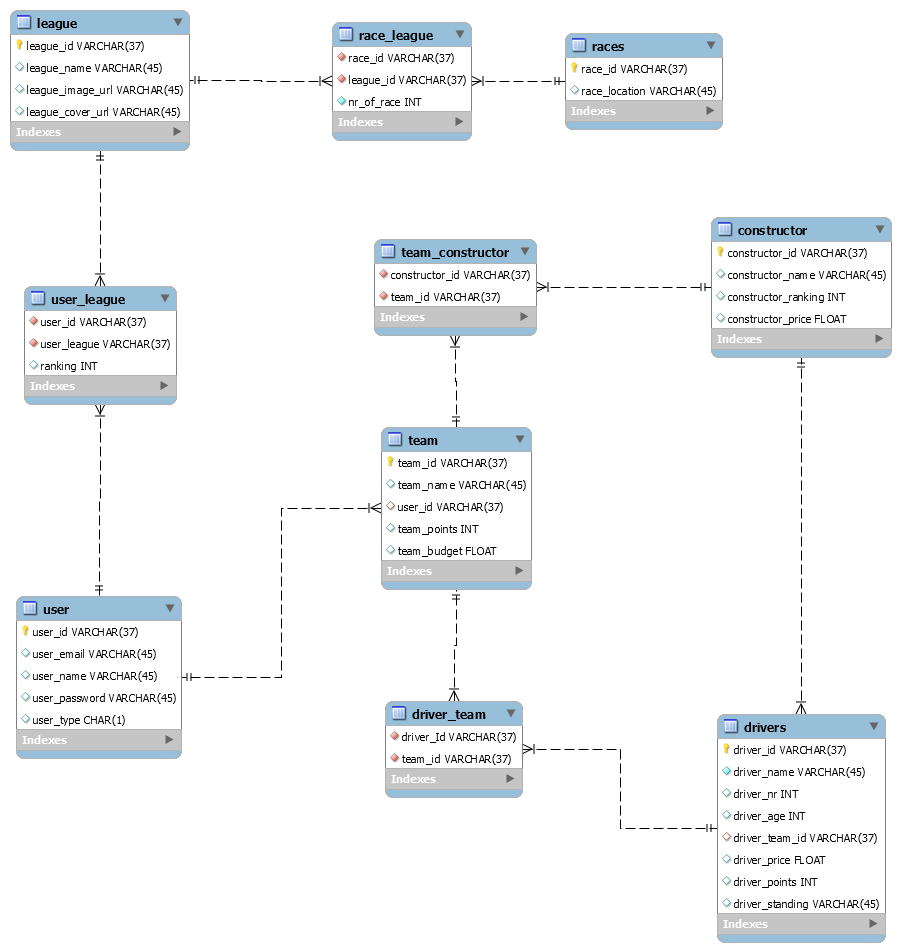
**3.2) Diagrama de pachete**

****

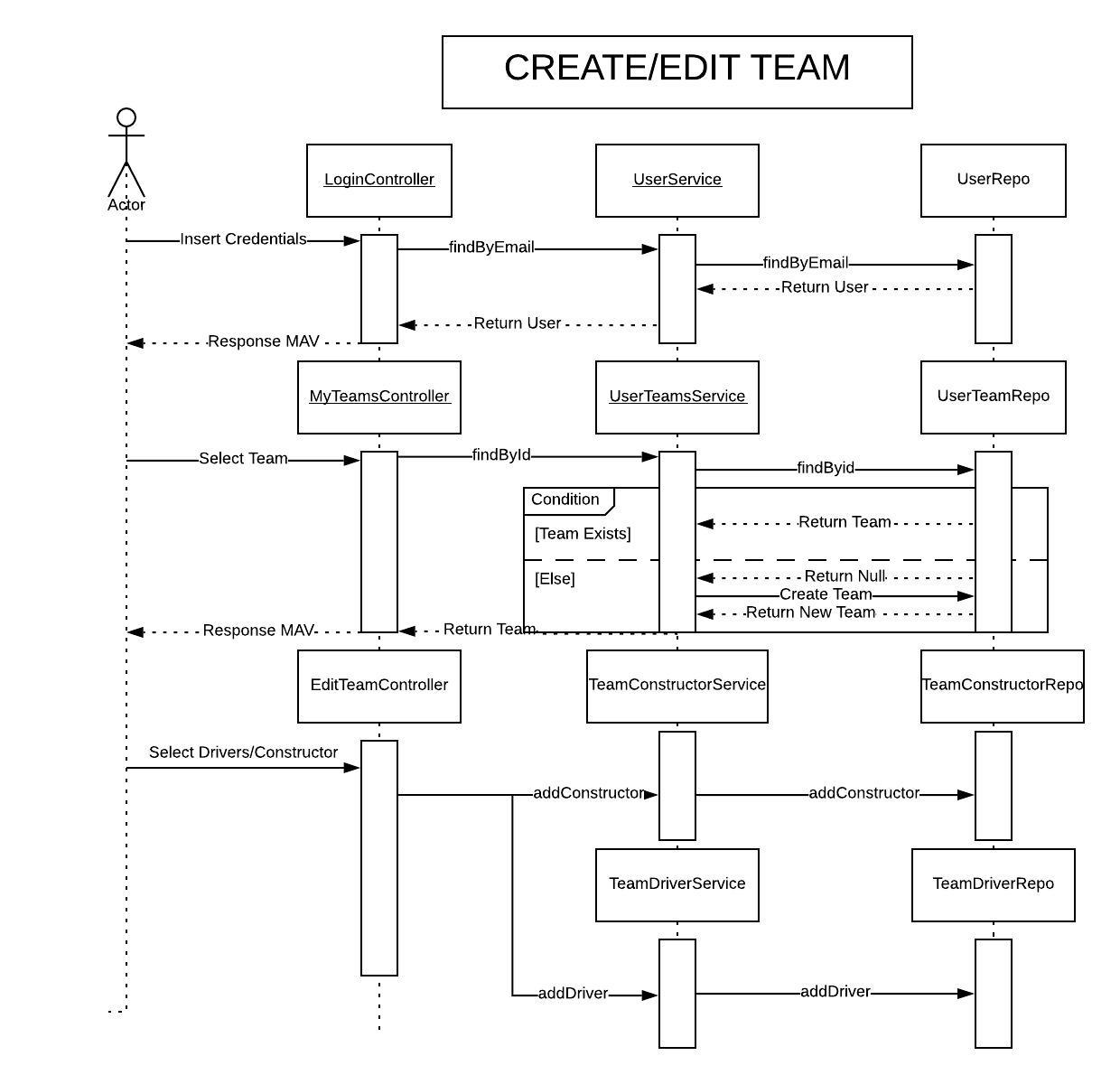
**3.3) Diagrama de deployment**

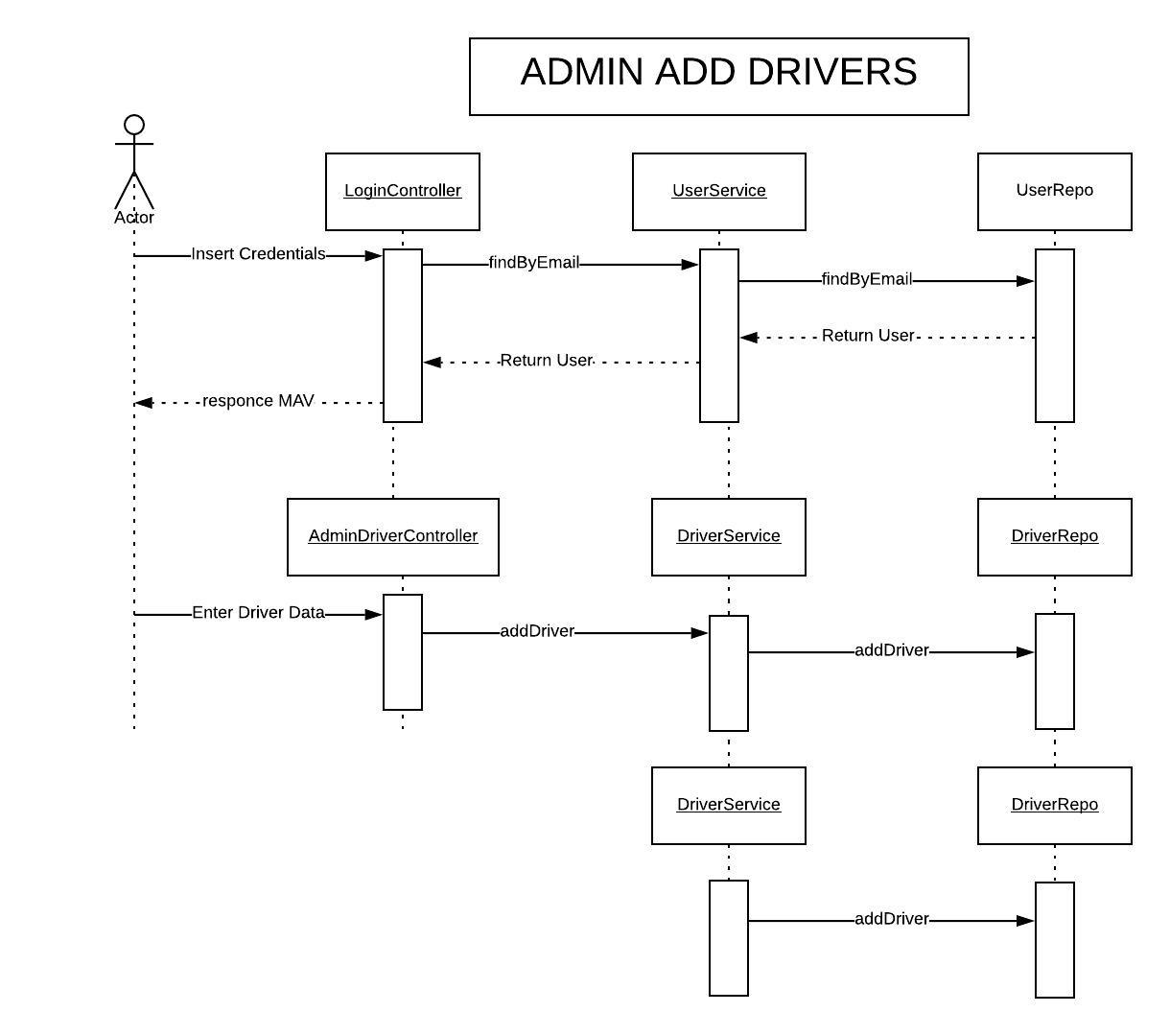
****

**3.4) Diagrama bazei de date**

****

**3.5) Diagrame de secventa**

****

**3.6) Diagrame de activitati**

**4) Specificatii suplimentare**

**4.1 Specificatii non-functionale**

Specificatiile non-functionale reprezinta criteriile de integritate ale datelor, dar si aspecte ce influenteaza performanta aplicatiei, securitatea acesteia, recuperabilitatea din erori.

*Scalabilitate:* Aplicatia trebuie sa functioneze normal atunci cand creste numarul de utilizatori ce o folosesc in acelasi timp.

*Capacitate:* Cat poti stoca in baza de date a aplicatiei.

*Integritatea datelor:* Toate datele trebuie sa respecte anumite reguli impuse de catre programator. Programul trebuie sa fie capabil sa recunoasca incalcarea acestor reguli si sa reactioneze la aparitia lor.

*Timpul de raspuns al aplicatiei:* Aplicatia trebuie sa aiba un timp de raspuns cat mai mic pentru a putea fi pe placul utilizatorilor.

*Securitatea aplicatiei:* Atributele aplicatiei trebuie sa fie ascunse de catre utilizator pentru a evita un posibil atac. O modalitate de a pastra confidentialitatea datelor este declararea variabilelor ca private, iar accesul la acestea se face prin intermediul unor gettere.

*Recuperabilitatea:* Programul trebuie sa fie capabil sa se intoarca la starea in care era inainte de a intampina o eroare.

**4.2 Constrangeri de design**

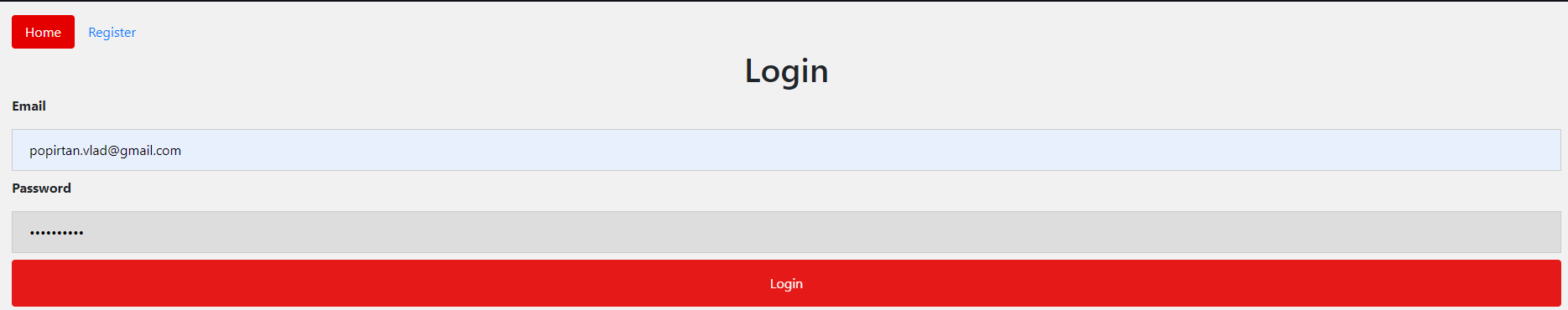
1. *Timp -* Site-ul trebuie sa fie gata in maxim 4 luni, pentru a putea fi folosit odata cu inceperea noului sezon de Formula 1.
2. *Buget –* Putem considera un buget destul de mare, deoarece site-ul este sponsorizat de FIA(Federation Internationale de l’Automobile)
3. *Marimea echipei –* Echipa ar trebui construita dnitr-o echipa mica, de maxim 10 oameni intre care sa existe un fotograf pentru pozele pilotilor, masinilor etc.
4. *Design minimalist –* Design-ul nu trebuie sa fie foarte colorat si obositor. Acesta trebiue sa respecte culorile Formula 1: rosu si alb, iar cardurile pilotilor trebuie sa aiba pe fundal culoarea echipei pentru care piloteaza.
5. *Materiale –* Echipa are permisiunea sa foloseasca orice se afla sub drepturile de autor FIA
6. *Adaptabilitate –* Informatiile despre piloti sa poata fi modificate in timpul sezonului

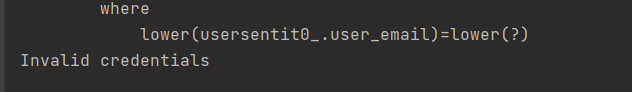
**5) Testare**

**5.1 Testarea functionalitatii aplicatiei**

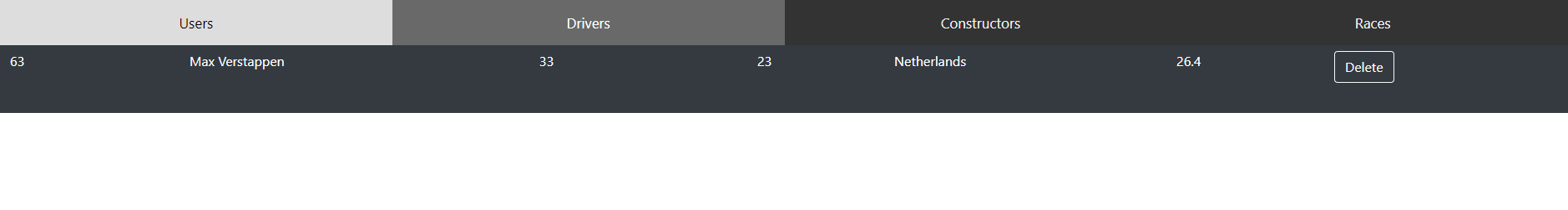
Aplicatia a fost testata manual, incercand urmatoarele operatii:

* Login folosind credentiale invalide

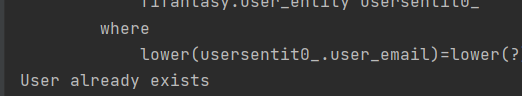




* Adaugarea unor piloti noi



* Inregistrarea unui utilizator al carui email exista deja in baza de date



**5.2 Dezvoltari ulterioare**

* Posibilitatea de a cumpara bilete pentru a participa la grand prix-uri.
* Posibilitatea de a vedea inregistrari ale grand prix-urilor pentru utilizatorii premium.
* Adaugarea de giveawayuri unde utilizatorii pot castiga: conversatii cu pilotii preferati, materiale promotionale din partea echipelor sau bilete la urmatoarele grand prix-uri.
* Aplicatie pentru android.

**6) Bibliografie**

* <https://fantasy.formula1.com> – Design
* <https://www.youtube.com/watch?v=-IMvhBWwQrc&t=1949s> - Thymeleaf
* <https://www.w3schools.com> – HTML & CSS