Platforma de invatare a informaticii si exersarea problemelor de algoritmica

Name: Branet Tudor-Andrei
Group: 30234

Table of Contents

Deliverable 1

Project Specification

Proiectul consta in realizarea unei platforme de invatare a informaticii, dedicata in special elevilor care doresc sa isi consolideze abilitatile de programare. Aceasta va include atat resurse teoretice, cat si un set larg de exercitii si probleme, structurate in diferite categorii si pe diferite niveluri de dificultate. Elevii pot selecta problema pe care doresc sa o rezolve, isi pot incarca propria solutie si primesc feedback in mod automat, sub forma unui punctaj (in functie de numarul de teste care au fost trecute). De asemenea, profesorii pot utiliza acest website pentru gestionarea mai usoara a clasei, putand de asemenea sa asigneze teme elevilor.

Functional Requirements

Cerintele functionale sunt:

- Aplicatia trebuie sa permita utilizatorului sa vizualizeze lista de probleme si sa acceseze oricare dintre acestea;
- Aplicatia trebuie sa permita utilizatorului sa isi incarce propria solutie la problema;
- Sistemul trebuie sa verifice solutia incarcata de utilizator si sa ii afiseze rezultatul acestuia;
- Profesorii trebuie sa poata sa isi creeze propriile clase;
- Aplicatia trebuie sa permita profesorilor sa creeze teme in cadrul claselor;
- Profesorii trebuie sa poata propune propriile probleme;
- Elevii trebuie sa se poata alatura unei clase prin intermediul unui ID si a unei parole;
- Administratorii trebuie sa poata sa vizualizeze statistici precum: numarul total de utilizatori, numarul de utilizatori inregistrati in ultima luna etc. in format XML sau TXT;
- Administratorii trebuie sa poata accepta sau respinge o problema propusa de un profesor;
- Utilizatorii se pot provoca unii pe ceilalti sa rezolve o anumita problema intr-un anumit interval de timp. Utilizatorul care incarca cea mai buna solutie (cu cel mai mare punctaj) la acea problema va primi un anumit numar de puncte stabilit de la inceput;
- Utilizatorii pot vedea 10 din userii cu cele mai multe puncte;
- Adminii pot vedea activitatea userilor (login si logout);
- Utilizatorii se pot abona la un newsletter si sa primeasca ocazional diferite emailuri (in care se mentioneaza cel mai activ utilizator din ziua curenta etc.)

Use Case Model

Use Cases Identification

1) **Use-Case: cautare problema dupa ID**

Level: user-goal

Primary Actor: elev

Main success scenario: elevul introduce ID-ul problemei dorite in field-ul de cautare de pe pagina cu toate problemele, iar in cazul in care exista o problema cu ID-ul introdus, atunci doar aceea va fi afisata.

Extensions: daca problema cu ID-ul introdus nu exista, se va afisa un mesaj de eroare

2) **Use-Case: crearea unei clase**

Level: user-goal

Primary Actor: profesor

Main success scenario: profesorul introduce numele clasei si parola de acces

Extensions: daca parola introdusa nu este valida, se va afisa un mesaj de eroare

3) **Use-Case: aprobarea unei probleme propuse**

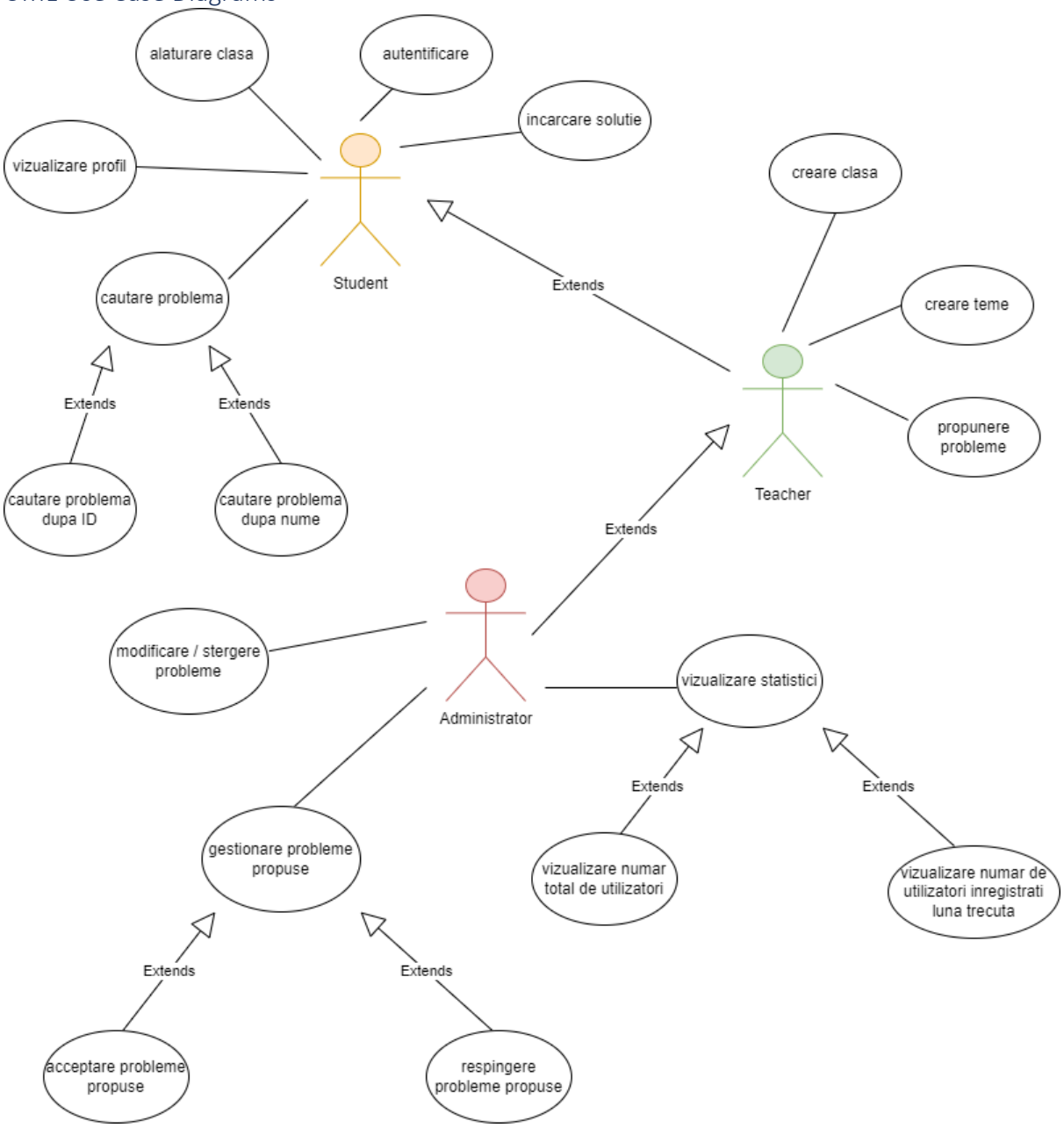
Level: user-goal

Primary Actor: administrator

Main success scenario: administratorul acceseaza lista cu probleme in asteptare, selecteaza optiunea de acceptare a problemei, iar apoi problema va fi vizibila de catre toti utilizatorii

Extensions: -

UML Use Case Diagrams



Supplementary Specification

Non-functional Requirements

1) Aplicatia sa fie usor de folosit si sa ofere o experienta placuta utilizatorului;
2) Timpul de raspuns – Incarcarea paginilor trebuie sa se efectueze intr-un timp cat mai scurt;
3) Fiabilitate – Site-ul trebuie sa fie disponibil si functional in mod constant. Astfel, trebuie sa fie testat in profunzime inainte de a fi public si trebuie folosite solutii de back-up pentru date;
4) Scalabilitate – Site-ul trebuie sa functioneze in parametri optimi (performanta si disponibilitatea nu trebuie sa fie afectate) in momentul in care se doreste cresterea complexitatii si implicit cresterea numarului de utilizatori. Astfel, operatiile pe baza de date trebuie sa se efectueze cat mai rapid, iar arhitectura trebuie sa poata fi extinsa simplu si eficient;
5) Securitate – Protectia datelor impotriva amenintarilor cibernetice. In acest sens, parolele trebuie sa fie criptate pentru a preveni atacatorii sa preia date la care in mod normal nu ar avea acces.

Design Constraints

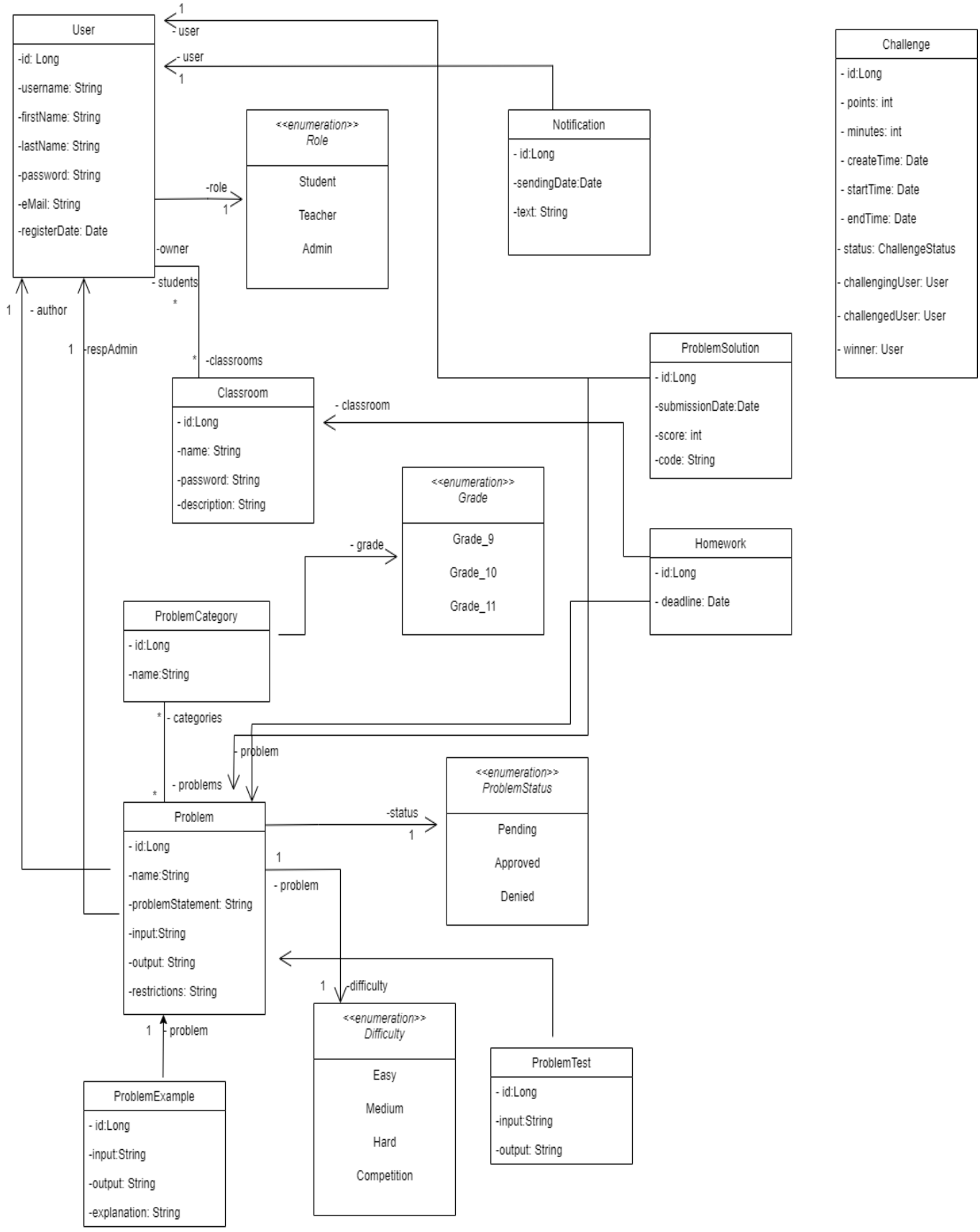
Constrangeri de proiectare: Se va folosi framework-ul Spring Boot pentru backend-ul aplicatiei. Este flexibil, rapid, si ofera un grad ridicat de securitate. Este construit dupa design pattern-ul MVC, iar limbajul folosit este Java. Ca si mediu de programare se foloseste IntelliJ, iar ca si tool se utilizeaza Maven. Acesta se ocupa de project management, in special de gestiunea dependentelor. Sistemul de gestiune a bazei de date este MySQL.

Glossary

- ➔ Spring Boot - un framework open-source pentru dezvoltarea de aplicatii Java;
- ➔ Maven - o unealta de gestionare a dependentelor si de construire a proiectelor Java;
- ➔ MySQL – sistem de gestiune a bazelor de date relationale;
- ➔ Model-View-Controller (MVC) – un design pattern arhitectural pentru dezvoltarea de aplicații software care separa logica, interfata utilizator si gestionarea datelor in trei componente diferite.

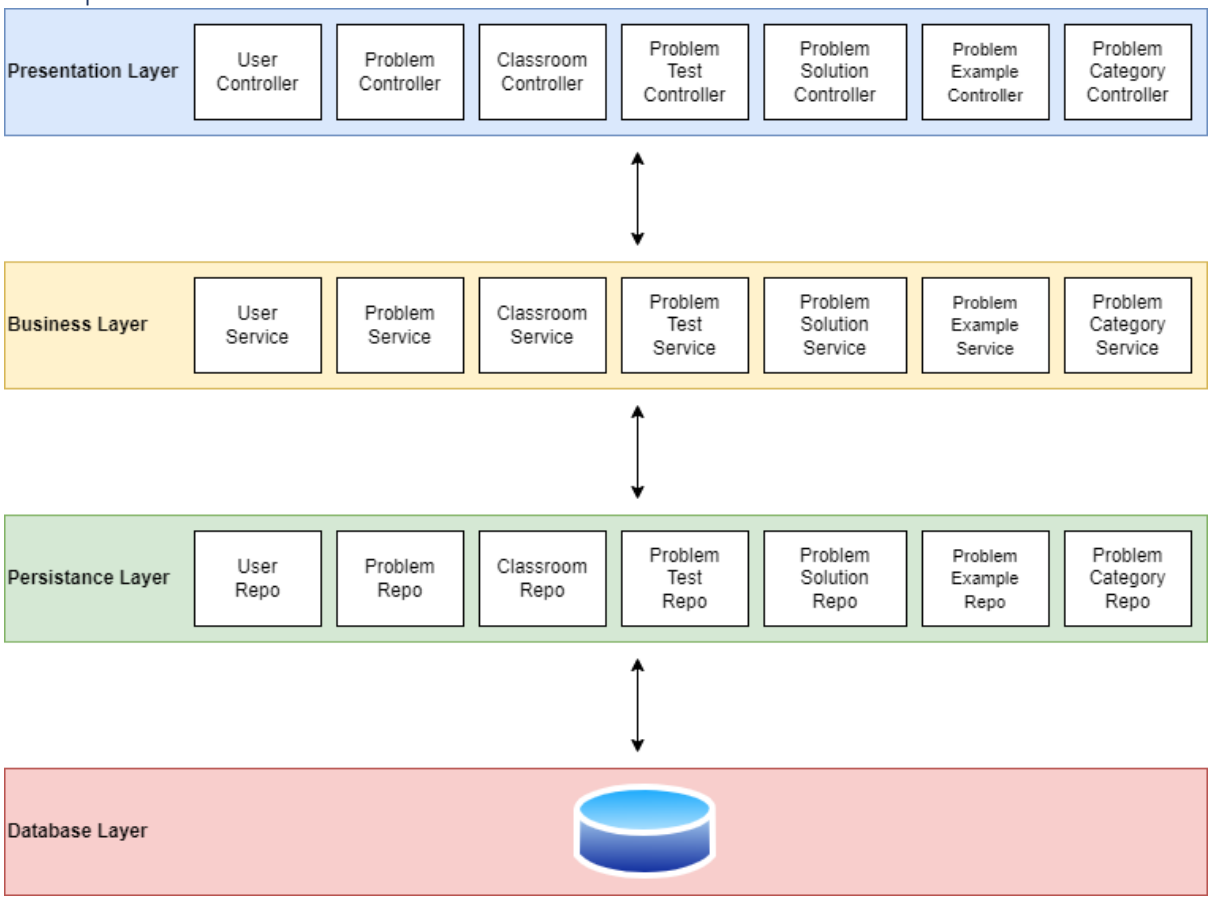
Deliverable 2

Domain Model



Architectural Design

Conceptual Architecture



Aplicatia este structurata in 4 layere, si anume:

1) Presentation Layer

Acest tip de layer se situeaza la nivelul cel mai inalt al arhitecturii unei aplicatii Spring Boot si se ocupa cu gestiunea request-urilor HTTP si realizarea autentificarii. E responsabil pentru convertirea campurilor JSON in obiecte Java (si invers) si trimiterea lor la urmatorul layer.

2) Business Layer

Acest layer contine toata logica de business, fiind alcatuit din clasele de service.

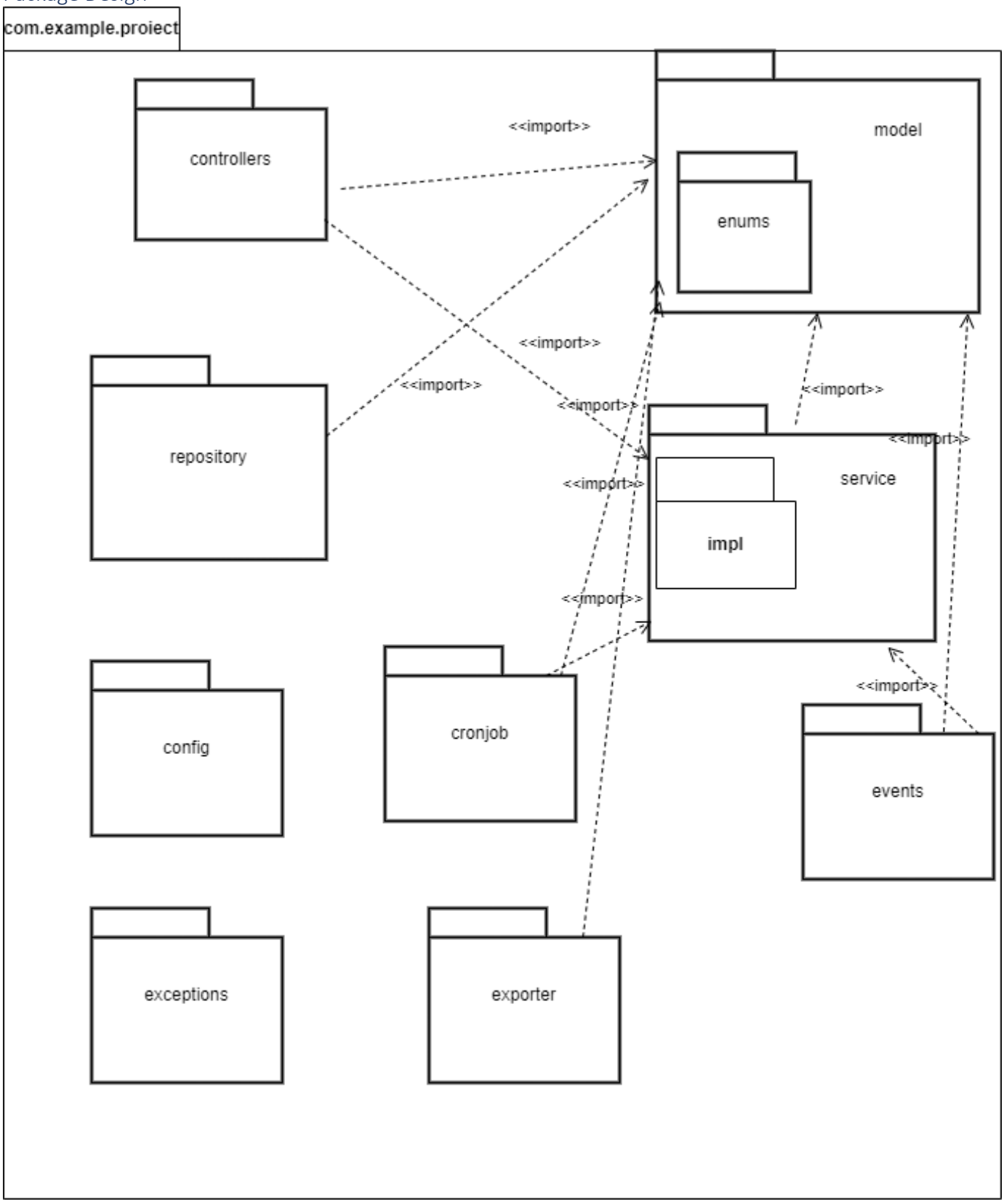
3) Persistence Layer

Persistance layer contine logica pentru gestiunea bazei de date si este alcatuit din clasele de repository.

4) Database Layer

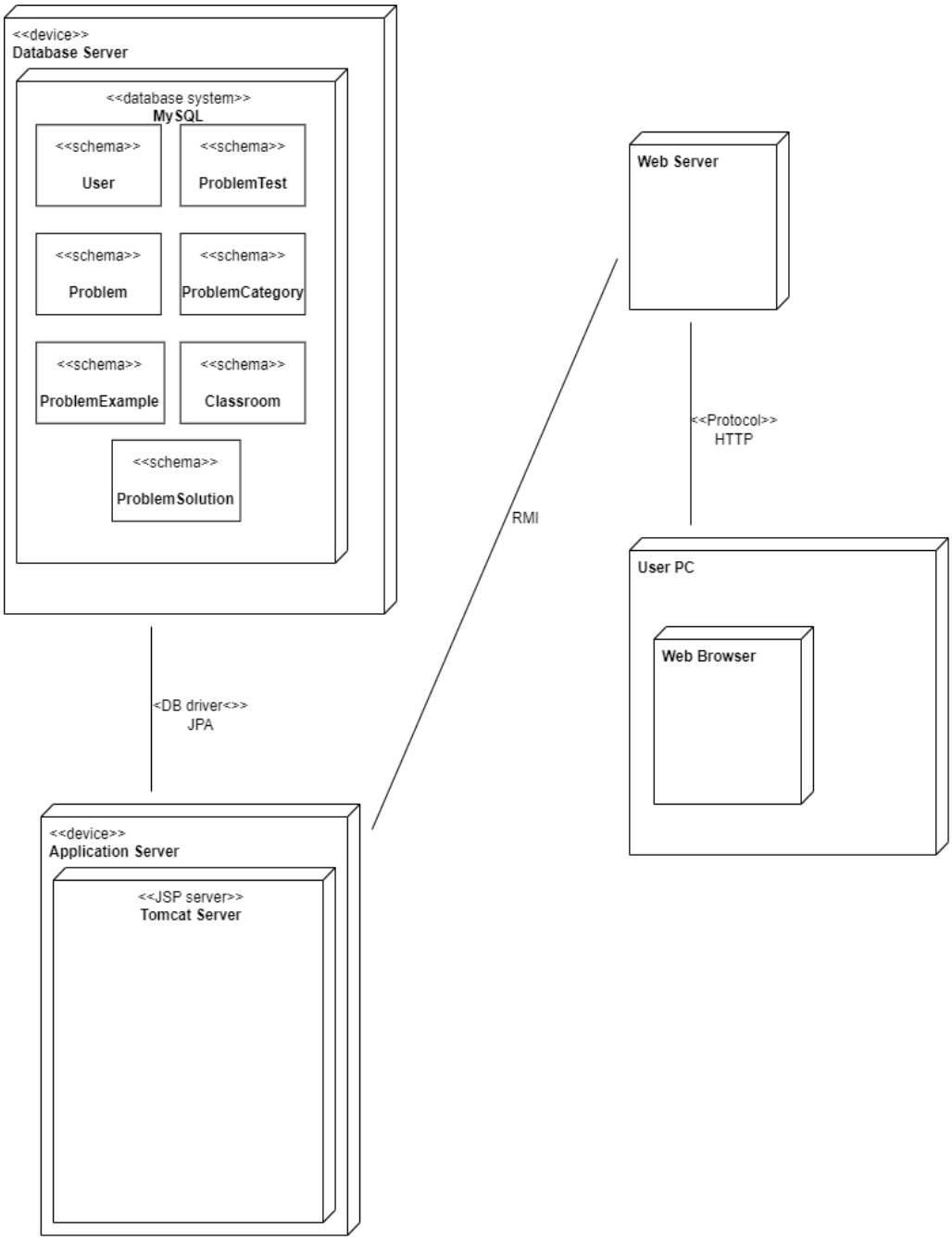
Acest layer contine baza de date propriu-zisa, si anume MySQL.

Package Design

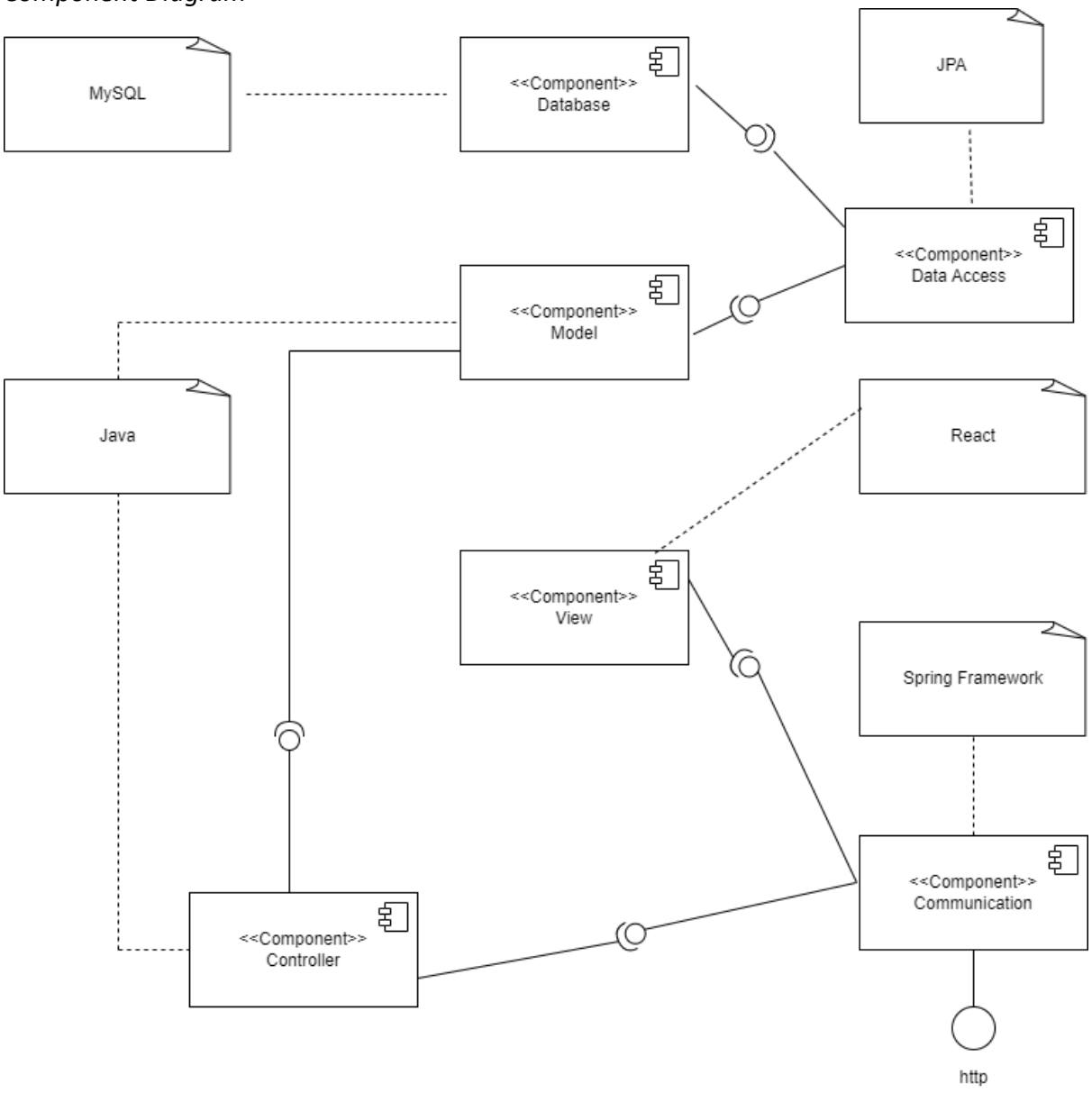


Component and Deployment Diagram

Deployment Diagram



Component Diagram

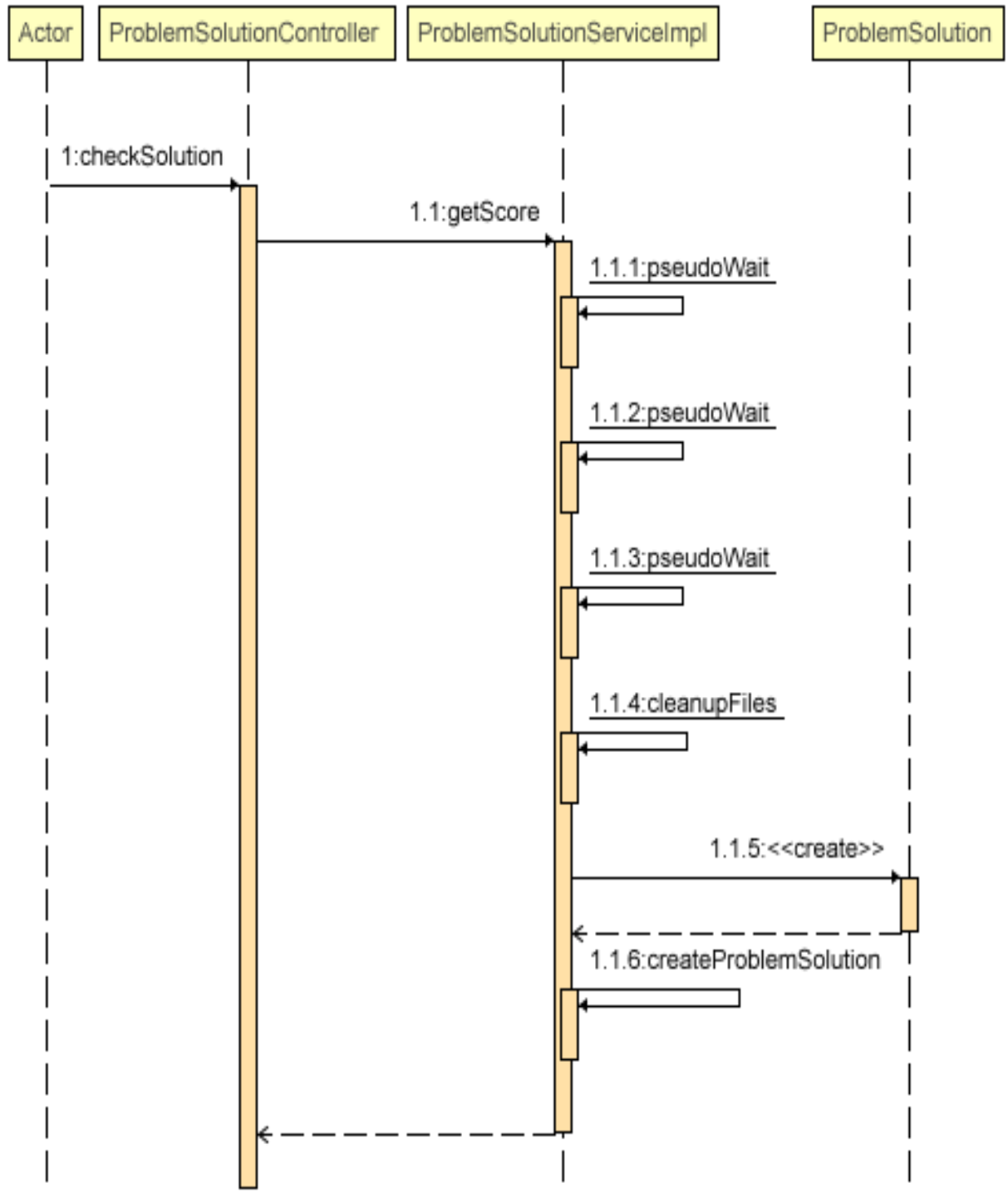


Deliverable 3

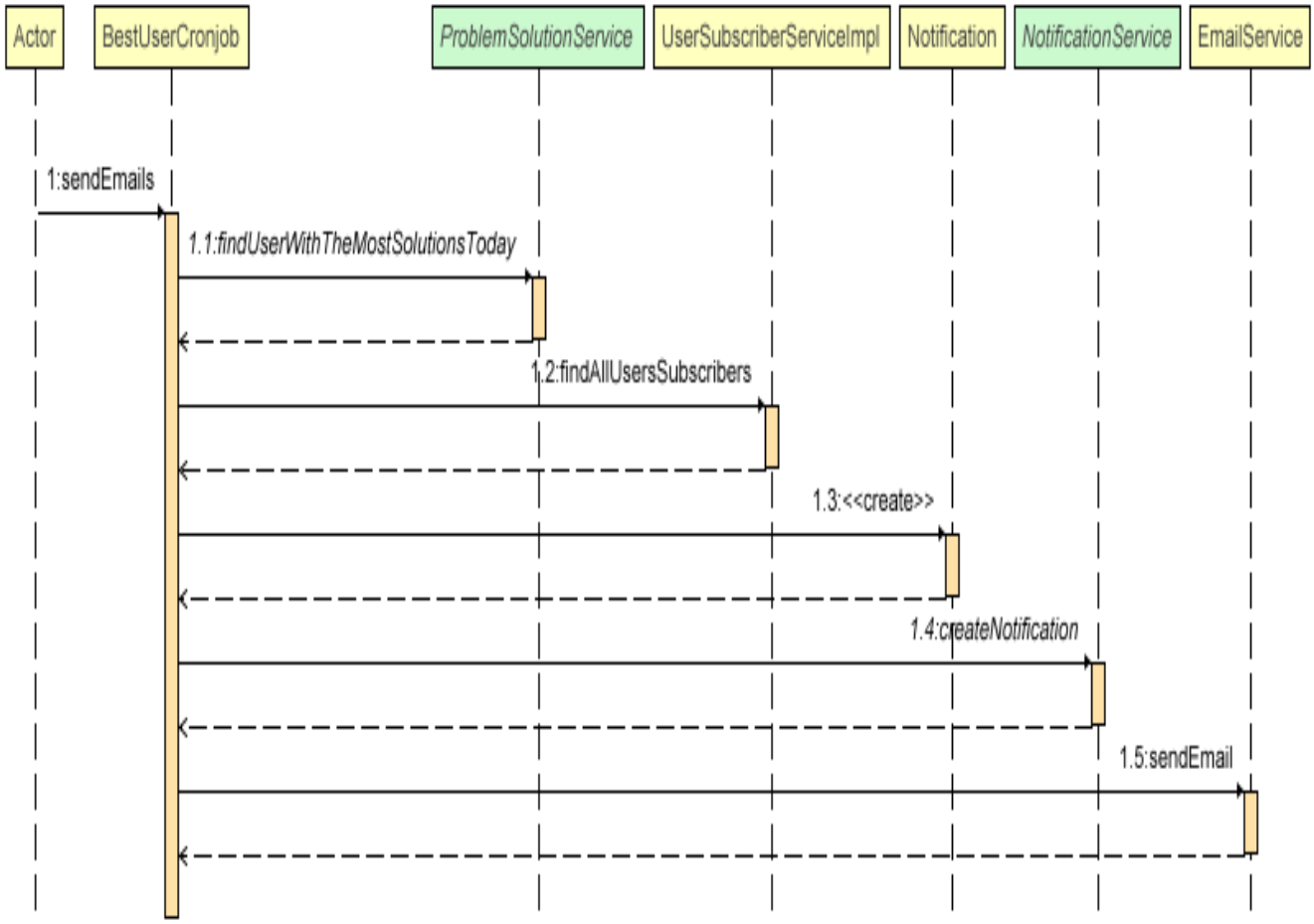
Design Model

Dynamic Behavior

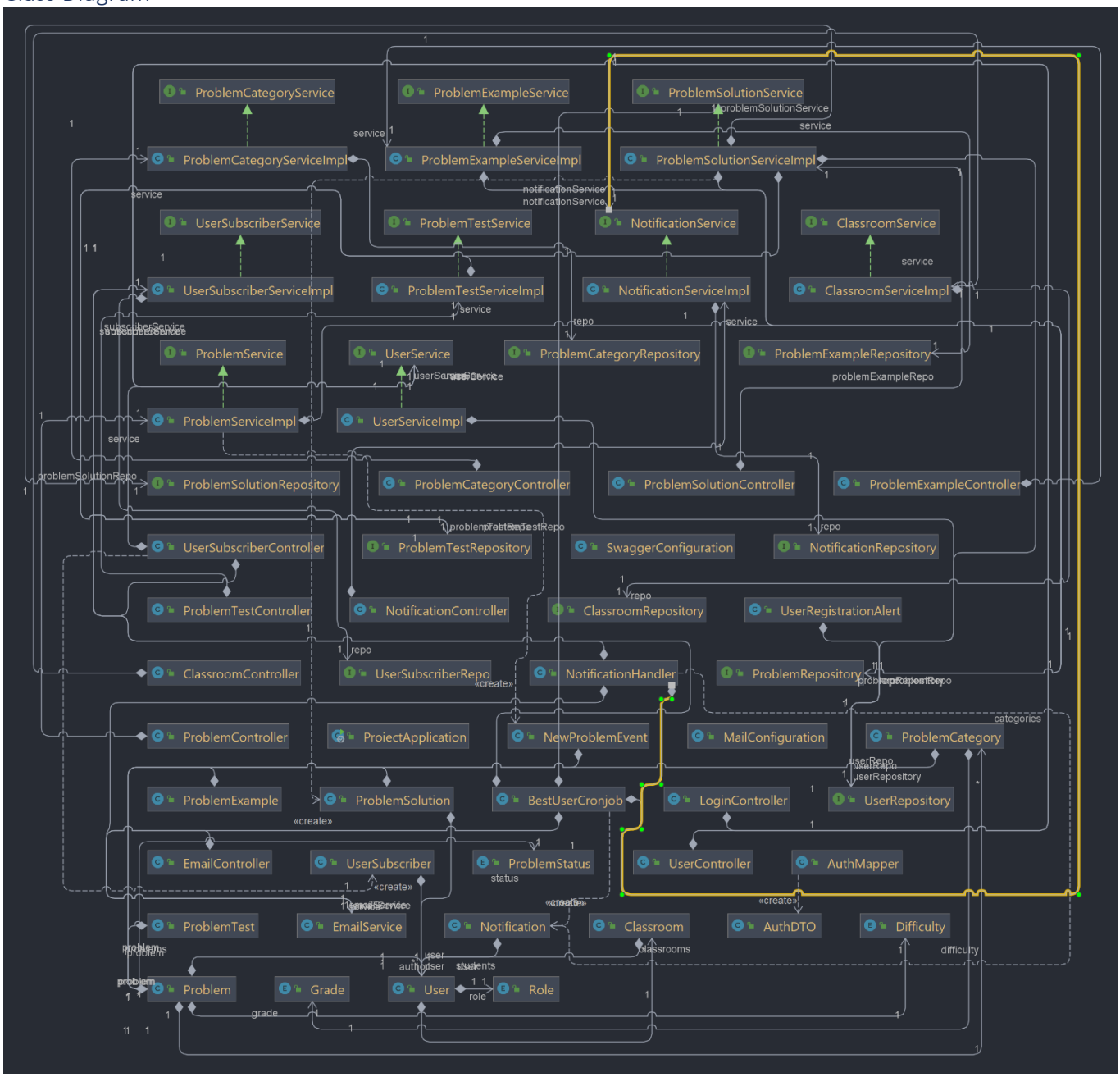
a) Diagrama de secventa pentru obtinerea punctajului aferent unei solutii incarcate de un utilizator la o anumita problema.



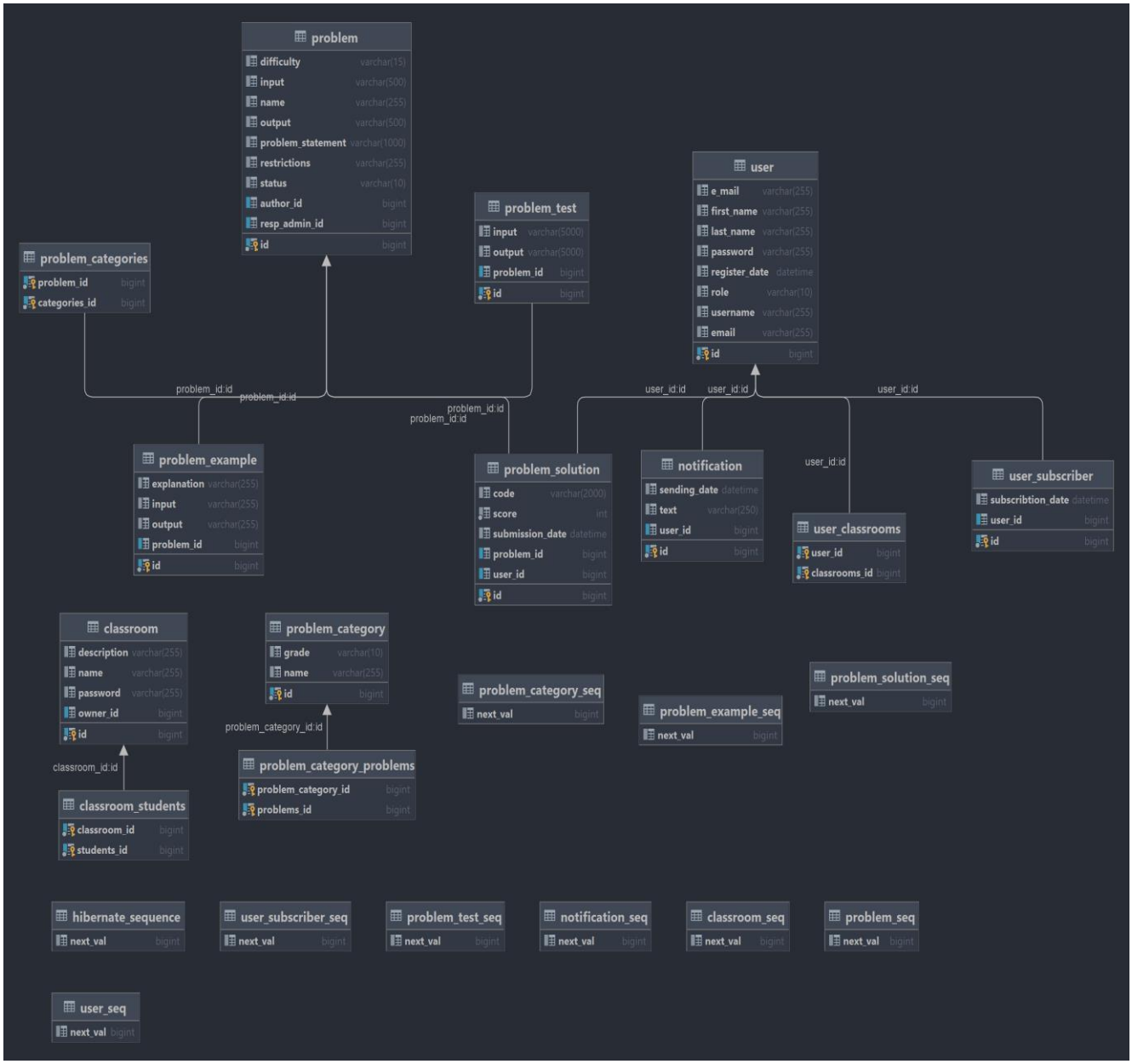
b) Diagrama de secventa pentru trimiterea tuturor abonatilor a unui email zilnic si a unei notificari in care este specificat utilizatorul care a rezolvat cele mai multe probleme corect in acea zi.



Class Diagram



Data Model



System Testing

Test case 1: givenExistingUsername_whenFindByUsername_thenFindOne()

- ➔ Acest caz de testare verifica faptul ca, daca se furnizeaza un nume de utilizator valid, atunci metoda findUserByUsername() din clasa UserServiceImpl returneaza un obiect User nenul cu numele de utilizator corect.

Test case 2: givenNonExistingUsername_whenFindByUsername_thenThrowException()

- ➔ Acest caz de testare verifica faptul ca, daca se furnizeaza un nume de utilizator neexistent, atunci metoda findUserByUsername() din clasa UserServiceImpl arunca o exceptie de tipul NullPointerException.

Test case 3: givenExistingUsername_whenDeleteByUsername_thenUserIsDeleted()

➔ Acest caz de testare verifica faptul ca, daca se furnizeaza un nume de utilizator valid, atunci metoda deleteUserByUsername() din clasa UserServiceImpl sterge obiectul User corespunzator din repository.

Test case 4: givenValidUser_whenCreateUser_thenUserIsCreated()

➔ Acest caz de testare verifica faptul ca, daca se furnizeaza un obiect de tipul User valid, atunci metoda createUser() din clasa UserServiceImpl creeaza un nou obiect de tipul User în repository.

Future Improvements

- Utilizatorii sa poata comunica intre ei prin intermediul unui chat;
- Sa existe mai multe limbaje posibile in care sa se poata scrie solutiile problemelor;
- Posibilitatea recuperarii parolei prin trimiterea unui cod prin email;
- Utilizatorii sa poata permite altor utilizatori sa le vizualizeze solutiile;
- Spring security

Conclusion

In concluzie, in urma realizarii acestui proiect, am invatat sa construiesc o aplicatie web folosind Java Spring Boot (backend) si React (frontend) si sa aplic diferite principii de proiectare software.

Bibliography