Enunt:

Sa se implementeze o aplicatie Web pentru realizarea de prognoze privitoare la punctajele obtinute de studenti la probele de evaluare -- teste scrise, activitati de laborator, proiecte etc. -- a cunostintelor la anumite discipline (e.g., Ingineria programarii, Practica SGBD, Tehnologii Web). Pentru fiecare runda a prognozelor, un utilizator (eventual, autentificat) va putea "ghici" doar o singura data nota pe care o va lua. In cazul in care nota precizata de acesta va coincide cu cea reala (preluata dintr-un document CSV, JSON sau XML separat), va primi P unitati in plus la punctaj, in caz contrar va obtine M unitati in minus. Dupa un numar de R runde, se vor putea afisa punctajele finale ale tuturor celor evaluati. Schimbarile de situatie vor fi disponibile via un flux RSS. Situatiile cu punctaje vor fi disponibile, de asemenea, ca documente adoptand formatele CSV si PDF.

LOGIN FLOW

1. **Login Model**

- **O clasa,** cu**:**
    1 . Metoda pentru **login** - care verifica credentialele introduse de user
        a. **Daca** sunt corecte, se creeaza un token pentru user
            i. Intai se face SELECT din baza de date la username si parola, pentru a vede ca ele coincid cu cele din $_POST["username"], $_POST["parola"]
            ii. Credentiale corecte → INSERT intr-o tabela de Token'uri, de (token, username)

        b. **Altfel**, return False
    2. Metoda pentru **logout -** destroy session

2. **Views** pentru login:
- login.view.php
- login_failed.view.php
- home.view.php

3. **Login Controller**

- session_start()
- Daca **HTTP Request Method** este:
    1. **GET**
        a. **Daca exista** un token ($_SESSION["token"]) -- redirectionare catre "/home.view.php"
        b. **Daca nu exista** si variabila de sesiune $_SESSION["failed_login"] nu este NULL --
            i. elimin variabila $_SESSION["failed_login"]
            ii. include_once "../views/login_fail.view.php";
        c. Altfel--
            i. include_once "../views/login.view.php";

    2. POST
        a. Include_once "../models/login.model.php";
        b. **Daca** se apasa butonul de login ($_POST["login_submit"]) --
            i. se creeaza un token pentru user, dupa verificarea cu succes a credentialelor
            ii. **Daca** user'ul se poate loga, retin in variabila de sesiune $_SESSION["token"] token'ul si il redirectionez spre pagina principala: header("Location: /home");

iii. **Daca** user'ul nu se poate loga, setez $_SESSION["login_fail"] = TRUE

REGISTER FLOW

1. **Model**
   - O **clasa** cu (sau in acceasi clasa cu Login.model?)**:**
     1. o metoda pentru register(parametri: campurile din form)
        a. (?)**Se verifica** daca nr matricol introdus, este in baza de date (asta presupune ca anterior am avea toate nr matricole ale studentilor, intr-un tabel, continand nr_matricol, nume, grupa, an …)
        b. **Daca** nr matricol este in baza de date, inseamna ca se poate crea contul
           i. **Daca** username'ul/e-mail'ul introdus este deja in baza de date, inseamna ca exista deja un cont pentru persoana respectiva → return False
           ii. **Finally** se face INSERT in baza de date, pentru toate informatiile introduse in register form: nume, prenume, nr. matricol, username, parola, verificare parola, email
           iii. **Se creeaza** un token pentru user: INSERT intr-o tabela de Token'uri, de (token, username), **return token**
2. **Views** pentru login:
   - register.view.php
   - register_fail.view.php
   - home.view.php

3. Controller

- session_start()
- Daca **HTTP Request Method** este:

2. GET

a. **Daca exista** un token ($_SESSION["token"]) -- redirectionare catre "/home.view.php"
b. **Daca nu exista un token** si variabila de sesiune $_SESSION["failed_register"] este TRUE --
   i. elimin variabila $_SESSION["failed_register"]
   ii. include_once "../views/register_fail.view.php";
c. Altfel
   i. include_once "../views/register.view.php";

2. POST

a. include_once "../models/register.model.php";

b. **Daca** se apasa butonul de “Inregistreaza-te” ($_POST["register_submit"]) --

iv. se creeaza un token pentru user, dupa verificarea cu **succes** a credentialelor (SignUp::register($_POST["nr_matricol"], ...))

v. **Daca** user’ul se poate inregistra(valoarea returnata de metoda nu este False), retin in variabila de sesiune $_SESSION["token"] token’ul si il redirectionez spre pagina principala: header("Location: /home");

vi. **Daca** user’ul nu se poate inregistra, setez $_SESSION["register_fail"] = TRUE si ramane pe pagina curenta

c. **Daca** se apasa butonul de “Anuleaza” ($_POST["register_cancel"])

i. Ramane pe pagina curenta

RUNDE FLOW

1. Model (round.model.php)

Clasa Round care contine:

1. **O metoda choose_grade()**, prin care un user autentificat isi poate alege nota pe care crede ca o va obtine.

a. **Functia va returna FALSE** in momentul in care studentul va incerca sa isi mai “Ghiceasca” inca o data nota (se face un SELECT din tabelul PREDICTION, daca se aduc date cu campul choose_grade diferit de NULL, inseamna ca studentul a mai facut o prognoza si nu mai are voie sa ghiceasca o nota)

b. **Functia va face INSERT** in tabelul PREDICTION, cu registration_number, nota, numele materiei, daca runda este ACTIVA/INACTIVA plus celelalte campuri

2. **A doua metoda get_results()** se ocupa cu gestionarea rezultatelor.

a. **Functia va returna FALSE** daca nu sunt rezultate disponibile
b. **Altfel returneaza path-ul** catre un document google docs (uploadat de catre admin), care va avea permisiuni de citire

3. **Functia check_round** se ocupa cu verificarea rundei, pentru a vedea daca este activa/inativa

2. **Controller (round.controller.php)**

Daca **HTTP Request Method** este:

a. GET

- daca $_SESSION[‘results’] = FALSE, atunci redirectioneaza user’ul spre “./comingsoon.view.html”
- daca $_SESSION[‘results’] =TRUE, atunci se redirectioneaza spre “./docs.google….”
- daca $_SESSION[‘prediction_done’] = TRUE ramane pe “./first_course.view.html” dar nu va fi butonul de **Submit** pe pagina
- daca $_SESSION[‘prediction_done’] = FALSE ramane pe “./first_course.view.html” dar va aparea si butonul de **Submit** pe pagina

b. **POST**

- include first_course.view.html
- daca se face **$_POST[‘lab’] sau $_POST[‘course’]**

    - **Daca** runda este **activa** (Round::**check_round()** returneaza **TRUE**):
        - **daca** se face $_POST[‘grade_submit’’], se apeleaza Round::choose_grade(params) si $_SESSION[‘prediction_done’] = TRUE

    - **Daca** runda este **inactiva** (Round::check_round() returneaza **FALSE**):
        - $_SESSION[‘prediction_done’] = FALSE

        - **daca nu sunt rezultate** uploadate:
            - $_SESSION[‘results’] = FALSE
        - **daca sunt rezultate** uploadate si se face $_POST[‘aici’’]:
            - $_SESSION[‘results’] =TRUE
            - Round::get_results(params)

3. **Views**

- comingsoon.view.html
- first_course.view.html