

Aplicație tip Moodle

*Inginerie Software*

### Autor: Hîrlav Marin-Andrei Grupa: 30237

### Facultatea de Automatica si Calculatoare

### 16 Noiembrie 2025

**Cuprins**

1. Introducere ............................................................... pag. 3-4
2. Funcționalitățile aplicației ........................................ pag. 5-6
3. Diagrama Use-Case .................................................. pag. 7
4. Diagrama de activitate .............................................. pag. 8
5. Design Patterns utilizate ........................................... pag. 9
6. Testare ...................................................................... pag. 10
7. Mod de rulare a aplicației ......................................... pag.11-12

**Introducere**

Aplicația Moodle dezvoltată este o platformă educațională web destinată gestionării cursurilor și assignmenturilor într-un mediu digital. Platforma permite utilizatorilor să își asume roluri de **profesor** sau **student**, fiecare având acces la funcționalități specifice menite să sprijine procesul educațional.

#### ****Scopul aplicației****

Principalul scop al aplicației este de a oferi un mediu digital eficient pentru colaborarea între profesori și studenți. Profesorii pot gestiona cursuri, assignmenturi și informații despre studenți, iar studenții pot accesa cursurile și assignmenturile pentru a contribui cu materialele lor.

### **Tehnologiile utilizate**

#### ****1. Java Spring Boot****

**Spring Boot** este un framework de dezvoltare backend utilizat pentru a crea aplicații Java robuste și scalabile. Acesta simplifică configurarea și permite dezvoltarea rapidă a aplicațiilor web prin:

* **Dependency Injection** – facilitează gestionarea componentelor aplicației.
* **MVC (Model-View-Controller)** – arhitectură utilizată pentru separarea logicii de prezentare.
* **Suport pentru baze de date** – include integrare cu Hibernate/JPA pentru gestionarea eficientă a datelor.
* **Securitate** – oferă mecanisme predefinite pentru autentificare și autorizare.

#### ****2. Thymeleaf****

**Thymeleaf** este un motor de template folosit pentru a genera interfața de utilizator (frontend) dinamică și integrată cu backend-ul Java. Avantaje:

* Integrare naturală cu Spring Boot.
* Suport pentru template-uri HTML, permițând separarea clară a logicii backend de vizualizarea frontend.
* Posibilitatea de a insera variabile din backend direct în pagini HTML.

### **3. Baza de date**

Aplicația utilizează o bază de date relațională pentru stocarea și gestionarea datelor esențiale, cum ar fi:

* **Utilizatori**: informații despre profesori și studenți (nume, email, roluri, autentificare etc.).
* **Cursuri**: detalii despre cursuri, inclusiv nume, descriere și profesorul asociat.
* **Assignmenturi**: sarcini asociate cursurilor, cu date despre deadline-uri și alte detalii relevante.
* **Fișiere atașate**: încărcate de studenți pentru assignmenturi.

#### ****Tehnologia utilizată****

* **MySQL** a fost aleasă ca sistem de gestionare a bazelor de date datorită fiabilității, performanței și suportului extins pentru aplicațiile Spring Boot.
* **Hibernate/JPA (Java Persistence API)** este utilizat pentru maparea obiect-relatională (ORM), oferind o modalitate simplificată de interacțiune cu baza de date. Hibernate permite maparea claselor Java către tabelele bazei de date și automatizează operațiunile CRUD (Create, Read, Update, Delete).

**Funcționalitățile aplicației**

#### ****1. Funcționalități generale****

Aplicația permite următoarele operațiuni comune pentru toți utilizatorii:

* **Autentificare și înregistrare:**
  + Utilizatorii pot crea un cont nou prin furnizarea datelor personale și alegerea unui rol (profesor sau student).
  + După autentificare, utilizatorii au acces la funcționalitățile specifice rolului lor.
* **Interfață prietenoasă:**
  + Aplicația utilizează Thymeleaf pentru a crea o interfață web intuitivă, accesibilă de pe diverse dispozitive.

#### ****2. Funcționalități pentru profesori****

Profesorii au acces la funcționalități avansate care le permit să gestioneze cursuri, assignmenturi și utilizatori:

1. **Gestionarea cursurilor:**
   * Crearea de noi cursuri, incluzând numele și descrierea acestora.
   * Editarea detaliilor cursurilor existente.
   * Ștergerea cursurilor care nu mai sunt relevante.
2. **Gestionarea assignmenturilor:**
   * Crearea de assignmenturi asociate unui curs.
   * Setarea termenelor limită pentru assignmenturi.
   * Modificarea detaliilor assignmenturilor, inclusiv numele și descrierea acestora.
   * Ștergerea assignmenturilor după finalizare.
3. **Vizualizarea utilizatorilor:**
   * Accesarea unei liste complete cu studenții înscriși în cadrul aplicației și în cursurile lor, inclusiv modificarea detaliilor acestora.
   * Vizualizarea informațiilor despre ceilalți profesori din platformă.
4. **Monitorizarea cursurilor și a activității:**
   * Vizualizarea listei cursurilor create.
   * Accesarea fișierelor încărcate de studenți în cadrul assignmenturilor.

#### ****3. Funcționalități pentru studenți****

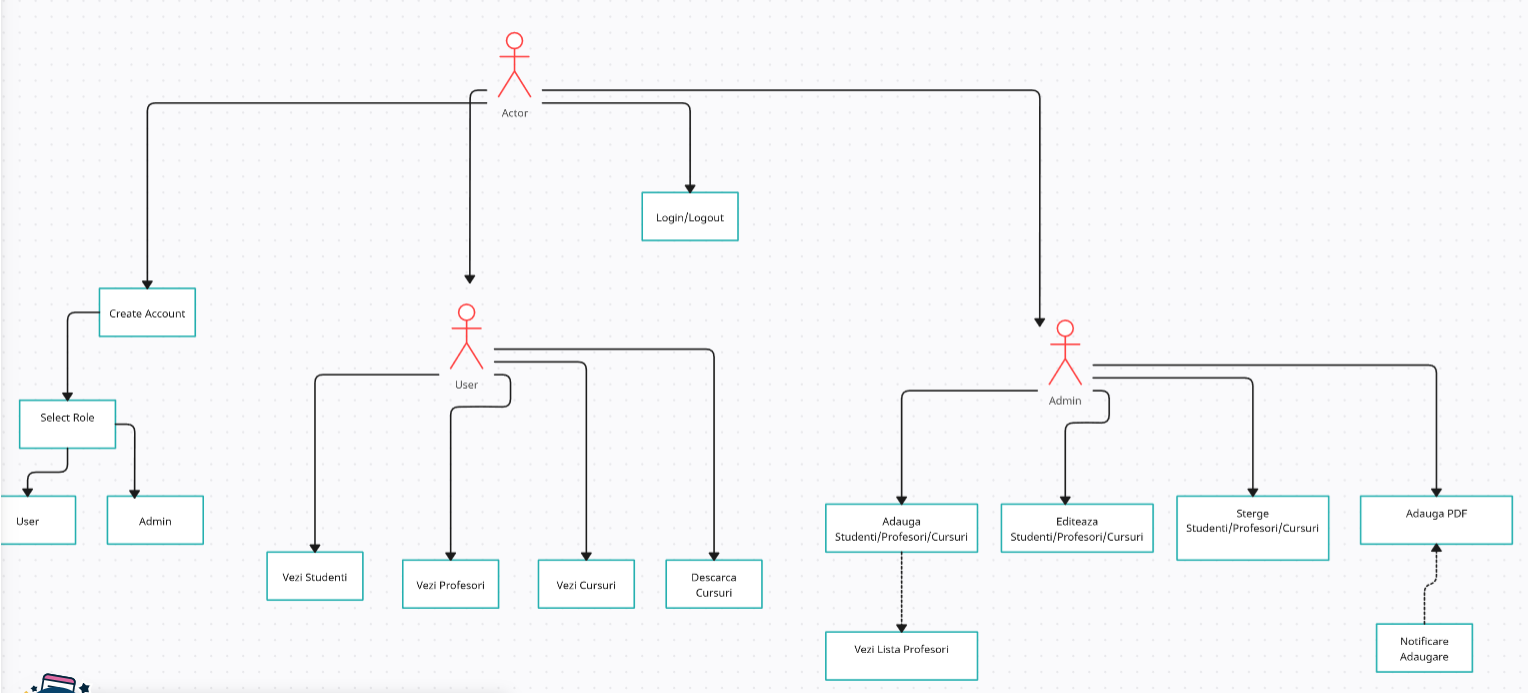
Studenții au acces la funcționalități specifice care le permit să participe activ la cursuri și assignmenturi:

1. **Accesarea cursurilor:**
   * Vizualizarea unei liste cu toate cursurile disponibile pe platformă.
   * Accesarea detaliilor cursurilor, inclusiv numele și descrierea acestora.
2. **Vizualizarea assignmenturilor:**
   * Vizualizarea listei assignmenturilor asociate cursurilor în care sunt înscriși.
   * Consultarea termenelor limită și a detaliilor assignmenturilor.
3. **Încărcarea fișierelor pentru assignmenturi:**
   * Studenții pot adăuga fișiere în cadrul assignmenturilor disponibile.
   * Fiecare fișier încărcat este asociat cu assignmentul și studentul care l-a trimis.
4. **Vizualizarea informațiilor despre profesori:**
   * Studenții pot accesa lista profesorilor pentru a vedea detalii despre aceștia și cursurile oferite de ei.

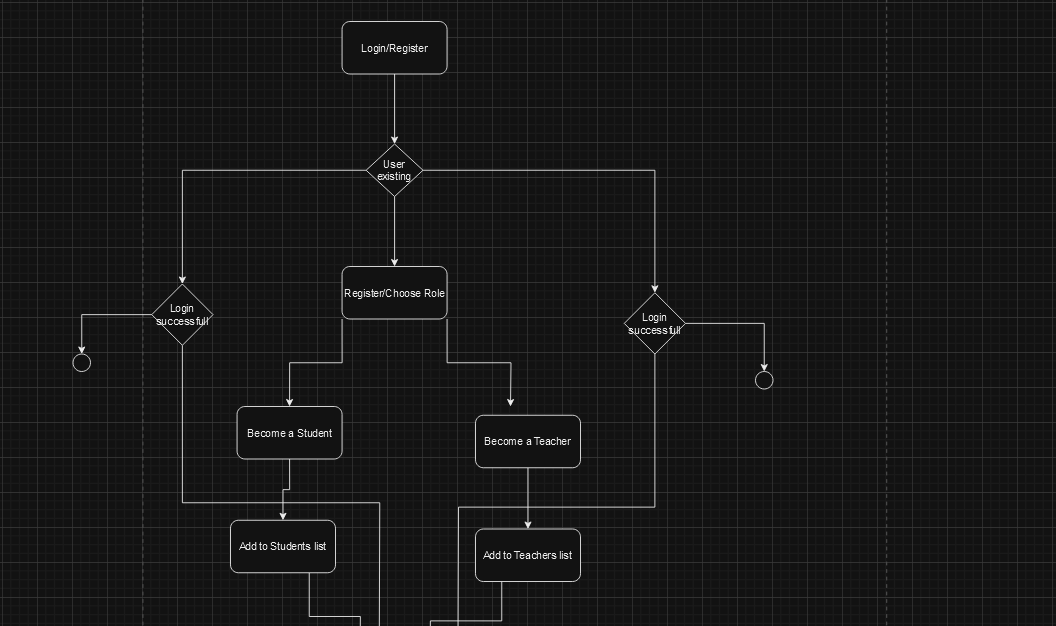
### **Fluxuri principale în aplicație**

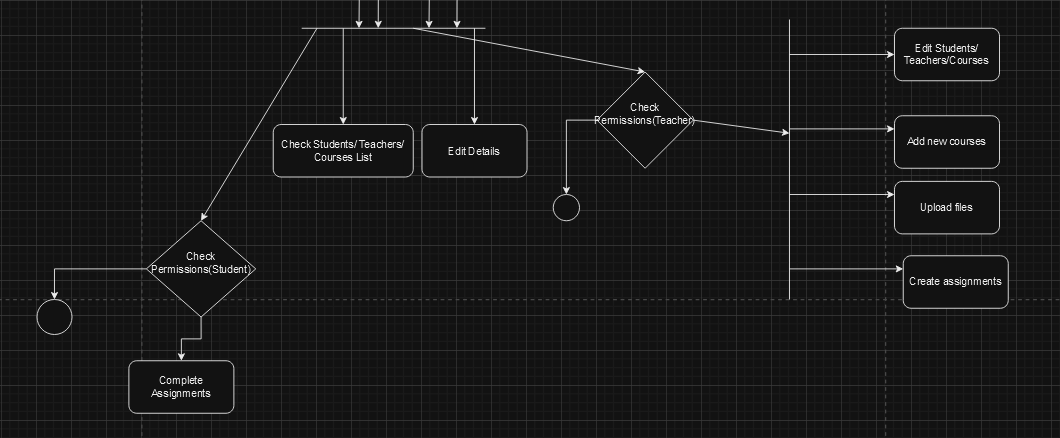
1. **Profesor:**
   * Crează un cont, se autentifică și începe să creeze cursuri.
   * Adaugă assignmenturi la cursuri.
   * Monitorizează progresul assignmenturilor și interacționează cu studenții.
2. **Student:**
   * Creează un cont, se autentifică și accesează lista de cursuri disponibile.
   * Vizualizează assignmenturile asociate și încarcă fișiere pentru a finaliza sarcinile.

**Diagrama Use-Case**

****

**Diagrama de activitate**

****

****

**Design Patterns utilizate**

#### ****1. Model-View-Controller (MVC)****

##### **Unde este utilizat:**

Întregul flux al aplicației este construit pe arhitectura **MVC**, care separă logica aplicației în trei componente principale:

* **Model:** reprezintă datele și logica de afaceri.
* **View:** interfața cu utilizatorul, implementată cu Thymeleaf.
* **Controller:** gestionează cererile utilizatorului și mediază interacțiunea între Model și View.

##### **Cum este implementat:**

* **Model:** Clasele JPA/Hibernate (de ex., User, Course, Assignment) sunt utilizate pentru a modela tabelele din baza de date.
* **Controller:** Clasele controller din Spring Boot (de ex., CourseController, AssignmentController) gestionează cererile HTTP (GET, POST) și trimit datele relevante către View.
* **View:** Thymeleaf generează paginile HTML dinamice bazate pe datele primite de la Controller.

#### ****2. Singleton****

##### **Unde este utilizat:**

Singleton este utilizat în contextul **gestionării bean-urilor Spring**. De exemplu:

* **Service-urile și repository-urile** din aplicație sunt gestionate de Spring ca Singleton beans.

##### **Cum este implementat:**

* În mod implicit, Spring Boot creează o singură instanță a fiecărui bean definit cu anotările @Service, @Repository, sau @Component. Acest lucru asigură o utilizare eficientă a resurselor și elimină necesitatea de a crea manual instanțe multiple.

**Testare**

Pentru a asigura funcționalitatea corectă a aplicației Moodle, testarea a fost realizată manual, simulând diverse scenarii și fluxuri utilizator, atât pentru profesori, cât și pentru studenți.

#### ****1. Testarea fluxurilor utilizatorilor****

Au fost create conturi pentru utilizatori diferiți (profesori și studenți) și s-au verificat funcționalitățile aplicației, urmând pașii următori:

1. **Crearea profesorilor și a cursurilor:**
   * S-au creat conturi de profesori și s-au autentificat în aplicație.
   * Fiecare profesor a creat unul sau mai multe cursuri și a completat detalii precum numele și descrierea cursului.
   * S-a verificat dacă cursurile apar în lista generală și dacă detaliile acestora pot fi editate.
2. **Crearea assignmenturilor:**
   * Profesorii au creat assignmenturi asociate cursurilor, specificând termenele limită și descrierile sarcinilor.
   * S-a verificat dacă assignmenturile sunt vizibile pentru studenții înscriși la cursuri.
3. **Crearea studenților și încărcarea fișierelor:**
   * S-au creat conturi de studenți și s-a verificat autentificarea acestora.
   * Studenții au accesat cursurile și assignmenturile disponibile.
   * Au fost încărcate fișiere în cadrul assignmenturilor, iar profesorii au verificat dacă acestea sunt vizibile și accesibile.
4. **Interacțiunea între utilizatori:**
   * S-a testat lista studenților pentru fiecare curs și atribuirea assignmenturilor.
   * Profesorii au monitorizat progresul fiecărui student în funcție de fișierele încărcate.

#### ****2. Scenarii testate****

1. **Scenarii pozitive:**
   * Crearea și editarea cu succes a cursurilor de către profesori.
   * Vizualizarea assignmenturilor de către studenți și încărcarea fișierelor fără erori.
   * Autentificarea corectă pentru utilizatori cu date valide.
2. **Scenarii negative:**
   * Testarea autentificării cu date incorecte (email sau parolă greșită).
   * Încercarea de a edita assignmenturile sau cursurile de către un student (acces interzis).
   * Încărcarea fișierelor neacceptate sau a fișierelor mari peste limita permisă.

**Mod de rulare a aplicației**

### **1. Cerințe de sistem**

* **Java Development Kit (JDK):** Versiunea 17 sau mai recentă.
* **Maven:** Pentru gestionarea dependențelor și construirea proiectului.
* **MySQL:** Versiunea 8.0 sau mai recentă pentru baza de date.
* **Browser modern:** Pentru accesarea interfeței utilizatorului (Chrome, Firefox, etc.).

### **2. Configurarea mediului de lucru**

#### ****2.1. Instalarea JDK****

* Descărcați și instalați JDK 17 de pe [site-ul oficial Oracle](https://www.oracle.com/java/technologies/javase-downloads.html) sau utilizând OpenJDK.
* Verificați instalarea rulând comanda (din terminal): **java –version**

#### ****2.2. Instalarea Maven****

* Descărcați și configurați Maven de pe [site-ul oficial Apache Maven](https://maven.apache.org/download.cgi).
* Verificați instalarea rulând comanda: **mvn –version**

#### ****2.3. Configurarea MySQL****

* Instalați MySQL și creați o bază de date cu următoarele detalii (pot fi schimbate):
  + Nume bază de date: moodle\_db
  + Utilizator: root
  + Parolă: password

### **3. Descărcarea și configurarea aplicației**

#### ****3.1. Descărcarea codului sursă****

* Clonați proiectul din repository-ul Git utilizând comenzile:

**git clone <URL-ul-repository-ului>** și **cd <numele-folderului-proiectului>**

#### ****3.2. Configurarea fișierului**** application.properties

* Navigați la fișierul de configurare situat în: **src/main/resources/application.properties**
* Actualizați configurațiile bazei de date (cu numele și parola date de dvs.):

### 

### **4. Construirea și rularea aplicației**

#### ****4.1. Construirea aplicației****

* În terminal, navigați la directorul principal al proiectului și rulați comanda: **mvn clean install**

#### ****4.2. Pornirea aplicației****

* După construirea cu succes, rulați aplicația folosind comanda: **mvn spring-boot:run**

#### ****4.3. Accesarea aplicației****

* Deschideți un browser și accesați adresa: http://localhost:8080