**UNIVERSITATEA POLITEHNICA TIMIȘOARA**

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

SPECIALIZAREA INGINERIA SISTEMELOR

**LUCRARE DE LICENȚĂ**

**APLICAȚIE WEB PENTRU GESTIONAREA PLANURILOR DE ÎNVĂȚĂMÂNT**

**Candidat: Daniel-Ionuț MÎRZA**

**Coordonator științific: Conf. dr. ing. Ciprian-Bogdan CHIRILĂ**

Sesiunea: Februarie 2025

**CUPRINS**

1. INTRODUCERE
2. TEHNOLOGII WEB
   1. Limbajul de programare Java
   2. Framework Spring Boot
   3. Biblioteca Apache POI
   4. Sistem de gestionare de date MySQL
   5. Limbajul de programare Thymeleaf
3. CERINȚA ȘI SPECIFICAȚIILE PROIECTULUI
   1. Cerințe funcționale
   2. Cerințe nefuncționale
   3. Cerințe tehnice
4. PROIECTAREA
5. IMPLEMENTAREA
6. TESTARE ȘI INSTALARE
7. CONCLUZII
8. BIBLIOGRAFIE

# INTRODUCERE

Într-un mediu academic în continuă schimbare, gestionarea eficientă a informațiilor referitoare la planurile de învățământ reprezintă o provocare semnificativă. Lucrarea de licență intitulată *"Aplicație web pentru gestionarea planurilor de învățământ"* își propune să răspundă acestei provocări prin dezvoltarea unei soluții software care să automatizeze și să simplifice procesul de administrare a datelor academice.

Aplicația este construită folosind tehnologii moderne, cum ar fi Java, Apache POI, MySQL, Spring Boot și Thymeleaf/Angular, oferind o platformă scalabilă și intuitivă pentru utilizatori. Dezvoltarea aplicației se bazează pe procesarea fișierelor Excel și Word existente, care conțin planurile de învățământ și fișele de disciplină, asigurând astfel o integrare eficientă a datelor.

Proiectul include trei module principale:

1. **Modul administrator** – permite autentificarea, importul planurilor de învățământ, gestionarea fișelor de disciplină și alocarea cadrelor didactice la discipline.
2. **Modul cadru didactic** – oferă posibilitatea autentificării, vizualizării și încărcării de fișe de disciplină.
3. **Modul student** – facilitează accesul la fișele de disciplină filtrate după generație, ciclu de studii și an de studiu.

Lucrarea evidențiază procesul de dezvoltare al aplicației, pornind de la modelarea bazei de date relaționale și integrarea fișierelor de tip Excel și Word, până la implementarea funcționalităților de administrare și utilizare practică.

Prin această aplicație, se dorește crearea unei platforme care să îmbunătățească experiența utilizatorilor și să reducă eforturile administrative, contribuind astfel la o gestionare mai eficientă a resurselor academice.

In capitolul 2 prezentam tehnologiile web folosite in proiect. In capitolul 3 descriem specificatiile proiectului...

1. **TEHNOLOGII WEB**
   1. **Limbajul de programare Java**

Java este un limbaj de programare orientată pe obiecte, bazată pe clase care este proiectată să aibă cât mai puține *dependențe* posibile. Este renumită pentru portabilitatea sa, oferind programatorilor de a scrie cod care poate rula pe orice platformă ce suportă Java fără a necesita recompilare, conform sloganului „Write Once, Run Anywhere” (WORA) [XXX].

Java a fost inițial dezvoltată de către *James Gosling,* Mike Sheridan și Patrik Naughton ca o unealtă de programare pentru televiziune interactivă în iunie 1991. Inițial numită Oak și mai târziu redenumita în Java după cafeaua Java de origine indoneziană, limbajul a fost lansată la Sun Microsystems în 23 mai 1995 ca o componentă principală a platformei Java.Astăzi este întreținută de Oracle (27 ianuarie 2010 prin achiziția Sun Microsystems) [XXX].

Java este recunoscută pentru securitatea sa robustă, care este asigurată printr-o combinație de caracteristici integrate și mecanisme avansate. Printre cele mai importante măsuri se numără sandboxing-ul, care izolează codul și previne accesul neautorizat la resurse, precum și verificarea bytecode-ului de către JVM, care asigură integritatea și conformitatea codului. Un alt element central este Security Manager-ul, care controlează accesul aplicațiilor la fișiere, rețele și alte resurse critice.

**Caracteristici principale ale Java**

1. **Orientarea pe obiecte**: Java este un limbaj pur orientat pe obiecte, ceea ce înseamnă că aproape totul este definit ca un obiect (cu excepția tipurilor de date primitive). Acest lucru face codul mai modular și mai ușor de întreținut.
2. **Independența platformei**: Programele Java sunt compilate în *bytecode*, un cod intermediar care rulează pe orice platformă ce suportă o *Java Virtual Machine* (JVM).
3. **Securitatea**: Java include mecanisme integrate de securitate, ceea ce îl face potrivit pentru aplicații sensibile.
4. **Multi-threading**: Java suportă multitasking și concurența, permițând rularea mai multor fire de execuție (thread-uri) în același timp.
5. **Bibliotecă bogată**: Java vine cu o bibliotecă standard extinsă (API Java), care include multe funcționalități predefinite, de la manipularea colecțiilor de date până la lucrul cu rețele.

Aceasta este o introducere de bază. Java are mult mai multe concepte avansate, cum ar fi interfețele, clasele abstracte, excepțiile, și gestionarea memoriei, care sunt esențiale pentru a deveni un programator Java experimentat.

Clase obiecte metode mostenire pachete incapsulare

* 1. **Biblioteca Spring Boot**

Spring Boot este un framework Java care facilitează crearea și execuția aplicațiilor Java. Simplifică procesul de configurare și setare, permițându-le developărilor să se focuseze mai mult pe scrierea de cod a aplicațiilor.

Spring Boot este dezvoltată de către Pivotal Team și este o combinație dintre Spring Framework și Servere Încorporate (Embedded Servers). În octombrie 2012 un client pe nume Mike Youngstrom a făcut o cerere Jira cerând „bootstraping the spring framework” pentru a putea porni mai repede. Astfel, la începutul lui 2013 Spring Boot a fost creat.

***Avantajele principale ale Spring Boot***

1. ***Configurare automată (Auto-configuration)****: Spring Boot detectează automat librăriile din aplicația ta și configurează în mod implicit componentele necesare, permițându-ți să începi rapid fără o configurare extensivă.*
2. ***Aplicații autonome****: Permite rularea aplicației ca un fișier JAR (Java ARchive) executabil, fără a necesita un server de aplicații extern (cum ar fi Tomcat sau Jetty), deoarece serverul este inclus în aplicație.*
3. ***Starter POMs****: Include o colecție de module (starters) care te ajută să gestionezi dependințele prin specificarea unui singur artefact în pom.xml (în cazul Maven) sau build.gradle (pentru Gradle), reducând necesitatea configurărilor manuale.*
4. ***Spring Boot Actuator****: Oferă puncte de monitorizare pentru a obține informații despre starea aplicației și performanță, utile pentru monitorizare și diagnosticare.*

***Componentele de bază ale Spring Boot***

1. ***Starter Templates****: Pachetele Starter sunt folosite pentru a configura rapid proiecte Spring Boot cu un set de dependințe specifice. De exemplu, spring-boot-starter-web include dependințele pentru crearea aplicațiilor web RESTful.*
2. ***Configurare în application.properties sau application.yml****: În loc să scrii fișiere de configurare XML lungi, Spring Boot permite configurarea simplă a proprietăților aplicației direct în fișierul application.properties sau application.yml.*
3. ***Spring Boot Initializr****: O interfață web (*[*https://start.spring.io/*](https://start.spring.io/)*) care ajută la generarea de proiecte Spring Boot preconfigurate. Poți selecta versiunea Java, tipul de aplicație, dependințele de bază și altele.*

Spring Boot face dezvoltarea aplicațiilor Java mai eficientă, reducând configurările complexe și oferind suport extins pentru monitorizare și scalabilitate.

* 1. **Biblioteca Apache POI**

Apache POI este o bibliotecă Java utilizată pentru a citi și scrie fișiere Microsoft Office, inclusiv formate precum Excel (XLS, XLSX), Word (DOC, DOCX) și PowerPoint (PPT, PPTX). Este foarte utilă în aplicații care necesită manipularea documentelor Office în mod programatic, fie că este vorba de generarea rapoartelor în format Excel, modificarea documentelor Word sau crearea de prezentări PowerPoint.

Numele a fost original un acronim pentru „*Poor Obfuscation Implementation*” (Implementarea slabă a confuziei), refecându-se într-un mod comic la faptul că formatul fișierelor erau deliberat confuze. Această explicație a fost mai târziu ștearsă de pe pagina web oficială pentru o mai bună comercializare către întreprinderi.

**Caracteristici principale ale Apache POI**

1. **Manipularea fișierelor Excel**: POI oferă două API-uri principale pentru fișiere Excel:
   * **HSSF** (Horrible Spreadsheet Format) pentru fișiere Excel 97-2003 (.xls)
   * **XSSF** (XML Spreadsheet Format) pentru fișiere Excel 2007+ (.xlsx)
2. **Suport pentru documente Word**:
   * **HWPF** (Horrible Word Processor Format) pentru fișiere DOC (Word 97-2003)
   * **XWPF** (XML Word Processor Format) pentru fișiere DOCX (Word 2007+)
3. **Prezentări PowerPoint**:
   * **HSLF** (Horrible Slide Layout Format) pentru fișiere PPT (PowerPoint 97-2003)
   * **XSLF** (XML Slide Layout Format) pentru fișiere PPTX (PowerPoint 2007+)
4. **Manipularea fișierelor și datelor personalizate**: POI permite crearea și manipularea diverselor obiecte din fișiere Office, cum ar fi foi de calcul, celule, paragrafe, tabele, imagini și stiluri.

**Avantajele folosirii Apache POI**

* **Gratuit**: Apache POI este un proiect open-source dezvoltat de Apache Software Foundation.
* **Compatibilitate cu multiple formate Office**: Permite manipularea mai multor tipuri de fișiere Office.
* **Documentație bogată**: API-ul este bine documentat și susținut de o comunitate activă.

Apache POI este o soluție excelentă pentru aplicațiile care au nevoie să integreze date Office în mod programatic.

* 1. **Sistem de gestionare de date MySQL**

MySQL este un sistem de gestionare a bazelor de date relaționale (RDBMS) foarte popular și puternic, utilizat pentru a stoca și gestiona date. Este open-source și este folosit în mod frecvent în aplicații web, fiind compatibil cu diferite sisteme de operare, precum Linux, Windows și macOS.

O bază de date este o colecție organizată de informații sau date care sunt stocate și gestionate electronic, de obicei într-un sistem de gestionare a bazelor de date (DBMS – Database Management System). Baza de date permite stocarea, manipularea, accesarea și actualizarea eficientă a datelor.

MySQL este numit după fiica co-fondatorului Monty Widenius, *My*, și SQL, acronimul pentru *Structured Query Language* (Limbajul de interogare structurat). *(Delfinul din logo se numește Sakila). (Aparține de Oracle)(Lansat in 23 mai 1995)*

**Caracteristici de bază ale MySQL**

1. **Sistem de management al bazelor de date relaționale (RDBMS)**: MySQL organizează datele în tabele cu rânduri și coloane, permițând utilizarea limbajului SQL (Structured Query Language) pentru interogare și gestionare.
2. **Performanță ridicată**: MySQL este cunoscut pentru viteza sa în procesarea și gestionarea volumelor mari de date, fiind utilizat frecvent în aplicații cu trafic mare, cum ar fi cele web.
3. **Securitate avansată**: MySQL include controale de acces și criptare, protejând datele împotriva accesului neautorizat.
4. **Scalabilitate**: Este capabil să gestioneze baze de date mari, suportând milioane de înregistrări.
5. **Compatibilitate cu diverse limbaje de programare**: MySQL poate fi integrat cu diverse limbaje de programare, cum ar fi PHP, Java, Python, și altele.

**Avantajele MySQL**

* **Open-source**: Disponibil gratuit, MySQL este susținut de o comunitate mare care contribuie constant la dezvoltarea și îmbunătățirea lui.
* **Compatibilitate ridicată**: Funcționează pe mai multe platforme și este foarte bine integrat cu alte tehnologii de server și limbaje de programare.
* **Ușurință în utilizare**: Interfața și instrumentele MySQL sunt prietenoase pentru dezvoltatori, oferind o experiență accesibilă și pentru începători.

MySQL este un instrument esențial pentru dezvoltatori și administratori de baze de date, permițând gestionarea eficientă a datelor într-o varietate de aplicații și industrii.

* 1. **Biblioteca Thymeleaf**

Thymeleaf este un motor de template Java folosit pentru a genera cod HTML, XML, JavaScript, CSS și text. Este des utilizat în aplicațiile de tip Spring Boot pentru a crea interfețe de utilizator dinamice și intuitive. Thymeleaf permite generarea de pagini web pe partea de server și poate fi integrat cu ușurință în arhitectura MVC (Model-View-Controller), fiind destinat în special pentru proiectele de tip server-side rendering.

Thymeleaf are scopul principal de a oferi un mod elegant și foarte ușor de întreținut de a crea șabloane. Pentru a realiza acest lucru se bazează pe conceptul de Șabloane Naturale pentru a-și insera logica în fișierele șablon într-un mod în care să nu afecteze șablonul de la a fi folosit ca un prototip de design. Acest lucru îmbunătățește comunicarea designului și reduce decalajul dintre echipele de proiectare și dezvoltare.

**Caracteristici cheie ale Thymeleaf**

1. **Sintaxă naturală**: Codul Thymeleaf este foarte apropiat de HTML, ceea ce face șabloanele mai lizibile și ușor de întreținut.
2. **Integrare cu Spring**: Thymeleaf este optimizat pentru a funcționa perfect cu Spring Framework, oferind suport nativ pentru Spring MVC.
3. **Mode de operare dual**: Poate fi folosit atât în mod „standalone” pentru a genera pagini fără un server web, cât și în aplicații web.
4. **Suport pentru template-uri fragmentate**: Thymeleaf permite reutilizarea fragmentelor de template, ceea ce simplifică organizarea și modularitatea codului HTML.
5. **Valori expresive**: Thymeleaf oferă suport pentru expresii condiționale, bucle și alte elemente logice direct în pagini HTML, fără să fie nevoie de alt limbaj adițional.

**Avantaje Thymeleaf**

* Ușor de utilizat și de integrat cu Spring Boot.
* Răspunsuri rapide pe partea de server, fiind ideal pentru aplicații web server-side.
* Reutilizarea fragmentelor HTML și posibilitatea de a scrie cod ușor de citit și întreținut.

Thymeleaf este o alegere excelentă pentru crearea interfețelor de utilizator server-side în proiectele Spring Boot.

1. **CERINȚA ȘI SPECIFICAȚIILE PROIECTULUI**

Proiectarea aplicației web pentru gestionarea planurilor de învățământ reflectă un set bine definit de cerințe funcționale, nefuncționale și tehnice, fiecare având un rol esențial în realizarea unei soluții robuste, eficiente și scalabile.

* 1. **Cerințe funcționale: (paragrafe!)**
  2. Import și procesare a fișierelor Excel și Word
* Posibilitatea de a importa fișiere Excel care conțin planuri de învățământ.
* Generarea automată a fișelor de disciplină pe baza șabloanelor Word.
  1. Autentificare și autorizare
* Sistem de autentificare pentru utilizatorii aplicației (administratori, cadre didactice, studenți).
* Gestionarea rolurilor și permisiunilor specifice fiecărui tip de utilizator.
  1. Managememntul planurilor de învățământ
* Crearea unui model relațional pentru stocarea informațiilor extrase din fișierele Excel.
* Posibilitatea de a vizualiza, edita și actualiza planurile existente.
* Copierea fișelor de disciplină în generații noi la schimbarea anului universitar.
  1. Alocarea resurselor
* Funcționalitate pentru alocarea cadrelor didactice la discipline.
  1. Interfață pentru cadre didactice
* Vizualizarea fișelor de disciplină alocate unui cadru didactic.
* Încărcarea de fișe de disciplină noi.
  1. Interfață pentru studenți
* Vizualizarea fișelor de disciplină pe baza generației, ciclului (licență/master) și anului de studiu.

Cerințele funcționale se concentrează pe oferirea unui set clar de funcționalități destinate diferitelor categorii de utilizatori (administratori, cadre didactice, studenți), având ca scop principal automatizarea proceselor administrative. Acestea includ importul fișierelor Excel, gestionarea fișelor de disciplină și asigurarea unei interfețe intuitive pentru utilizatori.

* 1. **Cerințe nefuncționale:**
  2. Performanță
* Aplicația trebuie să răspundă rapid la interacțiuni, chiar și atunci când procesează fișiere mari.
  1. Scalabilitate
* Structura aplicației trebuie să permită adăugarea de noi funcționalități fără a afecta performanța sau stabilitatea.
  1. Securitate
* Datele utilizatorilor și fișierele importate trebuie protejate împotriva accesului neautorizat.
* Autentificarea utilizatorilor trebuie realizată folosind standarde moderne, cum ar fi hashing pentru parole.
  1. Compatibilitate
* Suport pentru fișiere Excel și Word de formate diferite (ex. .xls, .xlsx, .doc, .docx).
* Funcționalități optimizate pentru browsere moderne și compatibilitate cross-platform.
  1. Usabilitate
* Interfața trebuie să fie intuitivă și ușor de utilizat pentru toate categoriile de utilizatori.
  1. Fiabilitate
* Aplicația trebuie să gestioneze corect erorile de utilizare (ex. importul de fișiere invalide) și să ofere mesaje de eroare clare.

Cerințele nefuncționale asigură calitatea și fiabilitatea aplicației prin standarde de performanță, securitate, scalabilitate și compatibilitate. Astfel, aplicația trebuie să funcționeze eficient chiar și în scenarii complexe, să protejeze datele utilizatorilor și să fie accesibilă pe multiple platforme și browsere moderne.

* 1. **Cerințe tehnice:**
  2. Tehnologii utilizate
* Backend: Java, Spring Boot.
* Frontend: Thymeleaf.
* Baza de date: MySQL.
* Manipularea documentelor: Apache POI pentru Excel și Word.
  1. Arhitectură
* Model MVC (Model-View-Controller) pentru separarea logicii aplicației de interfața utilizator.
* Integrarea REST API pentru interacțiuni între frontend și backend (mai ales dacă se folosește Angular).
  1. Testare
* Teste unitare pentru verificarea funcționalității fiecărei componente.
* Teste de integrare pentru validarea interacțiunii între componente.

Cerințele tehnice oferă cadrul tehnologic și arhitectural pentru implementare. Alegerea unui stack tehnologic solid (Java, Spring Boot, MySQL, Apache POI, Thymeleaf/Angular) garantează o infrastructură flexibilă și ușor de întreținut, în timp ce arhitectura MVC și utilizarea standardelor moderne (REST API, Spring Security) asigură o dezvoltare modulară și bine structurată.

Prin îmbinarea acestor cerințe, aplicația urmărește nu doar satisfacerea nevoilor curente ale utilizatorilor, ci și crearea unei soluții sustenabile și extensibile, capabile să se adapteze viitoarelor cerințe și schimbări din mediul academic. Acest echilibru între funcționalitate, performanță și tehnologie reprezintă fundamentul pentru succesul proiectului.

* 1. **PROIECTAREA**