UNIVERSITATEA “LUCIAN BLAGA” DIN SIBIU
FACULTATEA DE INGINERIE
DEPARTAMENTUL DE CALCULATOARE ŞI INGINERIE ELECTRICĂ

# PROIECT DE DIPLOMĂ

Conducător științific : Crețulescu Radu George

Îndrumător: Crețulescu Radu George

Absolvent: Popa Doroteea Cristina

Specializarea: Tehnologia Informației

- Sibiu, 2024 –

UNIVERSITATEA “LUCIAN BLAGA” DIN SIBIU
FACULTATEA DE INGINERIE
DEPARTAMENTUL DE CALCULATOARE ŞI INGINERIE ELECTRICĂ

# APLICAȚIE GESTIONARE DE PROIECTE DEP.CIE

Conducător științific: Crețulescu Radu George

Îndrumător: Crețulescu Radu George

Absolvent: Popa Doroteea Cristina

Specializarea: Tehnologia Informației

# CUPRINS

# 1. Prezentarea temei

## 1.1. Introducere

În era digitală modernă, gestionarea eficientă a proiectelor academice este esențială pentru succesul educațional. Instituțiile de învățământ superior se confruntă adesea cu provocări legate de organizarea și monitorizarea progresului studenților în cadrul proiectelor lor. Pentru a aborda aceste provocări, am dezvoltat aplicația "UniProject Manager", o platformă web destinată să sprijine studenții, profesorii și personalul administrativ în gestionarea proiectelor academice și a lucrărilor de licență. Această aplicație oferă un mediu centralizat și intuitiv, care facilitează comunicarea și colaborarea între toți participanții la procesul educațional.

## 1.2. Descrierea temei

Tema proiectului "UniProject Manager" se concentrează pe dezvoltarea unei aplicații web complexe, care să răspundă nevoilor specifice ale facultății de inginerie ULBS Sibiu. Aplicația este destinată să ajute cinci categorii principale de utilizatori: studenți, profesori, coordonatori, secretara și administratorul site-ului. Fiecare tip de utilizator beneficiază de funcționalități specifice care îmbunătățesc organizarea și gestionarea proiectelor.

**Studenții** pot accesa și gestiona informațiile legate de proiectele lor academice într-un mod organizat și eficient. Ei pot vizualiza datele personale, proiectele curente și notele aferente. În plus, aplicația permite studenților să actualizeze progresul proiectelor lor prin bifarea cerințelor îndeplinite și încărcarea de fișiere relevante. Pentru studenții din ani terminali, aplicația oferă opțiunea de a selecta și gestiona teme de licență, facilitând comunicarea cu coordonatorii de proiect.

**Profesorii** au posibilitatea de a gestiona proiectele studenților, de a adăuga cerințe pentru proiecte și de a evalua munca studenților prin vizualizarea și notarea arhivelor încărcate. Aplicația permite coordonatorilor să gestioneze cererile de licență, să accepte sau să respingă cererile studenților și să vizualizeze progresul proiectelor de licență.

**Secretara** joacă un rol crucial în administrarea utilizatorilor și a datelor aferente. Ea poate adăuga sau șterge profesori și studenți, poate introduce coordonatori și poate seta datele

de prezentare pentru temele de licență. De asemenea, secretara primește listele cu candidații pentru fiecare specializare și asigură o organizare eficientă a acestora.

**Administratorul** are drepturi limitate, dar esențiale pentru funcționarea aplicației. Acesta poate schimba secretara și poate actualiza anul academic, asigurând astfel o administrare eficientă a platformei.

## 1.3. Scop si obiective

Scopul principal al aplicației "UniProject Manager" este de a crea o platformă integrată și eficientă pentru gestionarea proiectelor academice și a lucrărilor de licență. Prin această aplicație, dorim să îmbunătățim organizarea și colaborarea între studenți, profesori și coordonatori, oferindu-le un instrument puternic și intuitiv.

Obiective specifice includ:

- **Facilitarea organizării proiectelor**: Aplicația oferă studenților un mod simplu și eficient de a-și gestiona proiectele academice, vizualizând cerințele și progresul în timp real.
- **Îmbunătățirea comunicării**: Prin intermediul platformei, studenții pot comunica mai ușor cu profesorii și coordonatorii lor, având acces la informații relevante și actualizate.
- **Simplificarea procesului de încărcare și evaluare**: Aplicația permite studenților să încarce fișiere relevante pentru proiectele lor, iar profesorii pot evalua aceste fișiere într-un mod structurat și eficient.
- **Centralizarea gestionării licențelor**: Studenții din ani terminali pot selecta și gestiona temele de licență propuse de coordonatori, facilitând astfel procesul de alegere și monitorizare a progresului lucrărilor de licență.

## 1.4. Tehnologii utilizate

Dezvoltarea aplicației "UniProject Manager" a implicat utilizarea unui set diversificat de tehnologii, care au fost alese pentru a asigura performanța, securitatea și scalabilitatea platformei.

- **Visual Studio Code**: Editorul de cod utilizat pentru dezvoltarea aplicației, oferind o interfață prietenoasă și multiple extensii care facilitează scrierea și depanarea codului.

- **PHP**: Limbajul de programare server-side folosit pentru logica aplicației și manipularea datelor pe server.
- **HTML și CSS**: Tehnologiile utilizate pentru structura și stilizarea interfeței utilizatorului, asigurând o experiență de utilizare ideală.
- **JavaScript**: Limbajul de programare utilizat pentru adăugarea de funcționalități dinamice și interactive în cadrul aplicației.
- **XAMPP**: Pachetul software care include Apache (server web), MySQL (bază de date) și PHP, utilizat pentru dezvoltarea și testarea locală a aplicației.
- **MySQL**: Sistemul de gestionare a bazelor de date utilizat pentru stocarea și gestionarea datelor aplicației.
- **phpMyAdmin**: Instrumentul folosit pentru administrarea bazelor de date MySQL, oferind o interfață grafică ușor de utilizat pentru gestionarea datelor.

## 1.5. Cazuri principale de utilizare a aplicației

Aplicația "UniProject Manager" este concepută pentru a acoperi o gamă largă de scenarii de utilizare, fiecare adresând nevoile specifice ale diferitelor categorii de utilizatori.

**Studenți**:

- **Organizarea proiectelor**: Studenții pot accesa secțiunea "Acasă" pentru a vizualiza datele personale și proiectele asociate. În secțiunea "Proiecte", aceștia pot vedea toate proiectele lor afișate sub formă de carduri colorate în funcție de an (albastru pentru anul 1, roz pentru anul 2, verde pentru anul 3, galben pentru anul 4 și portocaliu pentru licență). Fiecare card conține detalii despre proiect, iar la click, utilizatorul este redirecționat către detaliile proiectului, unde poate bifa cerințele îndeplinite, încărca fișiere și actualiza progresul.
- **Gestionarea licențelor**: Pentru studenții din anul 4, există un card special pentru licență. Acesta afișează datele de predare și prezentare a lucrării de licență. La click, sunt afișate temele de licență propuse de coordonatori, iar studenții pot trimite cereri coordonatorului pentru tema dorită. Statutul cererii (în așteptare, acceptată) este actualizat în timp real, iar studenții pot încărca arhivele necesare după ce cererea este acceptată de coordonator.

**Profesori**:

- **Gestionarea proiectelor**: Profesorii pot accesa secțiunea "Acasă" pentru a vedea un rezumat al activităților lor. În secțiunea "Proiecte", profesorii pot vedea carduri pentru fiecare semigrupă la care predă, cu detalii despre proiectele studenților. La click pe un card, profesorii pot adăuga cerințe, vizualiza arhivele încărcate și pot nota proiectele studenților.
- **Gestionarea licențelor**: În secțiunea dedicată licențelor, profesorii pot propune teme de licență, seta numărul de locuri disponibile și data predării. De asemenea, ei pot vizualiza cererile studenților și pot accepta sau respinge aceste cereri. Statutul cererilor și arhivele încărcate de studenți sunt gestionate eficient prin intermediul platformei.

**Secretara**:

- **Administrarea utilizatorilor**: Secretara poate adăuga și șterge profesori și studenți, poate introduce coordonatori. Aceste funcționalități sunt accesibile prin secțiunea "Acasă".
- **Gestionarea licențelor**: Secretara primește listele cu candidații pentru fiecare specializare și asigură organizarea eficientă a acestora, setând data de prezentare a licențelor de către fiecare semigrupă.

**Administrator**:

- **Gestionarea aplicației**: Administratorul are drepturi limitate, dar esențiale pentru funcționarea aplicației. Acesta poate schimba secretara și poate actualiza anul academic, asigurând astfel o administrare eficientă a platformei.

Aplicația "UniProject Manager" reprezintă un instrument puternic și versatil pentru gestionarea proiectelor academice și a lucrărilor de licență. Prin integrarea funcționalităților specifice pentru studenți, profesori, coordonatori și administratori, platforma îmbunătățește considerabil organizarea și comunicarea în cadrul instituției de învățământ. Utilizarea tehnologiilor moderne asigură performanța și scalabilitatea necesare pentru a răspunde nevoilor complexe ale utilizatorilor, facilitând un proces educațional eficient.

# 2. Considerații teoretice

## 2.1. PHP și PhpMyAdmin

### 2.1.1. Introducere

PHP (Hypertext Preprocessor) este un limbaj de scripting server-side, aceasta însemnând ca rulează la nivel de server, nu la nivel de client (browser). Acest limbaj e folosit în principal în dezvoltarea web. Acest limbaj de programare a fost creat de Rasmus Lerdorf în anul 1994 si este unul dintre cele mai utilizate limbaje in dezvoltarea web. PHP este preferat de către programatori deoarece oferă flexibilitate, este relativ simplu de utilizat și oferă posibilitatea de a lucra cu diferite baze de date, astfel asigurând dezvoltarea unor pagini web care oferă siguranță cibernetică si dinamicitate.

Unul dintre motivele pentru care PHP este un limbaj popular este faptul ca este open-source, acest lucru facilitând accesarea și modificarea codului sursă. Acest lucru a determinat formarea unei comunități de utilizatori care au îmbunătățit constant capacitățile PHP. De asemenea, PHP poate rula pe mai multe sisteme de operare, precum Linux, macOS si Windows si e compatibil cu servere web importante, cum ar fi Apache și Nginx.

Un alt avantaj al PHP-ului este sintaxa simplă, ușor de învățat, fiind accesibil pentru persoanele deja familiarizate cu limbaje de programare precum C, Java, Perl sau începătorilor care vor să învețe dezvoltare web. PHP oferă un proces simplificat de dezvoltare prin manipularea formularelor, operațiuni cu fișiere și gestionare de sesiuni.

### 2.1.2. Caracteristici cheie ale limbajului PHP

PHP are diverse caracteristici care îl fac să fie un limbaj atractiv pentru programatori [WWW02]. Câteva caracteristici importante a PHP-ului ar fi:

- Simplicitate și ușurință în învățare:

Sintaxa PHP e asemănătoare cu a altor limbaje de programare cunoscute (C, Perl), fiind astfel ușor de învățat pentru cei cu cunoștințe anterioare despre aceste limbaje. Sintaxa e simpla, ușor de învățat până și pentru persoanele care abia încep să învețe programare.

- Performanță și viteză:

PHP este un limbaj interpretat la nivel de server, aceasta însemnând că serverul execută codul PHP înainte de trimiterea paginii către client. Astfel, conținutul e generat dinamic, în timp real, fără un efect negativ semnificativ asupra performanței sau vitezei.

- Suport extins pentru baze de date:

PHP poate fi folosit ușor cu majoritatea sistemelor de gestionare a bazelor de date. Printre ele se regăsesc MySQL, PostgreSQL, Oracle si multe altele. Aceasta oferă dezvoltatorilor flexibilitate în alegerea bazei de date care e cea mai potrivită cu nevoile lor.

- Gestionarea sesiunilor:

Este o caracteristică esențială în dezvoltarea aplicațiilor web care folosesc autentificarea și păstrarea stării utilizatorilor. PHP oferă un mecanism simplu pentru a gestiona sesiunile, permițând urmărirea și menținerea informațiilor despre utilizatori pe parcursul mai multor pagini.

- Manipularea erorilor:

PHP conține funcții integrate pentru gestionare de excepții și erori. Dezvoltatorii pot folosi blocuri try-catch pentru gestionarea eficientă a excepțiilor și pot seta niveluri diferite pentru raportarea erorilor.

Alte caracteristici importante a PHP-ului ar fi portabilitatea și compatibilitatea cu diverse sisteme de operare (Linux, Windows, macOS), actualizări și îmbunătățiri continue de performanță de către o comunitate mare de dezvoltatori, precum și corecții de securitate, asigurând ca PHP rămâne un limbaj performant și relevant.

Toate aceste caracteristici determină popularitatea și accesibilitatea PHP-ului, fiind considerat și în prezent un limbaj adecvat pentru dezvoltarea aplicațiilor web interactive și dinamice.

### 2.1.3. PHP în dezvoltarea web

PHP este un limbaj important în dezvoltarea de aplicații web datorită scalabilității, flexibilității și posibilității oferite de a crea aplicații web dinamice și interactive. Cu ajutorul PHP se pot construi aplicații web sigure, rapide și ușor de întreținut datorită evoluții constante a limbajului. PHP facilitează diferite aspecte ale dezvoltării web, cum ar fi: generarea de conținut dinamic, integrarea cu baze de date, utilizarea framework-urilor PHP, resurse PHP, securitate și suport pentru servicii web.

Spre deosebire de HTML care este static, paginile PHP au posibilitatea de a genera conținut personalizat în mod automat prin afișarea informațiilor dintr-o bază de date,

personalizarea mesajelor pentru utilizatori sau prin actualizarea conținutului paginilor in timp real. Toate acestea contribuie la utilitatea și interactivitatea site-urilor web create folosind PHP.

### 2.1.4. Securitatea

Un aspect important în dezvoltarea aplicațiilor web îl reprezintă securitatea. Pentru asigurarea securității PHP oferă mecanisme de protejare a aplicațiilor și de diminuare a vulnerabilităților. Pentru asigurarea securității aplicațiilor web e necesară o bună cunoștință a acestor mecanisme și aplicarea diferitelor măsuri de securitate.

Prin validarea și curățarea datelor de intrare se pot preveni atacuri de tip injecție sau exploatări bazate pe input rău intenționat. Injecțiile SQL sunt o tehnică comună de atac al aplicațiilor web. Pentru a le putea evita e nevoie ca programatorii să implementeze interogări care să asigure faptul că datele primite de la utilizatori sunt tratate ca parametri, nu ca parte a comenzii SQL. Acest lucru se poate realiza prin intermediul extensiilor PDO (PHP Data Objects) sau MySQLi, suportate de limbajul PHP.

Alte metode importante în asigurarea securității sunt gestionarea sigură a sesiunilor și folosirea de funcții de criptare și decriptare. Gestionarea sigură a sesiunilor e necesară pentru integritatea și protecția datelor utilizatorilor. PHP conține mecanisme integrate pentru a simplifica gestionarea sesiunilor și funcții cu ajutorul cărora se pot crea, distruge și regenera ID-uri de sesiune (exemplu: session_start(), unset()).De asemenea, PHP pune la dispoziția programatorilor funcții pentru criptarea și decriptarea datelor cu caracter sensibil (exemplu: password_hash() și password_verify()). Aceste funcții sunt folosite pentru criptare și verificarea parolelor în siguranță, nepermițând accesul neautorizat.

Prin folosirea acestor metode se asigură dezvoltarea unor aplicații web robuste și sigure, protejate de atacuri cibernetice și care oferă integritatea datelor utilizatorilor.

### 2.1.5. Instrumente și resurse pentru dezvoltare

Dezvoltarea aplicațiilor web cu ajutorul PHP se poate realiza cu ajutorul unui set de instrumente și resurse ce simplifică procesul de scriere și gestionare a codului. Aceste instrumente includ editoare de cod IDE-uri, biblioteci și resurse online.

Printre cele mai populare editoare și IDE-uri se numără PHPStorm și Visual Studio Code. PHPStorm e un IDE dezvoltat de JetBrains, special pentru lucrul cu PHP, suportând framework-uri PHP, precum Laravel și Symfony. PHPStorm pune la dispoziție diferite facilități precum debugging ușor de realizat, integrarea bazelor de date, suport pentru testare și

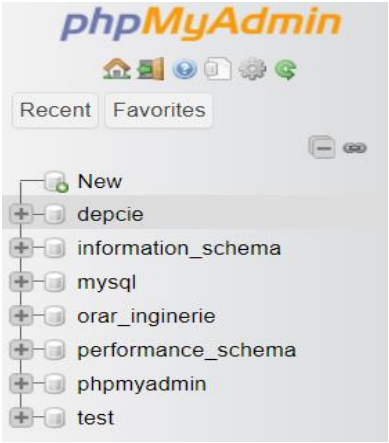
posibilitatea de refactorizare a codului și instrumente integrate de control al versiunilor (ex: GIT). De asemenea, Visual Studio Code e un editor popular datorită flexibilității și numărului de extensii oferite.

Un plus al folosirii PHP îl reprezintă comunitatea extinsă si resursele online disponibile. PFP.net este site-ul oficial al limbajului PHP care oferă o gamă largă de resurse, ghiduri, documentație cu exemple de cod și specificații detaliate care simplifică experiența dezvoltatorilor experimentați sau începători. Un beneficiu este, bineînțeles, faptul că documentația este actualizată constant și secțiunea de comentarii care permit utilizatorilor să își ofere feedback-ul și să contribuie la corectarea sau îmbunătățirea conținutului.

### 2.1.6. PhpMyAdmin

PhpMyAdmin este un instrument software open-source făcut pentru a ușura gestionarea și administrarea bazelor de date MySQL și MariaDB. Această platformă este realizată în principal cu ajutorul PHP. Popularitatea sa se datorează interfeței sale prietenoase și ușor de utilizat care îl face un tool ideal pentru programatorii începători sau avansați.

PhpMyAdmin oferă o interfață grafică intuitivă pentru crearea și gestionarea bazelor de date; în partea stângă a platformei sunt poziționate bazele de date cu butonul de “new” pentru crearea unei noi baze de date (Fig 2.1), iar în partea de sus avem un meniu cu diferitele funcții oferite de PhpMyAdmin Fig. 2.2




(fig. 2.1)


(fig. 2.2)

Utilizatorii pot executa interogări SQL direct din platforma PhpMyAdmin folosind butonul “SQL”. Acest lucru este foarte avantajos pentru crearea, modificarea și ștergerea datelor din baza de date, oferind un mod rapid de a interacționa cu baza de date. O altă caracteristică importantă o reprezintă suportul integrat pentru import și export al datelor în diferite formate, precum, CSV, SQL, JSON și XML. Aceasta asigură o bună realizare de backup-uri și integrarea datelor cu alte aplicații. PhpMyAdmin oferă și o funcționalitate de “designer” care e folosită pentru vizualizarea și gestionarea grafică a tabelelor din baza de date. Aceasta permite vizualizarea tabelelor și a relațiilor dintre ele și a conectării prin cheile străine. Utilizatorii pot genera și diagrame PDF ale structurii bazei de date, acest lucru fiind util în realizarea unei documentații sau pentru partajarea acesteia.

## 2.2. SQL și MySQL

### 2.2.1. Introducere

MySQL este un sistem de gestionare al bazelor de date relaționale, popular printre dezvoltatorii de aplicații web. A fost dezvoltat de MySQLAB dar este în prezent deținut de Oracle Corporation. Este folosit pe scară largă deoarece oferă diferite beneficii, precum scalabilitate, performanță ridicată și flexibilitate, fiind ideal pentru o gestionare eficientă de date. Performanța ridica se datorează vitezei și eficienței în gestionarea volumelor mari de date care se pot extinde pe mai multe servere. Un alt avantaj îl reprezintă suportul pentru SQL pentru facilitarea manipulării și controlului de date.

SQL (Structured Query Language) este un limbaj standardizat de programare folosit în gestionarea bazelor de date. A fost creat de IBM în 1970, iar de atunci popularitatea sa a crescut, devenind standard pentru manipularea bazelor de date relaționale. SQL are o sintaxă relativ ușor de învățat. Sintaxa este declarativă, utilizatorii putând specifica acțiunile dorite asupra bazei de date fără a descrie detaliat modul de execuție al operației.

### 2.2.2. Structură și funcționalități

Structura MySQL e compusă din următoarele componente: serverul MySQL, motoare de stocare, clientele MySQL și biblioteci API. Serverul MySQL e nucleul care se ocupă de operațiile legate de baza de date, conexiunile utilizatorilor și gestionarea interogărilor. MySQL suportă diferite motoare de stocare, printre care InnoDB (implicit) și MyISAM (rapid, folosit mai ales pentru tabele care nu necesită tranzacții). Clientele MySQL sunt aplicații care fac posibilă interacțiunea dintre utilizatori și serverul MySQL (MySQL Command Line Client,

MySQL Workbench). Bibliotecile API sunt oferite de MySQL pentru limbaje de programare PHP, Phyton, Java și altele, acestea ajutând în gestionare bazelor de date MySQL.

MySQL oferă funcționalități necesare în dezvoltarea de aplicații web. MySQL permite crearea, ștergerea, modificarea bazelor de date și inserarea, interogarea, actualizarea și ștergerea datelor din tabelele din baza de date [WWW01].

### 2.2.3. Exemple de utilizare

Pasul inițial în folosirea MySQL este crearea unei baze de date si a tabelelor conținute de aceasta. Aceasta se poate realiza folosind comenzile SQL:

```
CREATE DATABASE magazinOnline;

USE magazinOnline;

CREATE TABLE produse_din_magazin (

id INT AUTO_INCREMENT PRIMARY KEY,

nume VARCHAR(200),

preț INT,

stoc INT

);
```

(Ex. 1.1.)

Folosind aceste comenzi, am creat baza de date magazinOnline care conține tabelul produse_din_magazin. Tabelul conține un id, care e setat ca cheie primară care e incrementat automat și alte trei atribute care descriu produsele din magazin: nume, preț și stoc.

O altă operație des folosită este inserarea de date în tabele. În exemplul următor sunt inserate produsele telefon, cu prețul de 2000 și stocul de 35 și televizor cu prețul de 2500 și stocul de 20:

```
INSERT INTO produse_din_magazin(nume, preț, stoc) VALUES ('telefon', 2000, 35);

INSERT INTO produse_din_magazin(nume, preț, stoc) VALUES ('televizor', 2500, 20);
```

(Ex. 1.2.)

Alte operații importante în gestionarea bazelor de date sunt selectarea(SELECT), actualizarea(UPDATE) și ștergerea(DELETE) de date din tabele. Acestea sunt reprezentate în următorul exemplu unde selectăm toate produsele din tabelul de produse_din_magazin, actualizăm prețul telefonului la 2200 și ștergem televizorul:

```
SELECT * FROM produse_din_magazin;

UPDATE produse_din_magazin SET preț = 2200 WHERE nume = 'telefon';

DELETE FROM produse_din_magazin WHERE nume = 'televizor';
```

(Ex. 1.3.)

## 2.3. HTML

### 2.3.1. Introducere

HTML, prescurtare pentru HyperText Markup Language, este limbajul standard utilizat pentru crearea și structurarea paginilor web. Folosind "tag-uri", HTML permite dezvoltatorilor să organizeze și să prezinte conținutul pe internet într-un mod structurat. Inventat de Tim Berners-Lee în 1991, HTML a început cu doar 18 elemente. De atunci, a evoluat semnificativ, cu actualizări importante precum HTML 2.0 în 1995, HTML 3.2 în 1997 care a adus suport pentru tabele, HTML 4.01 în 1999 care a introdus suport pentru scripturi și HTML5 în 2014 care a adăugat capabilități pentru audio, video și elemente grafice avansate.

### 2.3.2. Structură de bază a unui document

Un document HTML are o structura standard, cu elemente specifice care definesc aspectul paginii web. La această structură se adaugă ulterior elemente noi în funcție de dorințele și cerințele dezvoltatorilor web. Structura unui document este afișată și descrisă mai jos [WWW04].

```
<!DOCTYPE html>

<html lang="ro">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document HTML Exemplu</title>
```

```
</head>
<body>
  <h1>Bun venit la HTML</h1>
  <p>Aceasta este o pagină web de bază.</p>
</body>
</html>
```

Tipul documentului și versiunea HTML este declarată folosind <!DOCTYPE html>. Tagul <html> este elementul rădăcină, care conține toate celelalte elemente HTML. Tagul <head> include metadate despre document, ca de exemplu titlu și link-uri către resurse externe. Tagul <title> e folosit pentru a da titlul browserului, <meta charset="UTF-8"> e utilizat pentru a specifica standardul de codare a caracterelor și în interiorul tagurilor <body> e structurat conținutul efectiv al paginii web (text, imagini, link-uri și alte elemente).

### 2.3.3. Elemente HTML

Elementele HTML se referă la blocurile de bază ale unui document HTML. Fiecare bloc este creat folosind un tag de deschidere (< >) și unul de închidere (</ >) și conținutul este plasat între aceste taguri. Titlurile pot fi specificare, folosind tagurile 'h1' până la 'h6' (exemplu: <h1> Titlu</h1>). Un paragraf se poate defini prin tagul 'p' ( Paragraful respectiv ). Listele ordonate și neordonate pot fi reprezentate cu tagurile 'ul', respective 'ol' și 'li' definește un element dintr-o listă. Listele sunt elemente importante a HTML, putând fi utilizate în construcția meniului unei pagini web. Alte elemente HTML importante sunt link-rile (<a href="https://www.link.com">link.com ) și imaginile (<img src="imagine.jpg" alt="descriere imagine" >).

Atributele au și ele un rol important în HTML: 'href' specifică URL-ul care conduce către link, 'src' specifică sursa imaginii, 'alt' e folosit pentru a oferi un text alternativ afișat dacă nu se poate încărca imaginea dorită, 'class' și 'id' sunt folosite pentru a adăuga stil și identificator unic elementelor HTML.

### 2.3.4. Practici de utilizare HTML

Pentru a crea cod ușor de înțeles și menținut, există diferite practici bune care se utilizează. Structurarea clară a codului e absolut necesară pentru a obține un astfel de cod. Utilizarea de indentare corespunzătoare și structurarea codului HTML în mod logic, cu

subsecțiuni clare sunt tehnici apreciate. O altă practică comună este utilizarea tag-urilor semantice, precum “<header>”, “<nav>”, “<footer>”, “<section>”, “<article>”, acestea îmbunătățind claritatea și accesibilitatea codului HTML. Un use case al acestora e reprezentat in Ex. 1.4.

```
<header>
  <h1>Titlu Site</h1>
  <nav>
    <ul>
      <li><a href="#home">Acasă</a></li>
      <li><a href="#about">Despre</a></li>
      <li><a href="#contact">Contact</a></li>
    </ul>
  </nav>
</header>
<section>
  <article>
    <h2>Articol Principal</h2>
    <p>Conținutul articolului.</p>
  </article>
</section>
<footer>
  <p>© 2024 Compania Mea</p>
</footer>
```

(Ex. 1.4.)

Alte practici aplicate sunt minimizarea codului HTML prin eliminarea spațiilor și comentariilor nefolositoare. Aceasta duce la reducerea fișierului HTML și ca urmare, la optimizarea și creșterea performanței.

Toate acestea fac din HTML un limbaj fundamental în dezvoltarea de aplicații web. Aplicarea acestor practici e esențială în crearea de site-uri web eficiente, moderne, accesibile și interactive.

## 2.4. CSS

### 2.4.1. Introducere

CSS (Cascading Style Sheets) este un limbaj de stil utilizat pentru a descrie aspectul unui document HTML. CSS facilitează gestionarea paginilor web deoarece permite programatorilor să separe conținutul paginilor de aspectul acestora. Prima versiune CSS a fost creată în 1996 de către W3C. Acesta conținea elemente de aspect fundamentale, precum culori, fonturi, margini. În anul 1998 a apărut versiunea CSS2, care a introdus funcționalități avansate, cum ar fi, poziționarea relativă și absolută. Versiunea CSS2.1, apărută în 2011, a corectat unele erori ale versiunii anterioare și a îmbunătățit specificațiile. Versiunea CSS3, care se află în curs de dezvoltare, permite dezvoltarea mai rapidă a funcționalităților, prin aducerea de îmbunătățiri semnificative. Această versiune include selectoare avansate, efecte de tranziții și animații.

### 2.4.2. Sintaxa și selectoarele CSS

CSS are o sintaxă simplă, creată din selectoare și declarații. Declarațiile sunt formate din proprietăți și valori [WWW05]. Această structură este reprezentată în exemplul Ex. 1.1.

Selectoarele CSS sunt folosite pentru a selecta elementele HTML pe care dezvoltatorul dorește să le stilizeze. Există diverse selectoare CSS care sunt utilizate în funcție de ceea ce se dorește a fi selectat. Selectoarele de tip sunt folosite pentru a selecta toate elementele de tipul respectiv (Ex. 1.2.). Selectoarele de clasă selectează toate elementele cu o anumită clasă (Ex. 1.3.). Selectoarele de ID selectează un element unic cu un anumit ID (Ex. 1.4.). Selectoarele de atribut selectează elementele pe baza unui atribut (Ex. 1.5.).

```
selector {proprietate: valoare;}
```

(Ex.1.1)

```
p {color: blue;}
```

(Ex. 1.2.)

```
.important {font-weight: bold;}
```

(Ex. 1.3.)

```
#header {background-color: gray;}
```

(Ex. 1.4.)

```
input[type="text"] { border: 1px solid  black;}
```

(Ex. 1.5.)

### 2.4.3. Proprietăți CSS esențiale

Câteva dintre cele mai importante proprietăți CSS folosite în stilizarea paginilor web sunt setarea de culori și fundaluri (Ex. 1.6.), fonturi și text (Ex. 1.7.), margini și spațieri (Ex. 1.8.), borduri și umbre (Ex. 1.9.), afișare și vizibilitate (Ex. 2.0.).

Pentru setarea de culori și fonturi se folosesc următoarele : ‘body’ cu care se setează culoarea textului unui element, ‘background-color’ setează culoarea de fundal al unui element, ‘background-image’ adaugă o imagine de fundal al unui element. Pentru setarea de fonturi și text sunt utilizate proprietățile: ‘font-family’ care specifică familia de fonturi pentru text, ‘font-size’ setează dimensiunea textului, ‘font-weight’ controlează grosimea fontului, ‘text-align’ aliniază textul în interiorul unui element, ‘text-decoration’ adaugă sau elimină decorarea textului. Pentru setarea de margini și spațieri se folosesc proprietățile: ‘margin’ setează spațiul exterior al unui element, ‘padding’ setează spațiul interior al elementelor. Pentru setarea de bordurilor și a umbrelor se folosesc: ‘border’ setează toate proprietățile bordurii (lățime, stil, culoare etc.), ‘border-radius’ setează colțurile rotunjite ale unui element, ‘box-shadow’ adaugă o umbra unui element. Afișarea și vizibilitatea sunt gestionate cu ‘display’ care controlează modul în care este afișat un element  și  ‘visibility’ care controlează vizibilitatea unui element.

```
body {background-color: #f0f0f0;
  color: #333;}
```

(Ex. 1.6)

```
h1 {font-family: 'Arial', sans-serif;
  font-size: 2px;
  text-align: center;}
```

(Ex. 1.7)

```
p {margin: 10px 0;
  padding: 5px;}
```

(Ex. 1.8)

```
div {border: 1px solid #ccc;}
```

(Ex. 1.9)

```
.block-element {display: block;}
```

(Ex. 2.0)

### 2.4.4. Practici de utilizare CSS

Pentru a asigura obținerea unui cod clar, eficient și ușor de gestionat este esențial să se urmeze anumite practici și tehnici bine definite. De asemenea, utilizarea de framework-uri CSS ajută la îmbunătățirea procesului de dezvoltare a aplicațiilor web interactive și cu un aspect modern.

O practică bună des întâlnită de organizare a codului CSS este păstrarea codului CSS in fișiere separate de cele de HTML. Aceasta determină obținerea de cod reutilizabil și ușor de întreținut și totodată îmbunătățește performanța, reducând timpul de încărcare a paginilor. O altă tehnică apreciată este utilizarea comentariilor pentru a oferi explicații suplimentare unde este cazul. Comentariile ajută atât programatorii noi care doresc să continue sau să modifice codul, cât și autorul codului. Utilizarea de clase și ID-uri conferă reutilizabilitate și lizibilitate codului, acestea putând fi aplicate pentru mai multe elemente.

Bootstrap este un framework des folosit, creat de Twitter. Acest framework oferă un set mare de componente predefinite și flexibilitate care simplifică procesul de dezvoltare a site-urilor web. Bootstrap este ușor de utilizat și este avantajos deoarece oferă multe componente reutilizabile, dar are ca dezavantaj dimensiunea mare a fișierelor CSS și JS.

## 2.5. JavaScript

### 2.5.1. Introducere

JavaScript este un limbaj de programare interpretat, orientat pe obiecte, utilizat predominant pentru dezvoltarea de aplicații web interactive. Creat în 1995 de Brendan Eich în timp

ce lucra la Netscape Communications, JavaScript a evoluat rapid pentru a deveni unul dintre cele mai esențiale limbaje de programare în dezvoltarea web. Spre deosebire de HTML și CSS, care sunt folosite pentru structura și stilul paginilor web, JavaScript permite adăugarea de comportamente dinamice și interactive [WWW06].

Inițial, JavaScript a fost conceput pentru a rula pe partea clientului (client-side), ceea ce înseamnă că codul JavaScript este executat direct în browserul utilizatorului. Cu toate acestea, odată cu dezvoltarea Node.js, JavaScript a extins funcționalitatea sa și pe partea serverului (server-side), permițând dezvoltatorilor să scrie cod JavaScript pentru întregul stack de aplicații web.

JavaScript este standardizat prin specificația ECMAScript (ES), care definește caracteristicile și comportamentele limbajului. De-a lungul anilor, au fost lansate mai multe versiuni ale ECMAScript, fiecare aducând îmbunătățiri și noi funcționalități limbajului.

### 2.5.2. Principiile fundamentale ale JavaSript

Sintaxa de bază a JavaScript este similară cu cea a altor limbaje de programare, cum ar fi C și Java, dar are și propriile sale particularități. JavaScript suportă diverse tipuri de date, inclusiv numere, șiruri de caractere (strings), obiecte și funcții. Una dintre caracteristicile cheie ale JavaScript este tipul său dinamic, ceea ce înseamnă că variabilele nu au un tip fix și pot fi reasignate la valori de tipuri diferite.

Mai jos sunt prezentate câteva exemple fundamentale de declarații de variabile în JavaScript:

```
let num = 42;        // variabilă de tip număr

const pi = 3.14;     // constantă de tip număr

let str = "Hello";   // variabilă de tip șir de caractere (string)

let isTrue = true;   // variabilă de tip boolean
```

Funcțiile sunt blocuri de cod reutilizabile care pot fi definite și apelate în orice parte a programului. În JavaScript, funcțiile sunt considerate obiecte de primă clasă, ceea ce înseamnă că pot fi stocate în variabile, transmise ca argumente altor funcții și returnate de alte funcții.

Mai jos este prezentat un exemplu fundamental de declarație de funcție:

```
function greet(name) {
   return "Bună, " + name + "!";}
console.log(greet("Ana"));  // Outputul va fi : Bună, Ana!
```

Manipularea DOM (Document Object Model) este una dintre cele mai comune utilizări ale JavaScript în dezvoltarea web. DOM reprezintă structura documentului HTML ca un arbore de noduri, iar JavaScript poate interacționa cu aceste noduri pentru a modifica conținutul, structura și stilul paginii web. De exemplu, dezvoltatorii pot folosi JavaScript pentru a adăuga sau elimina elemente HTML, pentru a schimba atributele și stilurile CSS ale elementelor existente și pentru a răspunde la evenimentele de utilizator, cum ar fi clicurile și tastările.

Mai jos sunt prezentate câteva exemple fundamentale de declarații de selecție și modificare a elementelor DOM:

```
<!DOCTYPE html>
<html>
<head>
   <title>DOM Manipulation</title>
</head>
<body>
   <div id="myDiv">Hello World!</div>
   <button id="myButton">Click me</button>
   <script>
      // Selecția elementelor DOM
      let div = document.getElementById("myDiv");
      let button = document.getElementById("myButton");
      // Modificarea conținutului
```

```
    div.innerHTML = "Hello JavaScript!";
    // Adăugarea unui eveniment de click
    button.addEventListener("click", function() {
      div.style.color = "red";  // Modificarea stilului });
  </script>
</body>
</html>
```

### 2.5.3. JavaScript în dezvoltarea aplicațiilor web

- Integrarea cu HTML și CSS

JavaScript este esențial pentru crearea de aplicații web interactive și dinamice. Prin integrarea sa cu HTML și CSS, JavaScript permite dezvoltatorilor să manipuleze documentul web și să răspundă la acțiunile utilizatorilor în timp real. De exemplu, dezvoltatorii pot folosi JavaScript pentru a valida formulare sau pentru a crea meniuri derulante.

JavaScript poate fi inclus în paginile HTML fie direct în cadrul etichetelor `<script>`, fie ca fișiere externe. Utilizarea fișierelor externe este o practică recomandată deoarece permite separarea clară a logicii de scripting de structura documentului și de stiluri, ceea ce îmbunătățește mentenabilitatea și performanța paginilor web.

Un alt aspect important al integrării JavaScript cu HTML și CSS este manipularea DOM. Dezvoltatorii pot folosi metode precum getElementById, querySelector sau addEventListener pentru a selecta elemente din document și a le modifica sau pentru a adăuga ascultători de evenimente (event listeners) care declanșează acțiuni în răspuns la interacțiunile utilizatorilor.

- Utilizarea JavaScript pentru Aplicații Single Page (SPA)

Aplicațiile Single Page (SPA) au devenit din ce în ce mai populare datorită experienței de utilizare fluentă și rapidă pe care o oferă. Într-o aplicație SPA, tot conținutul necesar este

încărcat o singură dată și orice interacțiune ulterioară cu aplicația nu necesită reîncărcarea paginii. JavaScript joacă un rol central în construirea SPA-urilor prin gestionarea rutării (routing) și actualizarea dinamică a conținutului paginii.

Framework-urile și bibliotecile JavaScript precum Angular, React și Vue.js au revoluționat modul în care sunt dezvoltate SPA-urile. Aceste instrumente oferă structuri și paradigme de dezvoltare bine definite, care facilitează crearea de componente reutilizabile, gestionarea stării aplicației și optimizarea performanței.

- Securitate

Odată cu puterea și flexibilitatea pe care le oferă JavaScript vin și provocări de securitate. Atacuri precum Cross-Site Scripting (XSS) și Cross-Site Request Forgery (CSRF) sunt comune în aplicațiile web și pot compromite securitatea datelor utilizatorilor și a aplicațiilor. Este esențial ca dezvoltatorii să fie conștienți de aceste riscuri și să implementeze măsuri de securitate adecvate.

Pentru a preveni atacurile XSS, dezvoltatorii ar trebui să valideze și să igienizeze toate datele de intrare ale utilizatorilor. Utilizarea funcțiilor de escape atunci când se afișează datele introduse de utilizatori în paginile web este crucială pentru a preveni injectarea de cod malițios.

CSRF poate fi prevenit prin utilizarea token-urilor anti-CSRF. Aceste token-uri sunt generate pe server și incluse în formularele web, astfel încât orice cerere trimisă de utilizator trebuie să conțină un token valid. Această măsură asigură că cererile sunt trimise doar de surse de încredere.

Implementarea acestor practici de securitate și adoptarea unor best practices generale pentru dezvoltarea JavaScript, cum ar fi modularizarea codului și testarea automată, contribuie la crearea de aplicații web robuste și sigure.

În concluzie, JavaScript este un limbaj de programare versatil și esențial în dezvoltarea web modernă. De la principii fundamentale și manipularea DOM până la utilizarea tehnicilor avansate și dezvoltarea aplicațiilor complexe, JavaScript continuă să evolueze și să ofere noi capabilități pentru dezvoltatori. Înțelegerea profundă a acestui limbaj și adoptarea unor practici de securitate adecvate sunt esențiale pentru crearea de aplicații web interactive, performante și sigure.

## 2.6. XAMPP

### 2.6.1. Introducere

XAMPP este un pachet de software open-source care oferă o soluție pentru dezvoltarea și testarea aplicațiilor web. Numele XAMPP este acronim pentru X(platformă multiplă), Apache, MySQL, PHP și Perl. XAMPP este un pachet software ce pune la dispoziția dezvoltatorilor web un server Apache, un sistem prin care se pot gestiona bazele de date MySQL și suport pentru limbajele de programare PHP și Perl. Toate acestea fac din XAMPP un software ideal pentru dezvoltatorii web [WWW03].

XAMPP a fost dezvoltat de compania Apache Friends în anul 2002. De-a lungul anilor a evoluat, devenind unul dintre cele mai populare pachete de software folosite în dezvoltarea aplicațiilor și a site-urilor web. XAMPP este ușor de utilizat și este suportat pe mai multe sisteme de operare, inclusiv Windows, Linux și macOS. XAMPP este actualizat constant pentru a suporta noile tehnologii și versiuni ale componentelor pe care le pune la dispoziție.

### 2.6.2. Instalarea și configurarea XAMPP

XAMPP poate fi descărcat de pe site-ul Apache Friends oficial, unde se pot găsi versiuni pentru Windows, Linux și macOS. Pentru instalare trebuie urmați câțiva pași simpli: selectarea componentelor care se doresc instalate, alegerea directorului de instalare (implicit, C:\xampp).

Panoul de Control XAMPP este interfața principală de unde sunt gestionate serviciile. După cum se poate observa din figura Fig 2.3, panoul de control conține butoane de start și stop al componentelor Apache, MySQL, FileZilla, Mercury și Tomcat (Apache și MySQL sunt pornite). În plus, fiecare serviciu are un buton care deschide fișierele de configurare, deci permite editarea și configurarea acestora. În consola de jos se pot observa mesaje relevante.


Fig. 2.3

Apache poate fi configurat accesând fișierul principal de configurare (httpd.conf), unde se pot seta diferite configurații, precum porturile pe care rulează. De exemplu, schimbarea portului de la 80 (implicit) la 8080 se poate face folosind: Listen 8080. Configurarea MySQL se face în fișierul my.ini; acesta poate fi accesat apăsând pe butonul de config de la MySQL sau din directorul ‚xampp/mysql/bin'. De aici se pot modifica setările de memorie, cache și alte setări avansate. De exemplu, pentru a schimba portul de la 3306 (portul implicit) la 3307 se folosește: port=3307. Configurarea PHP se face în fișierul php.ini, care poate fi accesat la butonul de config din dreptul Apache sau în directorul ‚xampp'. Aici se pot configura memoria, extensiile încărcate și alte opțiuni. De exemplu pentru mărirea limitei de încărcare a fișierelor la 50M se utilizează: upload_max_filesize=50M;

### 2.6.3. XAMPP în dezvoltarea și testarea aplicațiilor web

Crearea unui proiect folosind platforma XAMPP este relativ simplă. În directorul 'htdocs', situat în directorul de instalare XAMPP se vor plasa toate fișierele proiectului creat. O tehnică apreciată este crearea unui folder nou pentru fiecare proiect, pentru a menține o structură și o ordine clară. Apoi în folder-ul proiectului curent se vor crea fișierele de HTML, PHP, JS și altele, în funcție de preferințele dezvoltatorilor. Accesul la proiectul creat se face pornind serverul Apache și MySQL din panoul de control XAMPP, după care din browser se poate accesa 'http://localhost/numele_proiectului' pentru a vedea interfața grafică a acestuia.

Administrarea bazelor de date poate fi realizată din interfața phpMyAdmin. XAMPP include phpMyAdmin, care poate fi accesat din browser cu link-ul 'http://localhost/phpmyadmin'. Cu ajutorul phpMyAdmin se pot gestiona bazele de date MySQL sau MariaDB. Prin interfața phpMyAdmin se facilitează modificarea, crearea și ștergerea bazelor de date și a tabelelor conținute de acestea. Crearea unei baze de date se pate face din fila 'Databases' prin introducerea unui nume pentru baza de date și apăsarea butonului 'Create'. Tabelele din baza de date pot fi create folosind interogări SQL. PhpMyAdmin permite executarea acestora direct din interfața sa. La fila 'SQL' se poate introduce codul SQL și se poate executa prin apăsarea butonului 'Go'.

Pentru logare și debugging, XAMPP are fișiere de log și modul de debugging PHP. XAMPP conține fișiere de log pentru toate serviciile, care pot fi accesate în două moduri, cu click pe butonul de Log din dreptul serviciului la care se dorește logare sau în subdirectoarele XAMPP. Prin verificarea fișierelor de log se pot identifica și rezolva probleme de configurare sau erori de executare.

XAMPP suportă managementul proiectelor și colaborarea prin sisteme de suport al versiunilor și posibilitatea configurării proiectelor multiple. Utilizarea sistemelor de control al versiunilor, de exemplu Git, în colaborare cu XAMPP este benefic pentru gestionarea codului și colaborare în echipă, dacă este cazul. Crearea unui repository de git în folderul proiectului și utilizarea unei platforme precum GitHub sau GitLab ajută la crearea de versiuni a proiectului, astfel menținând o structurare și o claritate a proiectului. Totodată, XAMPP susține gestionarea de proiecte multiple prin configurarea de virtual hosts în Apache. Astfel, se pot defini mai multe domenii locale care să direcționeze către diferite directoare de proiect:

```
<VirtualHost *:80>
    DocumentRoot 'C:/xampp/htdocs/primul_proiect'
    ServerName primul_proiect.local
</VirtualHost>
<VirtualHost *:80>
    DocumentRoot 'C:/xampp/htdocs/alDoilea_proiect'
    ServerName alDoilea_proiect.local
</VirtualHost>
```

După definirea domeniilor locale este necesar ca aceste domenii să fie adăugate în fișierul 'hosts' al sistemului de operare:

```
127.0.0.1 primul_proiect.local
127.0.0.1 alDoilea_proiect.local
```

În concluzie, XAMPP este o platformă esențială pentru dezvoltarea și testarea aplicațiilor web locale, deoarece oferă un pachet cu toate componentele necesare pentru crearea unei aplicații web completă și robustă. XAMPP este avantajos pentru dezvoltatori, deoarece le permite o instalare și configurare ușoară, fără a necesita gestionarea efectivă a serverului. Alte beneficii care sporesc popularitatea XAMPP este interfața intuitivă, suportul pentru diverse sisteme de operare (Windows, Linux, macOS), simularea unui server web, flexibilitate și scalabilitate obținută prin configurarea de proiecte multiple.

## 2.7. Git

### 2.7.1. Introducere

Git este un sistem distribuit de control al versiunilor dezvoltat în anul 2005 de către Linus Torvalds. A fost creat inițial pentru a sprijini dezvoltarea kernelului Linux. Git este eficient în gestionarea proiectelor deoarece permite utilizatorilor să mențină un istoric al versiunilor de cod și să colaboreze între ei.

Git oferă multiple beneficii utilizatoriilor, câteva dintre acestea sunt:

- Controlul versiunilor

Git permite controlul versiunilor prin funcționalitatea de commit. Realizarea unui commit se referă la salvarea stării/versiunii codului la moment dat cu o descriere sugestivă a funcționalității implementate. Commiturile sunt folositoare, deoarece se poate observa ușor evoluția proiectului, modificările făcute și problemele apărute.

- Distribuitate

Fiecare clonă a unui repository conține istoricul complet al proiectului. Git oferă posibilitatea dezvoltatorilor de a lucra offline și de a-și sincroniza mai apoi modificările când au acces la rețea.

- Eficient și rapid

Eficiența Git constă în faptul că este ideal pentru gestionarea de proiecte mari, deoarece împărțirea taskurilor în commituri ajută la organizarea și mentenanța codului. Efectuarea de operații local este rapidă, iar dimensiunea repository-ului este redusă printr-un sistem de compresie a datelor.

- Gestionarea branch-urilor

O funcționalitate de bază a Git-ului este crearea de branch-uri. Branchu-rile izolează lucrul la o funcționalitate nouă sau un bug fix fără să afecteze codul branch-ului principal, unde se află codul pentru proiect până în momentul curent. Git permite o gestionare simplă și eficientă a branch-urilor prin operații de merge și rebase  care facilitează integrarea modificărilor din diferite branch-uri.

### 2.7.2. GitHub și GitHub Desktop

GitHub este o platformă ce conține repository-urile de Git. GitHub adaugă funcționalități suplimentare pentru gestionarea de proiecte și colaborarea între mai mulți utilizatori dacă este cazul. GitHub suportă stocarea, gestionarea și revizuirea codului într-un mod organizat și eficient. Totodată, GitHub oferă câteva funcționalități cheie, cum ar fi: repository-uri remote, pull request-uri și issue tracking.

Prin intermediul repository-ului remote, GitHub oferă accesibilitate la repository-uri oriunde și oricând. Acestea pot fi publice sau private, aceasta reoferindu-se la modul de acces asupra lor. Cele private sunt disponibile tuturor utilizatorilor GitHub, ceea ce o face o platformă ideală pentru proiectele open-source. Pull request-urile sunt esențiale pentru lucrul în echipă, permițând revizuiri și modificări la codul brancu-rilor înainte să fie integrate în branch-ul principal. Astfel se evită introducerea de bug-uri sau funcționalități nedorite în proiect.

GitHub Desktop este o aplicație grafică pentru GitHub. Interfața intuitivă simplifică utilizarea Git și eficientizează procesul de versionare și gestionare al proiectelor. GitHub Desktop este disponibil pentru Windows și macOS și oferă funcționalități avansate pentru gestionarea repository-urilor de Git.

GitHub Desktop are o interfață prietenoasă pentru utilizatori Fig 2.4 afișând într-un mod clar modificările, commit-urile, branch-urile și celelalte funcționalități. Astfel, dezvoltatorii pot vizualiza și gestiona modificările din proiecte și pot crea, integra și schimba cu ușurință între branch-uri și proiecte. În imaginea de mai jos, putem observa mai multe elemente și funcționalități ale interfeței GitHub Desktop.

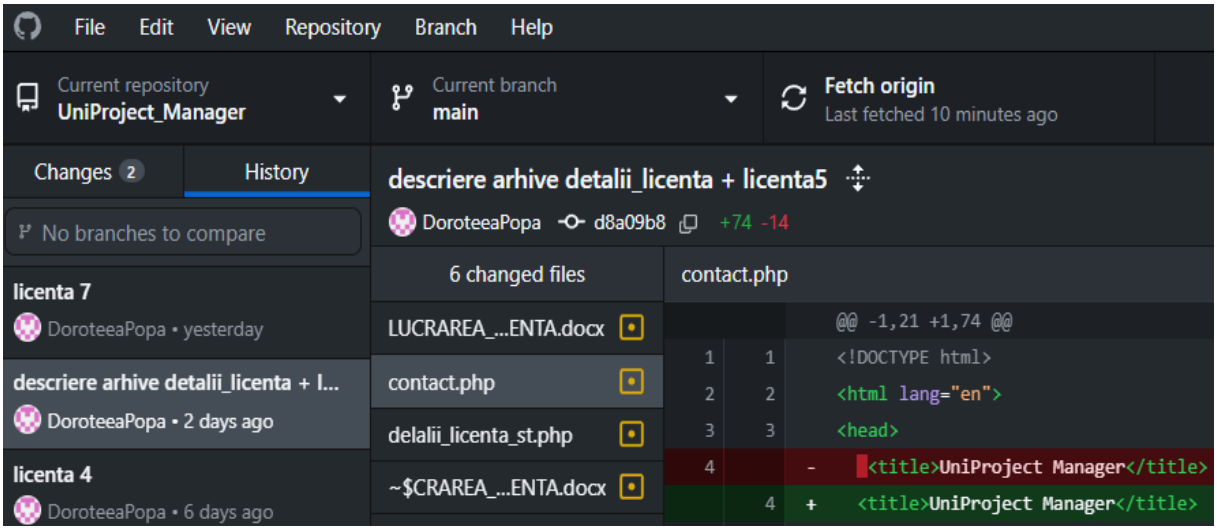



Fig. 2.4

Bara de navigare principală conține meniuri precum File, Edit, View, Repository, Branch și Help, care oferă acces la diverse funcționalități și setări ale GitHub Desktop. Meniul File oferă opțiuni pentru a deschide, clona sau crea repository-uri noi, Edit include opțiuni de modificare a setărilor și preferințelor, View permite ajustarea aspectului interfeței și vizualizarea detaliilor commit-urilor, Repository permite gestionarea repository-urilor, cum ar fi sincronizarea și setările remote.

În partea de sus a interfeței, vedem informații despre repository-ul curent, numit UniProject_Manager, și branch-ul curent selectat, numit main. Utilizatorii pot schimba repository-urile folosind meniul derulant și pot schimba între branch-uri existente sau pot crea branch-uri noi. De asemenea, funcționalitatea Fetch origin permite utilizatorilor să sincronizeze repository-ul local cu cel remote, obținând ultimele modificări de pe GitHub. Indicatorul Last fetched 10 minutes ago arată ultima dată când repository-ul a fost sincronizat.

Panoul de modificări afișează modificările nesalvate (unstaged changes) în repository. Utilizatorii pot vedea fișierele modificate și pot selecta care dintre acestea să fie incluse într-un commit. Aceștia pot face click pe fiecare fișier pentru a vedea diferențele (diff) linie cu linie, comparând modificările față de versiunea anterioară. De asemenea, există o secțiune pentru istoricul commit-urilor, care oferă o vedere de ansamblu asupra istoricului commit-urilor din repository. Fiecare commit este afișat cu mesajul său, autorul și timestamp-ul.

Lista de fișiere modificate afișată în panoul de modificări arată fișierele care au fost modificate, create sau șterse. Fiecare fișier are o legendă de culori care indică tipul modificărilor, de exemplu, verde pentru adăugări și roșu pentru ștergeri. Utilizatorii pot deschide fiecare fișier modificat pentru a vedea diferențele. Linia originală este afișată în roșu, iar linia modificată este afișată în verde. Această funcționalitate este esențială pentru a înțelege exact ce modificări au fost făcute înainte de a le include într-un commit.

### 2.7.3. Mod de utilizare git

Utilizarea Git în mod eficient necesită înțelegerea unor fluxuri de lucru comune și adoptarea unor bune practici care să asigure coerența și calitatea codului.

- Fluxuri de lucru Git

Unul dintre cele mai populare fluxuri de lucru este Git Flow, un model de dezvoltare bazat pe branch-uri, propus de Vincent Driessen. Acesta include branch-uri pentru dezvoltare, funcționalități, release-uri și bug fix-uri, permițând un flux de lucru organizat și structurat. În

Git Flow, branch-ul main conține versiunea stabilă și lansată a proiectului, iar branch-ul develop conține codul pentru dezvoltarea curentă și este folosit pentru integrarea continuă. Branch-urile de funcționalitate (feature) sunt folosite pentru dezvoltarea funcționalităților noi, care sunt apoi integrate în develop. Branch-urile de release (release) sunt folosite pentru pregătirea versiunilor de producție, incluzând ultimele modificări din develop. Branch-urile de bug fix (hotfix) sunt folosite pentru rezolvarea rapidă a bug-urilor critice din main.

- Practici apreciate în utilizarea Git

Adoptarea unor bune practici în utilizarea Git este esențială pentru menținerea calității codului și a eficienței în colaborare. Una dintre cele mai importante practici este scrierea de mesaje de commit clare și descriptive. Un mesaj de commit ar trebui să explice clar ce modificări au fost făcute și de ce, pentru a facilita înțelegerea istoricului proiectului de către toți membrii echipei. Este recomandat să folosiți un format standard pentru mesajele de commit, care să includă un titlu concis urmat de o descriere detaliată.

De asemenea, este esențial să se facă commit-uri frecvente și mici. Aceasta permite urmărirea mai ușoară a modificărilor și facilitează identificarea și rezolvarea problemelor. Commit-urile ar trebui să fie atomice, adică să conțină o singură modificare logică sau funcționalitate. Evitați commit-urile mari și complexe care îngreunează revizuirea și înțelegerea modificărilor.

Sincronizarea regulată a branch-urilor cu repository-ul remote este crucială pentru a menține consistența codului și a evita conflictele. Utilizați comanda git pull pentru a aduce modificările recente din branch-ul remote în branch-ul local și rezolvați conflictele imediat ce apar. De asemenea, utilizați git push pentru a trimite modificările locale către repository-ul remote.

În concluzie, adoptarea unor fluxuri de lucru bine definite și a unor bune practici în utilizarea Git poate îmbunătăți semnificativ eficiența și calitatea dezvoltării software. Prin scrierea de mesaje de commit clare, efectuarea de commit-uri frecvente și mici, utilizarea branch-urilor separate pentru funcționalități și bug fix-uri și sincronizarea regulată a branch-urilor, se asigură un flux de lucru organizat și eficient, menținând în același timp calitatea și coerența codului sursă.

## 2.8. Visual Studio Code

### 2.8.1. Introducere

Visual Studio Code (VS Code) este un editor de cod sursă dezvoltat de Microsoft, care a devenit extrem de popular datorită funcționalităților sale puternice și flexibilității. Este gratuit, open-source și rulează pe diverse platforme, inclusiv Windows, macOS și Linux.

Visual Studio Code a fost lansat de Microsoft în aprilie 2015 și a devenit rapid unul dintre cele mai populare editoare de cod datorită caracteristicilor sale puternice și suportului extins pentru multiple limbaje de programare. Este conceput pentru a oferi o experiență de editare a codului eficientă și ușor de utilizat, integrând funcționalități avansate precum IntelliSense, debugging, și integrare Git.

Printre caracteristicile cheie ale Visual Studio Code se numără:

- **IntelliSense:** Completarea automată a codului bazată pe context și informații despre tipuri, oferind sugestii inteligente pentru completarea codului.
- **Debugging:** Un sistem integrat de debugging care permite dezvoltatorilor să își testeze și să își depaneze aplicațiile direct din editor.
- **Controlul Versiunilor:** Suport nativ pentru Git, facilitând gestionarea versiunilor și colaborarea în echipă.
- **Extensii:** O piață vastă de extensii care permite personalizarea și extinderea funcționalităților editorului pentru a se potrivi nevoilor specifice ale dezvoltatorilor.
- **Suport pentru Multiple Limbaje de Programare:** Visual Studio Code suportă o gamă largă de limbaje de programare, incluzând PHP, JavaScript, Python, Java, C++, și multe altele.

### 2.8.2. Interfața Visual Studio Code

Interfața Visual Studio Code Fig 2.5 este concepută pentru a fi intuitivă și ușor de utilizat, oferind acces rapid la funcționalitățile esențiale pentru dezvoltarea de software. În imaginea de mai jos, putem observa mai multe componente cheie ale interfeței.

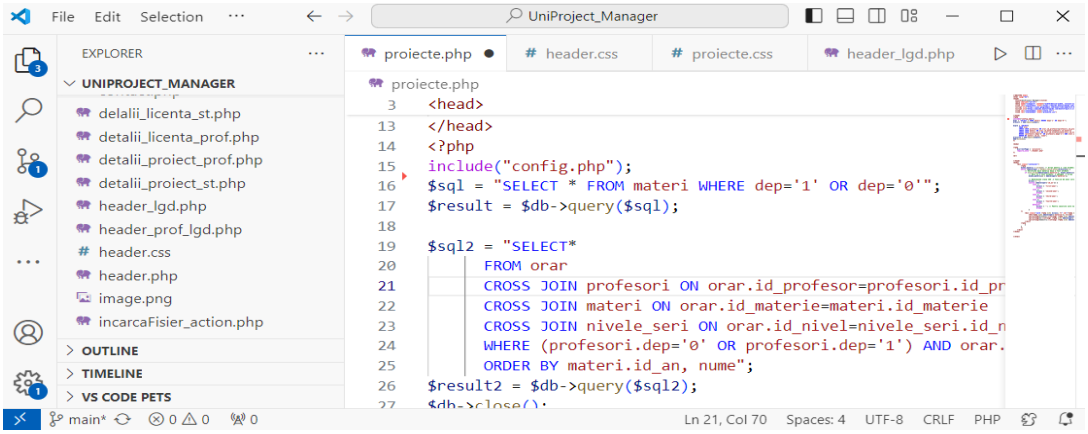



Fig 2.5

**Explorer:** În partea stângă a interfeței, avem panoul Explorer, care afișează structura proiectului. Acesta permite utilizatorilor să navigheze între fișiere și directoare, să creeze, să șteargă sau să redenumească fișiere, și să deschidă fișiere pentru editare. În imagine, vedem că proiectul curent se numește "UNIPROJECT_MANAGER", iar diverse fișiere PHP sunt listate în structura proiectului. Indicatorii pentru modificările nesalvate și numărul de modificări sau conflicte sunt vizibili, ajutând la gestionarea eficientă a proiectului.

**Tab-uri de Editare:** În partea de sus a interfeței, vedem tab-urile de editare, care permit utilizatorilor să lucreze simultan la mai multe fișiere deschise. În imagine, fișierul "proiecte.php" este deschis și editat. Utilizatorii pot comuta între fișierele deschise folosind aceste tab-uri.

**Editor de Cod:** Editorul de cod ocupă cea mai mare parte a interfeței și este locul unde utilizatorii scriu și editează codul. În imagine, vedem codul PHP din fișierul "proiecte.php". Codul include o instrucțiune "include" pentru a adăuga conținutul unui alt fișier PHP, o interogare SQL și o execuție a acestei interogări folosind un obiect de bază de date.

**Bara de Status:** În partea de jos a interfeței, vedem bara de status, care oferă informații rapide despre starea curentă a editorului și a proiectului. Aceasta poate afișa informații despre ramura Git curentă, erori și avertismente în cod, și alte detalii relevante. În imagine, vedem că ramura curentă este "main" și sunt afișate indicatoare pentru erori și avertismente, precum și detalii despre cod, cum ar fi limbajul de programare, formatul fișierului și setările de indentare.

**Panoul de Activități Laterale:** În partea stângă a interfeței, sub panoul Explorer, vedem panoul de activități laterale, care include pictograme pentru diverse funcționalități, cum ar

fi explorarea fișierelor, controlul versiunilor, căutarea în proiect și accesul la extensii. Pictogramele vizibile includ Explorer, pentru navigarea între fișierele și directoarele proiectului, Search, pentru căutarea textului în întregul proiect, Source Control, pentru integrarea cu sistemul de control al versiunilor (Git în acest caz), și Extensions, pentru gestionarea extensiilor instalate și descărcarea de noi extensii.

### 2.8.3. Utilizarea în dezvoltarea web

Visual Studio Code este un instrument extrem de puternic pentru dezvoltarea web, datorită suportului său robust pentru limbaje de programare web precum HTML, CSS, JavaScript și framework-uri moderne. Un aspect important al utilizării Visual Studio Code în dezvoltarea web este suportul său excelent pentru PHP, un limbaj server-side popular pentru dezvoltarea aplicațiilor web.

Visual Studio Code oferă suport complet pentru editarea HTML și CSS, inclusiv completarea automată a tag-urilor, sugestii pentru atribute și valori, și verificarea sintaxei. Dezvoltatorii pot folosi extensii precum Live Server pentru a lansa un server local și a vedea modificările în timp real pe măsură ce editează fișierele HTML și CSS.

Visual Studio Code oferă un suport extins pentru PHP prin intermediul extensiilor disponibile pe marketplace-ul său. Visual Studio Code are funcționalități avansate precum IntelliSense, care oferă completarea automată a codului bazată pe context și informații despre tipurile de date, diagnosticare a erorilor în timp real și navigare ușoară prin cod.

Un alt aspect important al dezvoltării web cu Visual Studio Code este debugging-ul și controlul versiunilor. Editorul include un debugger integrat care permite dezvoltatorilor să seteze puncte de oprire (breakpoints), să examineze variabile și să urmărească fluxul de execuție al aplicației lor.

Visual Studio Code este un editor de cod sursă extrem de versatil și puternic, care oferă o gamă largă de funcționalități pentru a sprijini dezvoltatorii în toate aspectele procesului de dezvoltare. De la o interfață intuitivă și ușor de utilizat, până la suportul robust pentru multiple limbaje de programare și integrarea cu instrumente de debugging și control al versiunilor, Visual Studio Code este un instrument indispensabil pentru orice dezvoltator.

# 3. Arhitectura aplicației

## 3.1. Baza de date

Baza de date folosită se numește `depcie` și conține în total 17 tabele, dintre care 5 tabele (`orar`, `profesori`, `materi`, `niveles_seri`, `tipuri`) au fost importate din baza de date a departamentului de calculatoare și inginerie electrică din cadrul facultății de inginerie ULBS. Importul a fost realizat direct din interfața phpMyAdmin folosind fișierul `depcie.sql`. Importul se face astfel:

1. Accesați interfața phpMyAdmin.

2. Selectați baza de date în care doriți să importați tabelele.

3. Mergi la tab-ul "Import".

4. Apăsați pe "Choose File" și selectați fișierul `depcie.sql` din calculator și apăsați "Go".

Această procedură va importa tabelele și structura lor exact așa cum sunt definite în fișierul SQL. Restul de 12 tabele au fost create pentru a ajuta studenții să își gestioneze proiectele (proiect și licență la anii terminali), profesorii să gestioneze proiectele predate și studenții participanți, iar secretara să gestioneze studenții și profesorii.

## 3.2. Descrierea tabelelor și relațiilor dintre ele

În figura Fig. 3.1 este reprezentată o diagramă cu toate tabelele din baza de date depcie și toate relațiile dintre acestea. Relațiile dintre tabele sunt prezentate prin linii colorate pentru a facilita vizualizarea și înțelegerea acestora.

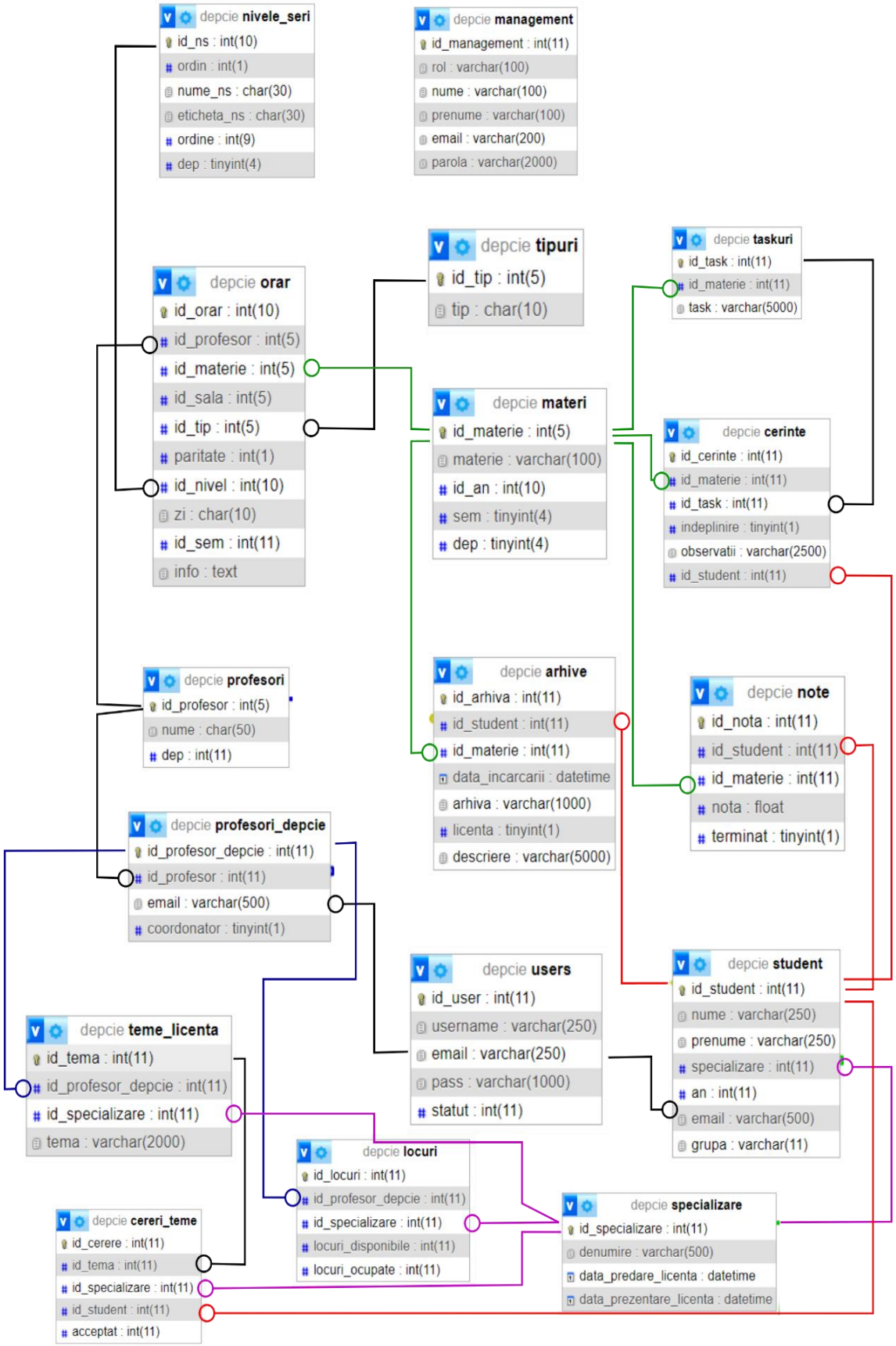

depcie nivele_seri
id_ns : int(10)
ordin : int(1)
nume_ns : char(30)
eticheta_ns : char(30)
ordine : int(9)
dep : tinyint(4)
depcie management
id_management : int(11)
rol : varchar(100)
nume : varchar(100)
prenume : varchar(100)
email : varchar(200)
parola : varchar(2000)
depcie tipuri
id_tip : int(5)
tip : char(10)
depcie taskuri
id_task : int(11)
id_materie : int(11)
task : varchar(5000)
depcie orar
id_orar : int(10)
id_profesor : int(5)
id_materie : int(5)
id_sala : int(5)
id_tip : int(5)
paritate : int(1)
id_nivel : int(10)
zi : char(10)
id_sem : int(11)
info : text
depcie materi
id_materie : int(5)
materie : varchar(100)
id_an : int(10)
sem : tinyint(4)
dep : tinyint(4)
depcie cerinte
id_cerinte : int(11)
id_materie : int(11)
id_task : int(11)
indeplinire : tinyint(1)
observatii : varchar(2500)
id_student : int(11)
depcie profesori
id_profesor : int(5)
nume : char(50)
dep : int(11)
depcie arhive
id_arhiva : int(11)
id_student : int(11)
id_materie : int(11)
data_incarcarii : datetime
arhiva : varchar(1000)
licenta : tinyint(1)
descriere : varchar(5000)
depcie note
id_nota : int(11)
id_student : int(11)
id_materie : int(11)
nota : float
terminat : tinyint(1)
depcie profesori_depcie
id_profesor_depcie : int(11)
id_profesor : int(11)
email : varchar(500)
coordonator : tinyint(1)
depcie users
id_user : int(11)
username : varchar(250)
email : varchar(250)
pass : varchar(1000)
statut : int(11)
depcie student
id_student : int(11)
nume : varchar(250)
prenume : varchar(250)
specializare : int(11)
an : int(11)
email : varchar(500)
grupa : varchar(11)
depcie teme_licenta
id_tema : int(11)
id_profesor_depcie : int(11)
id_specializare : int(11)
tema : varchar(2000)
depcie locuri
id_locuri : int(11)
id_profesor_depcie : int(11)
id_specializare : int(11)
locuri_disponibile : int(11)
locuri_ocupate : int(11)
depcie specializare
id_specializare : int(11)
denumire : varchar(500)
data_predare_licenta : datetime
data_prezentare_licenta : datetime
depcie cereri_teme
id_cerere : int(11)
id_tema : int(11)
id_specializare : int(11)
id_student : int(11)
acceptat : int(11)

Fig. 3.1

- Tabelele Importate

1. orar:

- Conține informații despre orarul cursurilor.

- Chei străine: `id_profesor`, `id_materie`, `id_nivel`, `id_tip`.

2. profesori:

- Conține informații despre profesori.

- Cheie primară: `id_profesor`.

3. materi:

- Conține informații despre materiile predate.

- Cheie primară: `id_materie`.

4. niveles_seri:

- Conține informații despre nivelele și seriile de cursuri.

- Cheie primară: `id_ns` care reprezintă semigrupa studenților.

5. tipuri:

- Conține informații despre tipurile de cursuri (e.g., curs, seminar).

- Cheie primară: `id_tip`; pentru aplicația aceasta avem nevoie de proiecte, deci id_tip=4.

- Tabelele Create

6. taskuri:

- Conține informații despre sarcinile asociate cu fiecare materie.

- Cheie primară: `id_task`.

- Cheie străină: `id_materie`.

7. cerinte:

- Conține cerințele specifice fiecărui student pentru fiecare materie a sa. Cu ajutorul acestui tabel se ține evidența progresului fiecărui student (coloana îndeplinire).

- Cheie primară: `id_cerinte`.

- Chei străine: `id_task`, `id_materie`, `id_student`.

8. note:

- Conține notele studenților pentru fiecare materie și dacă studentul a marcat sau nu proiectul ca terminat.

- Cheie primară: `id_nota`.

- Chei străine: `id_student`, `id_materie`.

9. arhive:

- Conține arhivele încărcate de studenți pentru fiecare materie. De asemenea, se reține data încărcării arhivei, dacă este pentru licență(1) sau nu(0) și descrierea dată de student pentru aceasta.

- Cheie primară: `id_arhiva`.

- Chei străine: `id_student`, `id_materie`.

10. profesori_depcie:

- Conține informații suplimentare despre profesori, inclusiv emailul lor și dacă sunt coordonatori de licență(1) sau nu(0).

- Cheie primară: `id_profesor_depcie`.

- Cheie străină: `id_profesor`.

11. users:

- Conține informații despre utilizatorii sistemului. Daca sunt studenți la statut vom avea 0, pentru profesori vom avea 1 și pentru profesori coordonatori vom avea 2.

- Cheie primară: `id_user`.

12. teme_licenta:

- Conține temele de licență, profesorul care le-a dat și specializarea pentru care au fost date .

- Cheie primară: `id_tema`.

- Chei străine: `id_profesor_depcie`, `id_specializare.

13. cereri_teme:

- Conține cererile de teme pentru licență trimise de studenți și dacă a fost acceptată(2), refuzată(1) sau în așteptare(0) în coloana acceptat.

- Cheie primară: `id_cerere`.

- Chei străine: `id_tema`, `id_student`, `id_specializare`.

14. locuri:

- Conține informații despre locurile disponibile pentru licență. Cu ajutorul acestui tabel se ține cont de fiecare coordonator cu specializarea la care coordonează.

- Cheie primară: `id_locuri`.

- Chei străine: `id_profesor_depcie`, `id_specializare.

15. specializare:

- Conține informații despre specializările disponibile și data de predare și prezentare a licenței la fiecare specializare.

- Cheie primară: `id_specializare`.

16. student:

- Conține informații despre studenți.

- Cheie primară: `id_student`.

- Cheie străină: `id_specializare`.

- Relațiile Între Tabele

Liniile verzi reprezintă relațiile în care `id_materie` din tabelul `materi` este cheie primară, iar `id_materie` din tabelele `taskuri`, `cerinte`, `note` și `arhive` sunt chei străine. O cheie primară este un identificator unic pentru fiecare înregistrare dintr-un tabel, în timp ce o cheie străină este un câmp dintr-un tabel care face legătura cu o cheie primară din alt tabel. Aceasta ajută la menținerea integrității referențiale între tabele.

Liniile albastre conectează tabelul `profesori_depcie` (cheie primară `id_profesor_depcie`) cu tabelele teme-licenta și locuri (cheie străină `id_profesor_depcie`).

Aceasta arată care profesorii sunt parte a departamentului de calculatoare și inginerie electrică și au un rol important în gestionarea studenților și în desfășurarea licenței.

Liniile roșii arată relațiile între `student` (cheie primară `id_student`) și tabelele `cerinte`, `note`, `arhive`, `cereri_teme`. Aceste relații sunt esențiale pentru a urmări performanța și activitățile studenților în cadrul diferitelor materii și teme de licență.

Liniile roz conectează tabelul `specializare` (cheie primară `id_specializare`) cu tabelele `teme_licenta`, `cereri_teme`, `locuri` și `student`. Aceste relații sunt importante pentru a gestiona specializările, temele de licență și alocarea locurilor pentru licență.

Această structură a bazei de date permite o gestionare eficientă și integrată a informațiilor legate de studenți, profesori, materii și teme de licență, asigurând în același timp integritatea și coerența datelor.

# 4. Implementarea practică

## 4.1. Actori și use-cases

Aplicația web dezvoltată este un sistem de gestionare a proiectelor universitare din cadrul departamentului Calculatoare și Inginerie Electrică, destinat să faciliteze interacțiunea între studenți, profesori, secretariat, coordonatori de licență și administratori. Scopul principal al aplicației este de a centraliza informațiile despre proiecte, de a urmări progresul studenților și de a simplifica procesul de evaluare și gestionare a datelor.

Diagrama de use-cases prezentată in figura Fig. 4.1 reflectă interacțiunile posibile ale diferitelor tipuri de utilizatori (actori) cu sistemul. Actorii identificați sunt: Vizitator, Student, Profesor, Secretară, Coordonator de licență și Administrator.

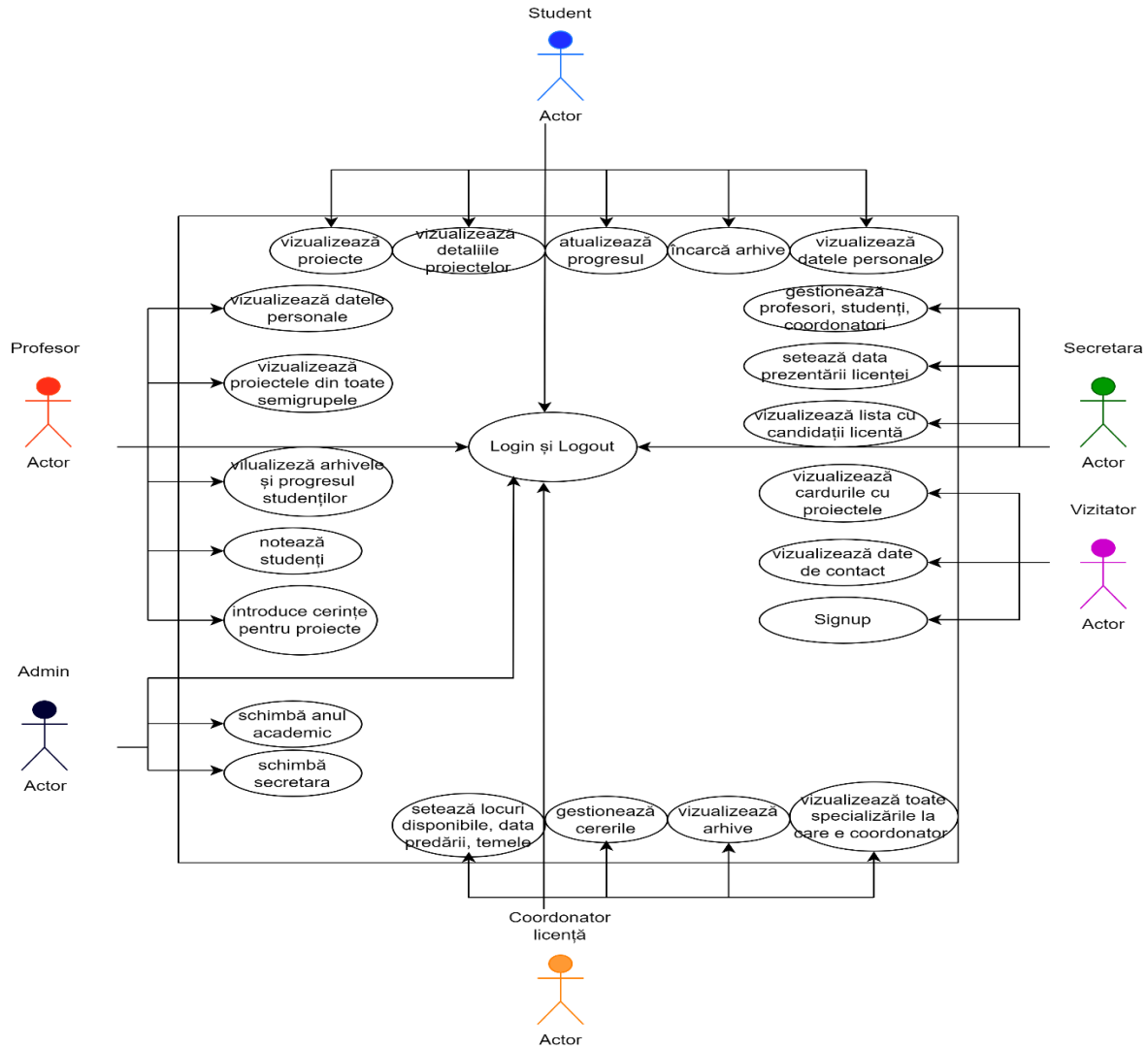



Fig. 4.1.

Fiecare actor are un set specific de funcționalități (use-cases) pe care le poate accesa sau executa în cadrul sistemului:

- **Vizitator**: Utilizator neautentificat care poate doar să vizualizeze anumite informații publice.
- **Student**: Utilizator autentificat care poate vizualiza și gestiona proiectele proprii, încărca arhive și actualiza progresul.
- **Profesor**: Utilizator autentificat care poate vizualiza, nota și gestiona proiectele studenților, introduce cerințe pentru proiecte și vizualiza progresul acestora.
- **Secretară**: Utilizator autentificat responsabil cu gestionarea datelor administrative, setarea datei de prezentare a licențelor și vizualizarea listelor de candidați.
- **Coordonator de licență**: Utilizator autentificat care gestionează aspectele specifice legate de proiectele de licență, incluzând setarea locurilor disponibile, a temelor de licență și a datei de predare a acestora.
- **Administrator**: Utilizator autentificat care are rolul de a actualiza anul și grupele studenților și de a schimba secretara dacă este cazul.

## 4.2. Vizitator use-cases

Vizitatorul reprezintă orice persoană care accesează aplicația fără a avea un cont autentificat. Această categorie de utilizatori are acces limitat la funcționalitățile aplicației, putând doar să vizualizeze informațiile publice disponibile despre proiecte. Vizitatorii pot să se înregistreze pentru a obține acces complet la funcționalitățile oferite de aplicație. Funcționalitățile disponibile pentru vizitator includ:

- **Înregistrare (Signup)**: Vizitatorii au opțiunea de a se înregistra în sistem pentru a obține acces complet la funcționalitățile aplicației, completând un formular cu informațiile necesare.

- **Vizualizarea cardurilor cu proiectele**: Vizitatorii pot vedea o prezentare generală a proiectelor disponibile, inclusiv titlurile și descrierile sumare.

- **Vizualizarea datelor de contact**: Vizitatorii pot accesa informațiile de contact ale directorului de departament, ale secretariatului și ale administratorului site-ului UniProject Manager.

**Implementare**

**A. Header-ul (Antetul)**


Fig 4.2

După cum se poate observa în figura Fig. 4.2 , header-ul aplicației UniProject Manager include meniul de navigare și logo-ul universității, oferind utilizatorilor acces rapid la paginile principale: Acasă, Proiecte și Contact. Meniul de navigare utilizează clasa CSS `'active'` pentru a evidenția pagina curentă, iar butoanele de autentificare și înregistrare sunt plasate în partea dreaptă a header-ului pentru accesibilitate. Header-ul este stilizat pentru a avea un fundal albastru închis, cu text alb și o umbră subtilă la text. Header-ul aplicației UniProject Manager este implementat în fișierul header.php. Header-ul este implementat într-un fișier separat pentru a evita implementarea de cod duplicat pentru fiecare pagină disponibilă, acesta fiind inclus unde este nevoie.

**B. Signup**

Formularele pentru autentificare și înregistrare sunt afișate în modale (pop-up-uri) pentru a îmbunătăți experiența utilizatorului. Aceste formulare permit utilizatorilor să introducă email-ul, parola și alte detalii relevante. În cazul formularului de înregistrare, utilizatorii pot selecta statutul lor (Student, Profesor, Profesor Coordonator). Butoanele de trimitere și anulare sunt clar vizibile și funcționale.

Scriptul JavaScript este responsabil pentru controlul afișării și ascunderii modalelor. Modalurile sunt închise automat când utilizatorul face click în afara lor, îmbunătățind astfel experiența de utilizare. Acest script simplu asigură funcționalitatea esențială a modalelor fără a fi nevoie de biblioteci suplimentare.

Afișarea modalelor se face în funcție de id, astfel:

```
<div id="id01" class="modal">

  <form class="modal-content animate" action="login_action.php" method="post">

    ...

  </form>

</div>
```

La trimiterea formularului de Signup se accesează fișierul signup_action.php unde sunt implementate etapele principale ale procesului de înregistrare. Codul începe cu inițializarea unei sesiuni PHP și includerea fișierului de configurare pentru baza de date (config.php). Sesiunea PHP este necesară pentru a păstra datele utilizatorului pe parcursul utilizării aplicației.

Datele introduse de utilizator (username, email, parola) în formularul de înregistrare sunt preluate și folosim instrucțiuni escape pentru a preveni atacurile de tip SQL Injection.

Exemplu: `$statut = mysqli_real_escape_string($db, $_POST['status']);`

În funcție de statutul selectat de utilizator la înregistrare(student, profesor, profesor coordonator), se execută o interogare pentru a verifica dacă utilizatorul există în baza de date în tabelul de studenți sau profesori pentru valida că utilizatorul are într-adevăr statutul selectat. După validare se încarcă datele acestuia într-o variabilă, $user_student respectiv $user_prof. Pe lângă aceste verificări se verifică și dacă utilizatorul există deja în baza de date în tabelul de users  pe baza email-ului. Dacă utilizatorul există, se redirecționează cu un mesaj de eroare.

Dacă utilizatorul nu există deja, se inserează un nou utilizator în baza de date users. Parola este hash-uită pentru securitate folosind password_hash(). Pentru a reprezenta statutul utilizatorului am folosit 0 pentru student, 1 pentru profesor și 2 pentru profesor coordonator.

```
$hashedPassword = password_hash($pass, PASSWORD_DEFAULT);
if ($statut == 'student' && $user_student) {
   // Inserarea studentului și asocierea cursurilor
   $sql = "INSERT INTO users (username, email, pass, statut) VALUES ('$username',
'$email', '$hashedPassword', 0)";
} else if ($statut == 'prof' && $user_prof) {
```

```
    $sql = "INSERT INTO users (username, email, pass, statut) VALUES ('$username',
'$email', '$hashedPassword', 1)";
} else if ($statut == 'prof_coord' && $user_prof) {
    $sql = "INSERT INTO users (username, email, pass, statut) VALUES ('$username',
'$email', '$hashedPassword', 2)";
}
```

Dacă utilizatorul este student, se inserează cerințele curente de proiect pentru proiectele lui în tabelul cerințe, pentru a asocia cursurile și cerințele specifice acestora cu studentul respectiv ca mai apoi acesta să își poată actualiza progresul. Tot la înregistrare se inserează și în tabelul cu note; pentru studentul care se înregistrează, la fiecare materie pe care o are se va pune 0, aceasta însemnând că nu a fost notat încă.

```
while ($course = mysqli_fetch_assoc($result_materie)) {  {//pt fiecare materie
  $id_materie = $course['id_materie'];
  $sql_nota="INSERT INTO note (id_student, id_materie, nota, terminat) VALUES
($id_student, $id_materie, 0, 0)";
  $db->query($sql_nota);

  $sql_task="SELECT id_task FROM taskuri WHERE id_materie=$id_materie";
  $result_task= $db->query($sql_task);
        while($task=mysqli_fetch_assoc($result_task)){  //adaugă fiecare task
        $id_task = $task['id_task'];
        // Inserează în tabelul cerințe
        $insert_cerinte_sql = "INSERT INTO cerinte (id_student, id_materie, id_task)
        VALUES ($id_student, $id_materie, $id_task)";
        $db->query($insert_cerinte_sql);
        }}
```

După inserarea cu succes a utilizatorului, acesta este redirecționat către pagina corespunzătoare statutului său (student, profesor, etc.) și sesiunea este actualizată pentru a reflecta autentificarea. Studenții vor fi redirecționați pe pagina index_logged.php, profesorii (inclusiv cei care sunt coordonatori) pe pagina index_prof_logged.php. La final, conexiunea la baza de date este închisă pentru a elibera resursele ($db->close(); ).

### C. Login

Funcționalitatea de autentificare (`login_action.php`) permite utilizatorilor să se conecteze la aplicația UniProject Manager. Procesul implică verificarea existenței utilizatorului în baza de date, validarea parolei și redirecționarea către paginile corespunzătoare rolului lor (student, profesor, secretară, admin).

Etapele principale ale procesului de autentificare:

1. Conectarea la baza de date și preluarea datelor de la utilizator:

Se include fișierul de configurare pentru conectarea la baza de date și se preiau datele de autentificare (email și parolă) trimise prin formularul de login.

```
include("config.php");

$email = $_POST['email'];

$parola = $_POST['password'];
```

2. Verificarea existenței utilizatorului în tabelul `users`:

Se caută utilizatorul în tabelul `users` pe baza email-ului. Dacă utilizatorul există, se verifică parola hash-uită folosind `password_verify()`. Dacă verificarea reușește, se inițializează sesiunea și se setează variabilele de sesiune astfel:

```
session_start();

$_SESSION['login'] = "user";

$_SESSION['email'] = $email;
```

După inițializarea sesiunii se redirecționează către paginile corespunzătoare, index_logged.php pentru studenți și index_prof_logged.php pentru profesori.

3. Verificarea existenței utilizatorului în tabelul `management`:

Dacă utilizatorul nu este găsit în tabelul `users`, se verifică dacă există în tabelul `management`. Se compară direct email-ul și parola. Dacă utilizatorul este secretară sau admin, se inițializează sesiunea și se setează variabilele de sesiune corespunzătoare, apoi se redirecționează către paginile lor respective, index_secretara_logged.php pentru secretară și admin.php pentru admin.

## D. Pagina Acasă

Pagina de acasă a aplicației UniProject Manager este concepută pentru a oferi utilizatorilor o prezentare clară și atractivă a funcționalităților principale ale platformei. Este implementată folosind HTML, CSS și PHP pentru a asigura o experiență de utilizare fluidă și modernă.

Structura și Componentele Paginii:

1. **Includerea Header-ului (Antetul):** Pagina începe prin includerea antetului (header.php), care conține meniul de navigare și logo-ul universității. Antetul este inclus folosind require_once pentru a menține consistența pe toate paginile site-ului și a facilita întreținerea codului.

```
<?php
  $currentPage = 'index';
  require_once "./header.php";
?>
```

2. **Stilizarea și aspectul general:** Stilizarea paginii este realizată folosind Bootstrap pentru componentele de bază și CSS personalizat pentru a adăuga un fundal gradient, culori și formatare specifică. Fonturile și dimensiunile sunt setate pentru a crea un aspect modern și profesional.
3. **Secțiunea de introducere:** Secțiunea de introducere (project-header) oferă o scurtă descriere a beneficiilor utilizării aplicației UniProject Manager. Textul este stilizat și evidențiat pe un fundal albastru închis pentru a atrage atenția utilizatorilor.
4. **Prezentarea funcționalităților cheie:** Pagina include o secțiune description unde sunt afișate imagini și descrieri detaliate ale funcționalităților cheie ale platformei. Imaginile sunt prezentate într-un format atractiv și accesibil, utilizând Bootstrap pentru grilă și stilizare personalizată pentru dimensiuni și spațiere.

```
<div class="description">
  <img src="interfata1.jpg" alt="Pagina Principală">
```

```
<p><strong>Profil Utilizator</strong>: Utilizatorii pot vizualiza detaliile profilului lor, inclusiv specializarea, numele, email-ul, anul de studiu și subgrupa...</p>

<!-- Continuarea conținutului -->

</div>
```

5. **Organizarea și dinamicitatea:** Pagina este concepută pentru a fi responsivă, adaptându-se la diferite dimensiuni ale ecranului. Aceasta asigură o experiență de utilizare optimă pe dispozitive mobile și desktop. Imaginile și textul sunt redimensionate și realiniate corespunzător.

```
<img style="width: 30%;" src="interfata3.jpg" alt="Profil Utilizator">
```

**E. Pagina Proiecte**

Pagina de proiecte (proiecte.php) din aplicația UniProject Manager este concepută pentru a afișa utilizatorilor o listă de materii (cursuri) disponibile, împreună cu detalii despre fiecare curs. Această pagină este dinamică și interactivă, utilizând PHP pentru gestionarea datelor și HTML/CSS/JavaScript pentru prezentarea și interacțiunea cu utilizatorul.

În secțiunea PHP, se include fișierul de configurare (config.php) și se execută interogările pentru a prelua materiile de  proiect (orar.id_tip='4') din cadrul departamentrului CIE (profesori.dep='0' OR profesori.dep='1') din baza de date. Pentru aceasta facem join-uri între tabelul orar și tabelele profesori, materi, nivele_seri și facem afișsarea în ordinea anului și a numelui de materie.

```
$sql2 = "SELECT*
    FROM orar
    CROSS JOIN profesori ON orar.id_profesor=profesori.id_profesor
    CROSS JOIN materi ON orar.id_materie=materi.id_materie
    CROSS JOIN nivele_seri ON orar.id_nivel=nivele_seri.id_ns
    WHERE (profesori.dep='0' OR profesori.dep='1') AND orar.id_tip='4'
    ORDER BY materi.id_an, nume";
```

Materia este afișată într-un container  cu clasa container, unde se iterează prin rezultatele interogării pentru a afișa fiecare materie ca un card. Un array $seen_materii este folosit pentru a evita duplicarea afișării materiei. Fiecare materie este afișată într-un card, iar clasa cardului este setată în funcție de anul (cardurile de anul 1 vor fi albastre, de anul 2 roșii, de anul 3 verzi, de anul 4 galbene). În figura Fig. 4.3 e prezentat un card cu un proiect de an 1.

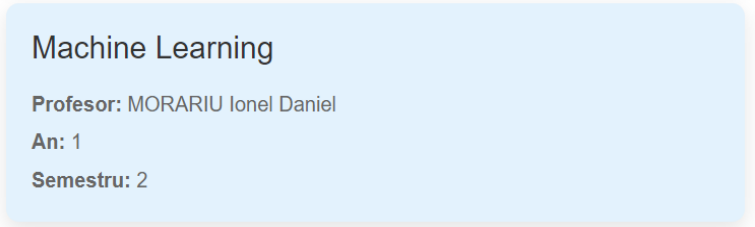



Fig. 4.3

La click pe un card un modal simplu este folosit pentru a informa utilizatorii că trebuie să se înregistreze pentru a accesa detalii suplimentare despre proiecte. Modalul este stilizat și poate fi închis prin apăsarea butonului "Închide" sau prin clic în afara modalului.

### F. Pagina Contact

Pagina de contact din aplicația UniProject Manager este concepută pentru a oferi utilizatorilor informații despre persoanele cheie din cadrul departamentului, cum ar fi directorul de departament, secretariatul și administratorul site-ului. Aceasta pagină include detalii de contact și funcționalități interactive pentru o experiență de utilizare plăcută și eficientă.

Informațiile de contact pentru directorul de departament, secretariatul și administratorul site-ului sunt afișate în carduri Bootstrap pentru o prezentare clară și organizată. Fiecare card conține numele, adresa, telefonul și email-ul persoanei de contact.

```
<div class="container mt-5">
 <div class="row">
  <div class="col-lg-4 mb-4">
   <div class="card border-0 shadow-sm h-100">
    <div class="card-body">
     <h5 class="card-title">Director Departament</h5> </div> </div>
</div>
```

```
</div>
```

Un script JavaScript simplu este folosit pentru a controla afișarea secțiunii "Despre Noi" atunci când utilizatorul face clic pe butonul "Despre Noi". Scriptul alternează vizibilitatea secțiunii între afișat și ascuns.

```
<script>
 function showDespreNoi() {
  var despreNoi = document.getElementById('despre-noi');
  if (despreNoi.style.display === 'none' || despreNoi.style.display === '') {
   despreNoi.style.display = 'block';
  } else {
   despreNoi.style.display = 'none';
  }}
</script>
```

Secțiunea "Despre Noi" este inițial ascunsă și conține informații detaliate despre platforma UniProject Manager, inclusiv scopul și funcționalitățile sale principale.

## 4.3. Student use-cases

Studentul este un utilizator autentificat care folosește aplicația pentru a gestiona proiectele academice proprii. Funcționalitățile disponibile includ:

- **Vizualizarea detaliilor proiectelor**: Studenții pot vedea detalii complete despre proiectele în care sunt implicați, inclusiv cerințe și termene limită în cazul temelor de licență.
- **Actualizarea progresului**: Studenții pot actualiza stadiul de realizare al proiectelor, oferind informații despre activitățile finalizate și cele în curs
- **Setarea ca terminat sau neterminat**: Pentru a putea ține evidența proiectelor terminate și neterminate și pentru ca profesorii să poată vizualiza și nota eficient studenții am implementat un buton prin care studentul poate seta starea curentă a proiectului.

- **Încărcarea arhivelor**: Studenții pot încărca fișiere și documentație asociate proiectelor lor și pot lăsa o descriere sugestivă pentru acestea.
- **Vizualizarea notelor și colaborarea cu profesorii**: Studenții pot vedea evaluările și comunica cu profesori pentru proiectele lor.

**Implementare**

**A. Pagina Acasă**

Pagina de acasă (index_logged.php) a studenților din aplicația UniProject Manager este concepută pentru a afișa informațiile personale ale studentului, precum și detalii despre cursuri și notele acestuia. Această pagină a fost implementată folosind PHP pentru a interacționa cu baza de date, HTML pentru structurarea paginii și CSS pentru stilizare. Pagina este accesibilă doar utilizatorilor autentificați, asigurându-se astfel protecția datelor personale.

La începutul paginii, se verifică dacă utilizatorul este autentificat prin sesiunea PHP. Dacă nu, utilizatorul este redirecționat către pagina de principală (index.php), variabila $x este folosită pentru a prelua emailul studentului ca mai apoi să-i afișăm informațiile personale în funcție de acesta.

```
session_start();
$x = $_SESSION['email'];
if (!(isset($_SESSION['login']))) {
   header("Location: index.php");
   exit();
}
```

După aceasta sunt executate mai multe interogări SQL pentru a prelua informațiile necesare despre student, inclusiv datele personale și detalii despre cursuri și profesori. Pentru a prelua materiile corespunzătoare studentului autentificat ne folosim de variabila `$specializare` în care punem denumirea specializării studentului și apoi o verificăm într-o structura if-else. În funcție de specializare sunt preluate materiile din semigrupele corespunzătoare (sunt verificate doar primele semigrupe din fiecare an deoarece și celelalte conțin aceleași materii) cu `nivele_seri.nume_ns`.

```
if ($specializare == 'Tehnologia Informatiei') {
   $sql3 = "SELECT*
```

```
FROM orar
CROSS JOIN profesori ON orar.id_profesor=profesori.id_profesor
CROSS JOIN materi ON orar.id_materie=materi.id_materie
CROSS JOIN nivele_seri ON orar.id_nivel=nivele_seri.id_ns
WHERE (profesori.dep='0' OR profesori.dep='1')
    AND orar.id_tip='4'
    AND (nivele_seri.nume_ns='214/1' OR nivele_seri.nume_ns='224/1' OR
nivele_seri.nume_ns='233/1' OR nivele_seri.nume_ns='243/1')
ORDER BY materi.id_an, nume";
$result_materii = $db->query($sql3);
}//codul continuă cu verificarea pentru celelalte specializări
```

Pagina Acasă include antetul comun pentru studenți, care este încărcat folosind require_once pentru a menține consistența în navigare pe toate paginile site-ului.

```
$currentPage = 'index_lgd';
require_once "./header_lgd.php";
```

Informațiile personale ale studentului sunt afișate într-un tabel în partea stângă cu stil personalizat prin aplicarea clasei . Tabelul este împărțit în două coloane, una pentru etichete și alta pentru valori.

În partea dreaptă a paginii, este afișat un tabel care conține informații despre cursuri, profesori și notele obținute de student. Datele sunt preluate și afișate într-un format structurat, folosind un tabel cu stilizat cu clasa `custom-table`. Folosindu-ne de $result_materii, pentru fiecare materie a studentului afișăm materia, profesorul, anul, semestrul și nota corespunzătoare.

**B. Pagina Proiecte**

Pagina proiecte_student.php din aplicația UniProject Manager este concepută pentru a afișa studenților informații despre materiile pe care le au, precum și detalii despre proiectele și profesorii aferenți fiecărei materii. Aceasta pagină utilizează PHP pentru a interacționa cu baza

de date și a extrage informațiile necesare, HTML pentru structurarea paginii și CSS pentru stilizare.

La începutul paginii, se verifică dacă utilizatorul este autentificat prin sesiunea PHP. Dacă nu, utilizatorul este redirecționat către pagina inițială (index.php).

Se include fișierul de configurare pentru conectarea la baza de date (config.php) și se execută mai multe interogări SQL pentru a prelua informațiile necesare despre student și cursurile acestuia. Se utilizează o interogare SQL pentru a selecta toate câmpurile din tabelele student și specializare unde emailul studentului corespunde cu $x, realizând un CROSS JOIN pentru a lega specializarea fiecărui student. Rezultatul interogării este preluat cu mysqli_fetch_assoc($result) și stocat într-un array asociativ.

```
include("config.php");

$sql = "SELECT * FROM student CROSS JOIN specializare ON student.specializare=specializare.id_specializare WHERE student.email = '$x'";

$result = $db->query($sql);

$developer = mysqli_fetch_assoc($result);

$specializare=$developer['denumire'];

//se folosește $developer pentru a prelua celelalte date dorite
```

În funcție de specializarea studentului, se execută o interogare SQL în care se specifică semigrupele corespunzătoare specializării pentru a prelua informațiile despre materii și profesori.

Pagina include antetul comun al site-ului, care este încărcat folosind require_once pentru a menține consistența în navigare pe toate paginile site-ului.

Materiile sunt afișate într-un container  cu clasa container. Pentru fiecare materie, se creează un card Bootstrap care conține detalii despre materie, profesor și an. Fiecare card este colorat în funcție de an.

Pe fiecare card Bootstrap se poate da click și se redirecționează automat utilizatorul către o pagină de detalii a proiectului (detalii_proiect_st.php), unde sunt afișate informații suplimentare despre proiectul specific.

Dacă studentul este în anul 4, se afișează un card suplimentar de culoare portocalie care conține informații despre licență, inclusiv datele de predare și prezentare.

```
if($an_curent==4){?>

        <div class="card licenta"
onclick="window.location.href='delalii_licenta_st.php?id_specializare=<?php echo
$id_specializare; ?>&id_student=<?php echo $id_student; ?>'">

            <h3>Licenta</h3>

            <p><strong>Data predarii:<?php echo  $data_predarii?></strong></p>

            <p><strong>Data prezentarii:<?php echo
$data_prezentarii?></strong></p>

        </div>
```

**C. Pagina Detalii Proiect**

Pagina detalii_proiect_st.php este o componentă esențială a aplicației UniProject Manager, concepută pentru a afișa detalii despre proiectele studenților. Aceasta poate fi accesată în două moduri, prin click în meniu pe "Detalii proiecte", astfel afișând cerințele de proiect pentru toate materiile studentului sau mai poate fi accesată prin click pe un card de materie din pagina de proiecte, astfel afișând informații despre materie, profesor, cerințele proiectului și permite încărcarea de fișiere și marcarea ca terminat a proiectului, oferind astfel o interfață completă pentru gestionarea proiectelor academice. Totodată, pagina va avea funcționalități diferite dacă cardul pe care s-a dat click este pentru o materie sau pentru licență.

Pagina de detalii proiect include următoarele funcționalități:

- Verificarea autentificării

La începutul paginii, se verifică dacă utilizatorul este autentificat prin sesiunea PHP. Dacă nu este autentificat, utilizatorul este redirecționat către pagina principală. Aceasta asigură că doar utilizatorii autorizați pot accesa detaliile proiectelor, protejând astfel informațiile sensibile ale studenților.

- Interogarea și prelucrarea datelor din baza de date

Pagina utilizează multiple interogări SQL pentru a prelua informațiile necesare despre student și proiectul acestuia. Mai întâi, se obțin detalii generale despre student, cum ar fi specializarea, anul curent și datele asociate licenței. În funcție de specializarea studentului, se execută o interogare specifică pentru a prelua informațiile despre materiile și profesorii aferenți.

- Afișarea detaliilor proiectului

Dacă un proiect specific este selectat, pagina afișează detalii precum numele materiei, numele profesorului, anul de studiu și semestrul. De asemenea, include un link pentru a trimite un email profesorului, facilitând astfel comunicarea între student și profesor. Informațiile sunt prezentate într-un mod organizat, utilizând carduri și tabele Bootstrap pentru a asigura o prezentare clară și estetică.

```
$sql = "SELECT* FROM orar

CROSS JOIN profesori ON orar.id_profesor=profesori.id_profesor

CROSS JOIN materi ON orar.id_materie=materi.id_materie

CROSS JOIN nivele_seri ON orar.id_nivel=nivele_seri.id_ns

WHERE (profesori.dep='0' OR profesori.dep='1')

   AND orar.id_tip='4'

   AND materi.id_materie = ? "

if ($stmt = $db->prepare($sql)) {

   $stmt->bind_param("i", $id_materie);

   $stmt->execute();

   $result = $stmt->get_result();

   $details = $result->fetch_assoc();
```

Această secțiune de cod PHP interoghează baza de date pentru a obține detalii despre o materie specifică folosind pregătirea unei instrucțiuni SQL (`prepared statement`). Dacă instrucțiunea pregătită este reușită, se leagă variabila `$id_materie` ca parametru, se execută instrucțiunea și se obține rezultatul. Rezultatul este apoi preluat cu `fetch_assoc()` pentru a obține detaliile materiei sub formă de array asociativ care va fi folosit pentru a le afișa. Instrucțiunile pregătite protejează împotriva injecțiilor SQL, o vulnerabilitate comună în aplicațiile web. Instrucțiunile pregătite folosesc parametri care sunt tratați separat de instrucțiunea SQL, astfel prevenind executarea codului malițios.

- Gestionarea cerințelor proiectului

Pagina permite studenților să vizualizeze și să actualizeze cerințele proiectului. Cerințele sunt afișate sub formă de checkbox-uri, permițând studenților să marcheze sarcinile îndeplinite. Aceste modificări sunt gestionate printr-un formular PHP care actualizează baza de date în funcție de statusul fiecărei sarcini, daca a fost bifată se va edita tabelul cerințe cu 1 la îndeplinire pentru studentul și materia respectivă, altfel se va edita cu 0 la îndeplinire. Aceasta oferă o modalitate simplă și eficientă pentru studenți de a-și urmări progresul.

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {

   $id_student = $_POST['id_student'];

   $id_materie = $_POST['id_materie'];

   if (isset($_POST['updateTasks'])) {

      $all_task_ids = fetchAllTaskIds($id_student, $id_materie);

      updateTasks($_POST['task'] ?? [], $all_task_ids);}
```

În acest cod se verifică dacă datele au fost trimise prin metoda POST, se preiau valorile id_student și id_materie și se verifică dacă formularul trimis conține câmpul `updateTasks` .Funcția fetchAllTaskIds returnează un array cu toate id-urile cerințelor care corespund studentului conectat și materii pe care a dat click. Apoi este apelată funcția `updateTasks` va edita câmpul îndeplinire din baza de date în funcție de ce a fost selectat de student.

- Funcționalitatea de încărcare a fișierelor

Studenții pot încărca fișiere legate de proiectele lor direct prin această pagină. Un formular permite selectarea și încărcarea fișierelor, împreună cu introducerea descrierii acestora. Fișierele încărcate sunt listate într-un tabel, oferind o referință rapidă și organizată a tuturor documentelor relevante pentru proiect.

Primul pas în procesul de încărcare a unui fișier este prezentarea unui formular utilizatorului. Formularul include un câmp pentru selectarea fișierului (<input type="file">), un câmp pentru descriere (<input type="text">), și două câmpuri ascunse pentru identificarea studentului și a materiei. Formularul utilizează metoda POST și atributul enctype="multipart/form-data" pentru a permite încărcarea fișierelor. La trimiterea formularului, datele sunt trimise către scriptul incarcaFisier_action.php pentru procesare. Atributul action al formularului specifică scriptul care va procesa datele, iar butonul de trimitere declanșează acest proces. Utilizarea funcției htmlspecialchars() în câmpurile ascunse asigură că datele sunt tratate corect și securizat.

Scriptul incarcaFisier_action.php gestionează procesul de încărcare a fișierului. Mai întâi, se verifică autentificarea utilizatorului și se preiau variabilele relevante din $_POST și $_FILES. Directorul de destinație pentru fișierele încărcate este definit, iar extensiile permise sunt specificate. Fișierul încărcat este verificat pentru erori și compatibilitate cu tipurile permise. Dacă verificările sunt trecute, fișierul este mutat din directorul temporar în directorul de destinație cu un nume unic generat prin md5.

```
if (in_array($fileExtension, $allowedfileExtensions)) {

  // Construiește noua cale a fișierului

  $newFileName = md5(time() . $fileName) . '.' . $fileExtension;

  $dest_path = $uploadURL . $newFileName;

  // Mută fișierul în directorul permanent

  if(move_uploaded_file($fileTmpPath, $uploadDirectory . $newFileName)) {

 //se inserează arhiva ;i datele despre ea în baza de date }}
```

Ulterior, scriptul conectează la baza de date și inseră detaliile fișierului în tabelul arhive, inclusiv id-ul studentului, id-ul materiei, data încărcării, calea fișierului, un 1 dacă proiectul este de curs și 0 dacă este de licență și descrierea. După inserare, utilizatorul este redirecționat fie la pagina detalii proiect, fie la pagina detalii licență, în funcție de contextul încărcării. Dacă apar erori în procesul de mutare a fișierului sau de încărcare, acestea sunt raportate utilizatorului.

- Marcare proiect ca terminat

Pagina include și funcționalitatea de a marca un proiect ca terminat. Printr-un buton dedicat, studenții pot seta statusul proiectului lor, ajutându-i să organizeze mai bine sarcinile finalizate și cele în desfășurare. După marcarea ca terminat, studenții au posibilitatea de a marca proiectul ca neterminat dacă este nevoie. Această acțiune este gestionată printr-un formular PHP care actualizează baza de date pentru a reflecta statusul curent al proiectului.

- Licența

Pagina `detalii_licenta_st.php` este concepută pentru a gestiona cererile și încărcările de fișiere pentru proiectele de licență ale studenților din anul 4. Pagina extrage datele necesare prin variabila $_GET, cum ar fi specializarea și ID-ul studentului, pentru a afișa detalii relevante despre proiectele de licență disponibile pentru specializarea specifică.

Primul pas al funcționalității paginii constă în gestionarea cererilor de teme de licență. Se efectuează o interogare în baza de date pentru a obține toate temele de licență disponibile pentru specializarea studentului, inclusiv detalii despre profesorii coordonatori. În cazul în care studentul a trimis deja o cerere pentru o temă de licență, statusul acesteia este verificat. Dacă cererea este în așteptare sau acceptată, detaliile cererii sunt afișate, inclusiv tema, profesorul coordonator și statusul cererii. Dacă cererea este acceptată, studentul are posibilitatea de a încărca fișierele asociate temei de licență prin intermediul unui formular.

Ultima parte a paginii gestionează încărcarea fișierelor pentru temele de licență acceptate. În cazul în care cererea a fost acceptată, studentul poate încărca fișierele printr-un formular special, iar aceste fișiere sunt stocate în baza de date și se specifică faptul că este pentru licență (se pune 1 la licență în tabelul arhive). Fișierele încărcate sunt afișate într-un tabel, incluzând data încărcării, numele fișierului și o descriere a acestuia. Dacă cererea nu a fost trimisă sau a fost refuzată, sunt afișate toate temele disponibile pentru specializarea studentului, împreună cu detaliile profesorului și locurile disponibile, permițând studentului să

trimită o nouă cerere. Aceasta asigură o gestionare completă a procesului de selectare și încărcare a proiectelor de licență, facilitând comunicarea și colaborarea între studenți și profesori.

Pagina detalii_proiect_st.php este un exemplu complex de utilizare a PHP pentru a interacționa cu baza de date și de a crea o interfață de utilizator interactivă și funcțională. Acest design asigură o experiență utilizator bine organizată și eficientă, permițând studenților să își gestioneze cu ușurință proiectele și să comunice eficient cu profesorii.

## 4.4. Profesor use-cases

Profesorul este un utilizator autentificat care are rolul de a supraveghea și evalua proiectele studenților. Funcționalitățile disponibile includ:

- **Vizualizarea detaliilor proiectelor**: Profesorii pot vedea toate informațiile relevante despre proiectele studenților din semigrupele pe care le coordonează.
- **Introducerea cerințelor pentru proiecte**: Profesorii pot adăuga sau modifica cerințele specifice pentru proiectele studenților.
- **Notarea studenților**: Profesorii pot evalua performanța studenților și pot oferi note și feedback.
- **Vizualizarea progresului**: Profesorii pot monitoriza progresul studenților și pot interveni atunci când este necesar.

Aceste funcționalități permit profesorilor să mențină un control riguros asupra activităților studenților și să asigure calitatea educațională.

**Implementare**

### A. Pagina Acasă

Pagina de acasă pentru profesori (index_prof_logged.php) în UniProject Manager oferă o interfață centralizată pentru profesori, în care aceștia pot vedea informații despre profilul lor și proiectele pe care le coordonează. La început, se inițiază o sesiune pentru a verifica dacă utilizatorul este autentificat. Dacă nu este autentificat, acesta este redirecționat către pagina index.php.

După autentificare, pagina începe cu un header, precum și includerea stilurilor CSS pentru Bootstrap și stiluri personalizate. Apoi, include un fișier PHP `header_prof_lgd.php`, care este un meniu de navigare comun pentru toate paginile din secțiunea profesorilor.

Pagina utilizează două interogări SQL principale. Prima selectează informațiile profesorului autentificat din tabelele `profesori_depcie` și `profesori` folosind adresa de email stocată în sesiune. A doua interogare selectează proiectele și materiile predate de profesorul respectiv, utilizând o combinație de tabele (`orar`, `profesori`, `materi`, și `nivele_seri).

Folosindu-ne de prima interogare, informațiile despre profilul profesorului sunt afișate într-un tabel mic plasat în partea stângă a paginii Fig. 4.4. Acest tabel include numele profesorului, emailul și dacă este coordonator sau nu.

| Username | constantinescu |
|---|---|
| Nume | CONSTANTINESCU Constantin |
| Email | constantin.constantinescu@ulbsibiu.ro |
| Coordonator | da |

Fig. 4.4

Folosindu-ne de a doua interogare, pe partea dreaptă a paginii, există un tabel care afișează detalii despre proiectele pe care le predă profesorul Fig. 4.5 Acest tabel include informații precum numele materiei, anul, semestrul, semigrupa și numărul de studenți participanți. Datele sunt afișate din interogarea `SELECT * FROM orar ... WHERE profesori.nume='$nume_prof'`, variabila $nume_prof conține numele profesorului autentificat pentru a putea afișa informațiile corespunzătoare lui.

| Proiecte | | | | |
|---|---|---|---|---|
| Materie | An | Semestru | Semigrupa | Nr. studenti |
| Programare WEB | 2 | 2 | 221/2 | 0 |
| Programare WEB | 2 | 2 | 222/1 | 0 |
| Programare WEB | 2 | 2 | 224/1 | 0 |
| Programare WEB | 2 | 2 | 221/1 | 2 |

Fig. 4.5

În concluzie, pagina de acasă pentru profesori în UniProject Manager este bine structurată și ușor de navigat, oferind informații esențiale despre activitatea și rolul profesorului. Fiecare secțiune este clar delimitată, iar utilizarea interogărilor SQL asigură că datele sunt actualizate și relevante pentru utilizator.

**B. Pagina Proiecte**

Pagina de proiecte pentru profesori (proiecte_profesor.php) din cadrul platformei Uni-Project Manager oferă profesorilor o interfață centralizată unde pot vizualiza informații despre proiectele pe care le predau și specializările pe care le coordonează la licență dacă este cazul. Această pagină este accesibilă doar profesorilor autentificați, iar verificarea autentificării este realizată la începutul scriptului prin inițierea unei sesiuni și verificarea existenței variabilei de sesiune 'login'.

După verificarea autentificării, pagina include un header specific pentru secțiunea profesorilor, importat prin fișierul 'header_prof_lgd.php'.

Pentru a obține informațiile necesare, facem mai multe interogări SQL. Prima interogare selectează informațiile profesorului din tabelele `profesori_depcie` și `profesori` folosind adresa de email stocată în sesiune pentru a putea prelua numele și id-ul profesorului autentificat.

A doua interogare selectează toate materiile predate de profesorul respectiv, utilizând tabelele `orar`, `profesori`, `materi` și `nivele_seri`. Materiile sunt afișate în carduri individuale care includ detalii despre materie, profesor, an, semestru și semigrupă Fig. 4.6 . Fiecare card are o clasă CSS specifică anului de studiu pentru a permite stilizarea diferențiată cu o culoare specifică pentru fiecare an de studiu.

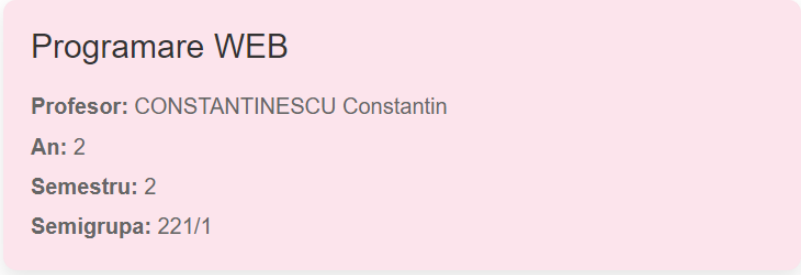



Fig. 4.6

Interfața este stilizată pentru a fi clară și ușor de navigat. Cardurile sunt bine delimitate și utilizatorii pot accesa detalii suplimentare despre fiecare proiect sau temă de licență printr-un simplu clic pe cardul respectiv. Acest lucru facilitează navigarea și permite profesorilor să acceseze rapid informațiile necesare.

**B. Pagina Detalii proiect**

Pagina "detalii_proiect_prof.php" din platforma UniProject Manager este concepută pentru a permite profesorilor să acceseze detalii specifice despre proiectele pe care le coordonează sau să vizualizeze studenții cu restanțe sau fără note. Această pagină poate fi accesată în două moduri: prin click pe un card al unui proiect specific sau printr-un link din meniu (Detalii proiect).

- Accesarea prin cardul unui proiect:

Când un profesor accesează pagina prin click pe un card al unui proiect, aceasta afişează detalii despre proiectul selectat. În primul rând, pagina include un header și o structură CSS pentru a asigura o prezentare vizuală coerentă. Codul PHP inițiază o sesiune și verifică autentificarea utilizatorului. Dacă utilizatorul nu este autentificat, acesta este redirecționat către pagina index.php.

După verificarea autentificării, pagina include datele profesorului și detaliile proiectului folosind mai multe interogări SQL. Prima interogare selectează informațiile profesorului, iar a doua interogare obține informațiile despre proiectul specific selectat. Detaliile proiectului includ numele materiei, numele profesorului, anul de studiu, semestrul, semigrupa, specializarea și numărul de studenți din semingrupă. Aceste informații sunt afișate într-un format bine structurat pentru a facilita vizualizarea.

În plus, pagina permite profesorilor să adauge, să editeze și să șteargă cerințele pentru proiectul specific. Cerințele sunt afișate într-un tabel, iar fiecare cerință poate fi modificată sau eliminată folosind formulare HTML. Profesorii pot, de asemenea, să adauge noi cerințe pentru proiectul respectiv. Aceasta permite o gestionare flexibilă și dinamică a cerințelor proiectului.

Fișierul "cerinteActions.php" este un script PHP care gestionează adăugarea, editarea și ștergerea cerințelor pentru proiectele studenților, în funcție de acțiunea specificată în URL. În

funcție de acțiunea specificată (`add`, `edit` sau `delete`), scriptul apelează funcțiile corespunzătoare pentru a efectua operațiile necesare pe baza de date. Scriptul utilizează instrucțiuni SQL pregătite (`prepared statements`) pentru a preveni atacurile de tip SQL injection și pentru a asigura manipularea corectă a datelor.

La început, scriptul inițializează câteva variabile utilizând datele primite prin metoda POST și GET. Funcția `addCerinta` inserează o nouă cerință în tabelul `taskuri`, iar apoi folosește `insertCerinteStudent` pentru a asocia această cerință cu studenții corespunzători. Funcția `insertCerinteStudent` inserează cerințele pentru fiecare student în funcție de materia selectată, asigurându-se că toate cerințele sunt corect alocate studenților relevanți și la materia corespunzătoare.

Funcția `deleteCerinta` elimină o cerință și asociațiile acesteia cu studenții prin apelarea funcției `deleteCerinteStudent` care șterge cerințele asociate task-ului primit ca parametru din tabelul `cerinte`.

```
function deleteCerinta($id_task) {

   include("config.php");

   $sql = "DELETE FROM taskuri WHERE id_task = ?";

   if ($stmt = $db->prepare($sql)) {

      $stmt->bind_param("i", $id_task);

      $stmt->execute();

      $stmt->close();}

   deleteCeriteStudent($id_task);

   $db->close();}
```

Funcția `editCerinta` actualizează o cerință existentă, prin editarea taskului căruia îi corespunde id-ul trimis ca parametru.

La final, scriptul redirecționează utilizatorul înapoi la pagina de detalii a proiectului (`detalii_proiect_prof.php`), trecând parametrii necesari prin URL, asigurându-se astfel că utilizatorul poate vedea modificările făcute fără a fi necesar să reîncarce manual pagina. Astfel, "cerinteActions.php" automatizează gestionarea cerințelor proiectelor studenților într-un mod eficient și securizat.

Pe lângă gestionarea cerințelor, pagina afișează și o listă a studenților din semigrupa selectată. Fiecare student este listat împreună cu numele, prenumele, emailul, statusul proiectului (terminat sau neterminat), nota (dacă există) și acțiuni posibile. Profesorii pot nota studenții și pot vizualiza progresul studenților și arhivele încărcate de aceștia pentru proiectul specific. Progresul include detalii despre cerințele îndeplinite de student, iar arhivele includ fișierele încărcate de studenți, împreună cu descrierea acestora. Pentru a realiza interogările corespunzătoare progresului sau arhivelor, se verifică metoda de cerere și numele butonului apăsat.

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {

  $id_student = $_POST['id_student'];

  $id_materie = $_POST['id_materie'];

  if (isset($_POST['progres'])) {

$sql_progres = "SELECT * FROM cerinte

                LEFT JOIN taskuri ON cerinte.id_task = taskuri.id_task

                WHERE cerinte.id_student = ? AND cerinte.id_materie = ?";

        //...

} else if(isset($_POST['arhive'])) {

$sql_arhive = "SELECT * FROM arhive WHERE id_student = ? AND id_materie = ? AND
licenta = 0 ORDER BY data_incarcarii DESC";

//...}
```

Fișierul `note_action.php` gestionează adăugarea și actualizarea notelor pentru studenți. La început, inițiază o sesiune și include configurația bazei de date. Scriptul verifică dacă datele necesare sunt trimise prin metoda POST. Dacă datele sunt disponibile, se verifică dacă există deja o notă pentru studentul și materia specificate. Dacă există, se actualizează nota curentă; dacă nu, se inserează o nouă înregistrare în tabelul `note`. După actualizare sau inserare, utilizatorul este redirecționat înapoi la pagina de detalii a proiectului. În caz de eroare, afișează un mesaj de eroare.

- Accesarea prin meniu:

Când pagina este accesată printr-un link din meniu, aceasta afișează studenții cu restanțe sau fără note pentru toate proiectele corespunzătoare profesorului respectiv. Pagina include un tabel detaliat care listează fiecare student care nu a obținut o notă de trecere sau nu a primit încă o notă pentru proiectele corespunzătoare profesorului. Tabelul include informații precum numele și prenumele studentului, emailul, semigrupa, nota și statusul proiectului (terminat sau neterminat).

```
$sql_restante = "SELECT s.id_student, s.nume, s.prenume, s.email, s.grupa, n.nota,
n.terminat FROM note n
        LEFT JOIN student s ON n.id_student = s.id_student
        WHERE n.id_materie = $id_mat AND ((n.nota < 5 AND n.nota > 0) OR n.nota = 0)";
$result_restante = $db->query($sql_restante);
```

Pentru fiecare student listat, profesorii au opțiunea de a vizualiza arhivele încărcate și progresul proiectului. De asemenea, profesorii pot adăuga sau modifica nota studentului direct din această pagină. Aceasta oferă profesorilor o modalitate eficientă de a gestiona studenții cu restanțe și de a se asigura că toți studenții primesc evaluările corespunzătoare pentru proiectele lor.

## 4.5. Secretara use-cases

Secretara este un utilizator autentificat cu rol administrativ în cadrul aplicației. Funcțiile sale includ:

- **Setarea datelor de prezentare a licențelor**: Secretara poate stabili și actualiza datele la care vor avea loc prezentările de licență.

- **Vizualizarea listelor cu candidați la licență**: Secretara poate accesa listele complete ale studenților care urmează să susțină proiectele de licență.
- **Ștergerea, editarea și inserarea de noi studenți și profesori**: Pentru a păstra o bună organizare a departamentrului, interfața secretarei conține butoane prin care sunt implementate aceste funcționalități.

Aceste funcționalități sunt cruciale pentru asigurarea unui flux administrativ eficient și pentru coordonarea evenimentelor academice importante.

**Implementare**

**A. Pagina Acasă**

Pagina de acasă pentru secretară (index_secretara_logged.php) în aplicația UniProject Manager este concepută pentru a oferi funcționalități esențiale legate de gestionarea studenților și profesorilor din cadrul universității. Aceasta include inserarea, editarea și ștergerea înregistrărilor din tabelele corespunzătoare, precum și sortarea datelor pentru o vizualizare mai eficientă.

Pagina este împărțită în două secțiuni principale: gestionarea studenților și gestionarea profesorilor. Fiecare secțiune are un tabel ce listează înregistrările curente și formulare pentru adăugarea, editarea sau ștergerea acestora. Această abordare modulară permite secretarei să acceseze și să administreze informațiile necesare într-un mod organizat și eficient.

La începutul scriptului PHP, fișierul de configurare al bazei de date (`config.php`) este inclus pentru a stabili conexiunea cu baza de date. Apoi, se gestionează acțiunile venite din formularele de pe pagină. Acestea includ inserarea, editarea și ștergerea înregistrărilor pentru studenți și profesori. Manipularea datelor se face pe baza valorii `action` transmisă prin metoda POST, ceea ce determină ce tip de operațiune să fie efectuată.

```
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    if (isset($_POST['action'])) {
        if ($_POST['action'] == 'insert_student') ...
```

În secțiunea de gestionare a studenților, reprezentată în figura Fig. 4.7, se afișează un tabel cu toți studenții înregistrați, ordonați după un criteriu selectat. Secretara poate alege să sorteze studenții după ID, nume, specializare, an sau grupă. Fiecare rând al tabelului reprezintă

un student și conține câmpuri editabile pentru toate informațiile relevante (nume, prenume, specializare, an, email, grupă). Secretara poate modifica informațiile direct în tabel și apoi poate apăsa butonul "Editează" pentru a salva modificările.

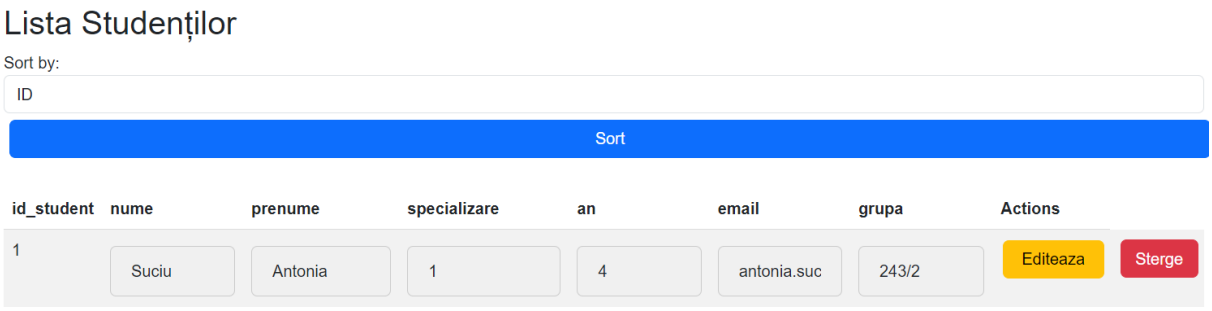



Fig. 4.7

Există și opțiunea de a șterge un student care este o funcționalitate necesară secretarei pentru a elimina studenții care s-au transferat la altă facultate, care au renunțat sau care nu au trecut anul. Ștergerea de studenți presupune confirmarea utilizatorului pentru a preveni ștergerile accidentale. Ștergerea unui student implică și ștergerea contului de utilizator asociat acestuia din tabelul `users`. De asemenea, există un formular la sfârșitul tabelului pentru a adăuga un nou student, completând informațiile necesare și apăsând butonul "Inserează" pentru ca secretara să poată introduce studenți care s-au transferat la o anumită specializare sau studenții noi de anul unu după ce au fost admiși.

Secțiunea de gestionare a profesorilor, reprezentată în figura Fig. 4.8, funcționează într-un mod similar. Aici, se afișează profesorii din departamentele 0 și 1(Calculatoare și inginerie electrică). Tabelul listează ID-ul profesorului, numele, departamentul, emailul și dacă este coordonator. Informațiile profesorilor pot fi editate direct în tabel, iar modificările sunt salvate prin apăsarea butonului "Editează".

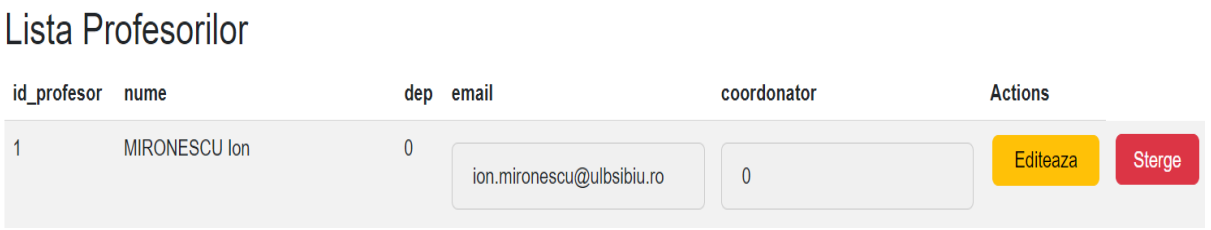



Fig. 4.8

Dacă un profesor nu există în tabela `profesori_depcie`, secretara poate introduce un nou profesor completând câmpurile corespunzătoare și apăsând butonul "Inserează". Similar cu

studenții, profesorii pot fi șterși din sistem, ceea ce implică și ștergerea contului de utilizator asociat.

Formularele sunt integrate direct în rândurile tabelului pentru a permite modificări rapide și eficiente, eliminând necesitatea unor pagini sau ferestre modale suplimentare.

Pagina de acasă pentru secretară este un instrument esențial pentru gestionarea eficientă a studenților și profesorilor din cadrul universității. Funcționalitățile de inserare, editare și ștergere sunt integrate într-o interfață intuitivă și prietenoasă, facilitând astfel munca secretarei. Această pagină centralizează toate acțiunile necesare pentru administrarea datelor academice, asigurând un flux de lucru fluid și eficient.

**B. Pagina Licenta**

Pagina de licență (licenta_secretariat.php) pentru secretariat din cadrul aplicației UniProject Manager este un instrument esențial pentru administrarea și gestionarea studenților care sunt înscriși pentru lucrarea de licență, a profesorilor coordonatori și a datelor de prezentare a licenței. Structura paginii este împărțită în trei secțiuni principale: studenți înscriși și temele lor de licență, specializări și data prezentării licenței, și profesori coordonatori.

Pagina utilizează framework-ul Bootstrap pentru stilizarea elementelor HTML și este realizată pentru a fi responsivă și ușor de utilizat. Codul PHP este integrat pentru a manipula baza de date și pentru a gestiona cererile HTTP POST. Interfața este organizată astfel încât să permită secretarei să vizualizeze, editeze și șteargă înregistrările necesare într-un mod intuitiv și eficient.

Prima secțiune a paginii afișează o listă cu studenții care s-au la licență și au fost acceptați. Se face o interogare a tabelului cereri_teme pentru a prelua detaliile relevante din baza de date: numele și prenumele studentului, email-ul, specializarea, tema de licență și numele profesorului coordonator. Datele sunt afișate într-un tabel pentru o vizualizare clară. Email-ul studentului este afișat ca un link de tip `mailto` pentru a facilita contactarea rapidă.

A doua secțiune a paginii se concentrează pe gestionarea specializărilor și a datelor de prezentare a licenței. Interogarea SQL preia toate specializările din baza de date împreună cu datele de prezentare a licenței pentru fiecare specializare. Datele sunt afișate într-un tabel cu coloane pentru specializare, data prezentării licenței și acțiuni unde avem buton de editare a datei de prezentare a licenței.

Ultima secțiune a paginii se ocupă de gestionarea profesorilor coordonatori. Inițial, se preiau toți profesorii coordonatori și specializările pentru care sunt responsabili din baza de date și sunt afișați într-un tabel. Tabelul include coloane pentru numele profesorului, specializare și acțiuni.

Pentru fiecare profesor coordonator, există opțiuni de ștergere a înregistrării respective. Formularul pentru ștergere este încorporat în rândul tabelului și include un buton "Șterge" care solicită confirmarea utilizatorului înainte de a elimina profesorul din baza de date. Acest lucru se realizează printr-o interogare SQL de tip DELETE.

Pe lângă ștergere, există și posibilitatea de a adăuga noi profesori coordonatori la o specializare. Secretara poate introduce ID-ul specializării în câmpul de text și poate trimite formularul pentru a adăuga un profesor coordonator cu o nouă specializare la care coordonează. La trimiterea formularului, datele sunt inserate în baza de date folosind o interogare SQL de tip INSERT.

Formularele din pagina sunt gestionate prin intermediul cererilor HTTP POST. În funcție de valoarea `action` transmisă, se determină ce acțiune trebuie efectuată:

1. Editarea datei de predare a licenței pentru fiecare specializare

2. Ștergerea profesorilor coordonatori: Ștergerea unei înregistrări din tabelul `locuri` care reprezintă asocierea unui profesor coordonator cu o specializare.

3. Inserarea profesorilor coordonatori: Adăugarea unui nou profesor coordonator pentru o specializare specificată.

Pagina de licență pentru secretariat este un instrument important pentru administrarea procesului de licență în cadrul universității. Aceasta oferă o interfață clară și funcționalități esențiale pentru gestionarea studenților, profesorilor și a datelor critice asociate procesului de licență. Interfața prietenoasă și bine organizată, împreună cu funcționalitățile de editare și ștergere integrate, asigură o administrare eficientă și precisă a datelor.

## 4.6. Coordonator de licență use-cases

Coordonatorul de licență este un utilizator autentificat responsabil de gestionarea proiectelor de licență. Funcționalitățile disponibile includ:

- **Setarea locurilor disponibile**: Coordonatorul poate defini numărul de studenți care pot fi înscriși la fiecare specializare pentru susținerea licenței.
- **Setarea temelor**: Coordonatorul poate adăuga, modifica sau șterge temele de licență disponibile pentru studenți.
- **Setarea datei de predare**: Coordonatorul poate stabili și actualiza termenele limită pentru predarea proiectelor de licență.
- **Gestionare cereri**: Coordonatorul poate respinge sau poate accepta cererile primite de la studenți
- **Vizualizează proiecte**: Coordonatorul poate vizualiza arhivele cu lucrările de licență încărcate de studenții pe care i-a acceptat

Aceste funcționalități permit coordonatorului să gestioneze eficient aspectele administrative și organizatorice ale proiectelor de licență.

**Implementare**

### A. Pagina Acasă

Pagina de Acasă este aceeași ca la profesorii care nu sunt și coordonatori, conține tabel cu informații despre profesor, unul cu proiectele predate și în plus un tabel cu informații despre licență.

Dacă profesorul este și coordonator de licență, se execută o altă interogare SQL pentru a selecta locurile disponibile și ocupate pentru teme de licență, afișate într-un alt tabel Fig. 4.9 Acest tabel include specializarea, locurile disponibile și locurile ocupate, oferind o privire de ansamblu asupra gestionării temelor de licență.

| Licenta | | |
|---|---|---|
| **Specializare** | **Locuri Disponibile** | **Locuri Ocupate** |
| Tehnologia Informatiei | 8 | 3 |
| ISM | 10 | 0 |

Fig 4.9

**B. Pagina Proiecte**

Dacă profesorul este și coordonator de licență, se execută o interogare SQL pentru a selecta locurile disponibile și ocupate pentru temele de licență pe care le coordonează. Aceste informații sunt afișate în carduri separate, care includ numele profesorului  specializarea, locurile disponibile și locurile ocupate Fig. 4.10

```
$sql4="SELECT * FROM locuri

CROSS JOIN specializare ON specializare.id_specializare=locuri.id_specializare

WHERE id_profesor_depcie=$id_profesor_depcie";

$result_licenta = $db->query($sql4);
```

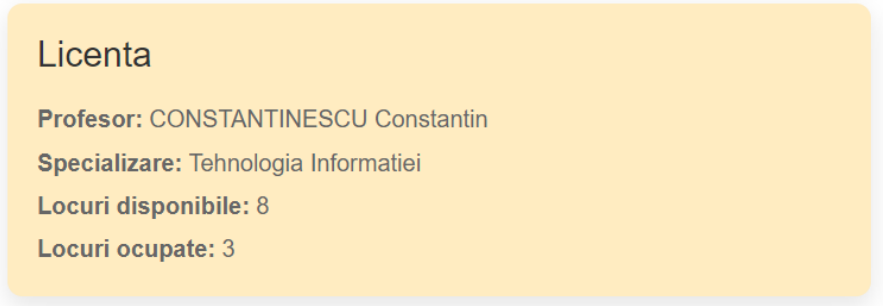



Fig. 4.10

La click pe acest card, profesorul e redirecționat către pagina de Detalii proiect (detalii_licenta_prof.php) unde sunt implementate funcționalități care ajută la gestionarea studenților și a temelor de licență corespunzătoare specializării.

**C. Pagina Detalii proiect**

Pagina `detalii_licenta_prof.php` este creată pentru a oferi coordonatorilor de licență o interfață completă pentru gestionarea studenților și temelor de licență. Aceasta este accesibilă prin click pe un card de licență din pagina de proiecte a aplicației. Funcționalitățile sale includ editarea locurilor disponibile, a datei de predare a licenței, introducerea de teme de licență, gestionarea cererilor de teme, și vizualizarea arhivelor încărcate de studenți.

Imediat ce pagina este accesată, se efectuează mai multe interogări SQL pentru a extrage informațiile relevante despre locurile disponibile și cererile de teme. Aceste date sunt apoi afișate într-un format tabular, folosind HTML și Bootstrap pentru o prezentare clară și ușor de utilizat. Acest lucru permite coordonatorilor să vadă rapid starea actuală a locurilor disponibile și detalii despre cererile studenților.

Coordonatorii de licență au posibilitatea de a modifica numărul de locuri disponibile pentru studenți și data limită pentru predarea lucrărilor de licență. Aceste funcționalități sunt implementate prin intermediul formularelor POST. Odată ce datele sunt trimise, interogările SQL actualizează baza de date pentru a reflecta noile informații. Aceasta asigură că toate modificările sunt realizate în timp real și informațiile afișate sunt mereu actualizate.

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['editLocuri']))
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['editDataPredare']))
```

Coordonatorii au, de asemenea, posibilitatea de a adăuga, edita sau șterge teme de licență disponibile pentru studenți. Aceste operațiuni implementate în fișierul temeActions.php cu un switch în funcție de acțiunea trimisă prin formular (add, edit, delete), fiecare temă având opțiunea de a fi modificată sau eliminată. Adăugarea de noi teme se face prin completarea unui câmp de text și trimiterea formularului corespunzător, iar modificările sunt salvate în baza de date în tabelul de teme_licenta.

Un aspect important al paginii este gestionarea cererilor de teme de licență din partea studenților realizată prin verificarea metodei și al numelui butonului. Coordonatorii pot accepta sau refuza aceste cereri folosind formulare dedicate; răspunsul coordonatorului este salvat în tabelul cereri_teme . La acceptarea unei cereri, locurile disponibile se ajustează automat, scăzând cu unul, iar cele ocupate crescând cu unul. În caz de refuz, starea cererii este actualizată corespunzător. Aceste acțiuni sunt esențiale pentru menținerea unui echilibru între numărul de teme disponibile și cererea studenților.

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['accepta'])) {...}
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['refuza'])) {...}
```

O altă funcționalitate importantă este vizualizarea arhivelor încărcate de studenți pentru lucrările de licență. Coordonatorii pot accesa aceste arhive prin formulare care declanșează interogări SQL pentru a extrage și afișa fișierele relevante. Acest lucru permite coordonatorilor să verifice progresul studenților și să se asigure că toate documentele necesare sunt în ordine.

```
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['arhive'])) {...}
```

Pagina `detalii_licenta_prof.php` este un instrument esențial pentru coordonatorii de licență, oferind o gamă largă de funcționalități pentru gestionarea eficientă a studenților și temelor de licență. De la modificarea locurilor disponibile și a datelor de predare, până la gestionarea cererilor și vizualizarea arhivelor, această pagină asigură o administrare centralizată și eficientă a procesului de licență.

## 4.7. Admin use-cases

Administratorul este utilizatorul autentificat cu privilegii specifice de administrare a sistemului. Funcționalitățile disponibile includ:

- **Actualizarea anului academic**: Administratorul poate seta și actualiza anul academic pentru întregul sistem, asigurându-se că toate datele și termenele sunt corect aliniate.
- **Schimbarea secretarei**: Administratorul are posibilitatea de a schimba persoana desemnată ca secretară, actualizând astfel informațiile de contact și responsabilitățile aferente.
- **Logare și gestionare cont**: Administratorul se poate autentifica în sistem pentru a efectua toate aceste activități.

Aceste funcționalități permit administratorului să asigure buna funcționare a aplicației și să gestioneze resursele și utilizatorii în mod eficient.

**Implementare**

Pagina de admin (`admin.php`) a platformei UniProject Manager este destinată gestionării centralizate a informațiilor despre studenți și secretariat. Rolul acestei pagini este de a permite administratorului să editeze detaliile studenților, să avanseze studenții în anul următor și să gestioneze informațiile despre secretariat.

Pagina începe cu includerea fișierului de configurație pentru baza de date (`config.php`) și definirea variabilei `$currentPage` pentru a indica că aceasta este pagina curentă a adminului. De asemenea, sunt gestionate cererile de formă trimise prin metoda POST pentru a actualiza detaliile studenților și ale secretariatului.

În funcție de acțiunea specificată în cererea POST, sunt executate diferite operațiuni:

- Editare student: Se preiau datele din formular și se actualizează în tabelul `student` din baza de date.

- Avansare an: Se preiau toți studenții, se extrage anul curent din grupa fiecărui student și, dacă anul este mai mic de 4 în cazul studenților la programul de licență, sau mai mic decât anul 2 în cazul celor de la master, se incrementează și se actualizează grupa studentului în baza de date.

- Editare detalii secretariat: Se preiau datele din formular și se actualizează în tabelul `management` pentru a modifica detaliile secretariatului.

Pagina admin folosește mai multe interogări SQL pentru a prelua datele necesare:

- Studenți: O interogare SQL complexă care face join între tabelele `student` și `specializare` pentru a aduce detaliile complete ale fiecărui student.

```
$sort_field = isset($_POST['sort']) ? $_POST['sort'] : 'grupa';

$sql_students = "SELECT * FROM student LEFT JOIN specializare ON specializare.id_specializare=student.specializare ORDER BY $sort_field";

$result_students = $db->query($sql_students);
```

- Detalii secretariat: O interogare pentru a prelua detaliile secretariatului din tabelul `management`.

```
$sql_secretary = "SELECT * FROM management WHERE rol='secretara' LIMIT 1";

$result_secretary = $db->query($sql_secretary);

$secretary = $result_secretary->fetch_assoc();
```

Pagina include o interfață de utilizator prietenoasă realizată cu ajutorul Bootstrap. Interfața are următoarele secțiuni:

- Meniu de navigare: Include link-uri către pagina de acasă a adminului și către opțiunea de logout. De asemenea, afișează logo-ul și titlul aplicației.

- Lista studenților: O tabelă detaliată care afișează informațiile despre studenți. Fiecare rând al tabelei conține un formular pentru editarea detaliilor fiecărui student. Există și un buton pentru avansarea anului pentru toți studenții.

- Detalii secretariat: Un formular simplu pentru editarea numelui și prenumelui persoanei din secretariat.

Funcționalități cheie implementate pentru admin:

- Avansare an: Printr-un simplu click pe butonul „Schimba Anul”, toți studenții sunt avansați în anul următor, iar grupele lor sunt actualizate corespunzător (exemplu: 221 va fi 231) .

- Editare și actualizare detalii studenți: Administratorul poate edita detaliile fiecărui student direct din tabelă. După modificare, datele sunt trimise prin formularul POST și actualizate în baza de date. Această funcționalitate ajută la editarea semigrupelor studenților în cazul în care după schimbarea anului rămân semigrupe cu prea puțini studenți.

- Editare detalii secretariat: Administratorul poate modifica detaliile secretariatului, de exemplu, în cazul în care secretara este înlocuită, asigurându-se că informațiile sunt actualizate și corecte.

Pagina de admin din aplicația UniProject Manager este esențială pentru gestionarea eficientă a studenților și a informațiilor secretariatului. Cu ajutorul unei interfețe intuitive și a unor funcționalități esențiale precum editarea datelor și avansarea anului, aceasta asigură o administrare ușoară și eficientă a departamentului de Calculatoare și Inginerie Electrică.

# 5. Concluzie

UniProject Manager este o platformă web fiabilă și performantă, creată pentru a îmbunătăți gestionarea proiectelor academice, a temelor de licență și a interacțiunilor dintre studenți, profesori și personalul administrativ din cadrul departamentului de Calculatoare și inginerie Electrică al facultății de inginerie ULBS Sibiu. Acest site a fost dezvoltat pentru a oferi soluții eficiente la nevoile specifice ale fiecărui utilizator, asigurându-se că toate procesele academice sunt ușor de gestionat și de urmărit.

UniProject Manager permite studenților să își gestioneze proiectele și temele de licență într-un mod organizat și centralizat. Aceștia pot încărca arhive, adăuga descrieri și vizualiza progresul lor în timp real. Profesorii, pe de altă parte, au acces la toate proiectele studenților, pot adăuga cerințe noi, pot evalua progresul și pot oferi feedback eficient.

Profesorii pot gestiona cu ușurință proiectele predate și studenții participanți. Platforma permite adăugarea, editarea și ștergerea temelor de licență, precum și gestionarea cererilor primite de la studenți. Secretariatul are la dispoziție funcționalități avansate pentru a gestiona datele studenților și profesorilor, asigurându-se că toate informațiile sunt actualizate și corecte.

Site-ul include funcționalități automate pentru a ușura munca administrativă, cum ar fi actualizarea anului de studiu al studenților și gestionarea locurilor disponibile pentru temele de licență. Aceste funcții reduc erorile umane și economisesc timp pentru personalul administrativ.

Dezvoltarea ulterioară a platformei UniProject Manager va include următoarele îmbunătățiri:

1. Observații pentru studenți:

- Profesorii vor putea adăuga observații individuale pentru fiecare student la fiecare cerință de proiect. Acest lucru va permite feedback mai detaliat și personalizat, contribuind la îmbunătățirea performanțelor studenților și la o mai bună înțelegere a cerințelor academice.

2. Data limită de predare:

- Profesorii vor putea seta date limită de predare pentru proiectele studenților. Aceasta va ajuta la gestionarea mai eficientă a timpului și va asigura respectarea termenelor de către studenți, facilitând astfel planificarea și evaluarea proiectelor.

3. Colaborare între studenți (Echipe):

- Studenții vor putea colabora între ei prin formarea de echipe pentru proiectele lor. Această funcționalitate va încuraja munca în echipă, va îmbunătăți abilitățile de colaborare și va permite studenților să abordeze proiectele mai complex.

4. Împărțirea automată a studenților în semigrupe:

- La actualizarea anului de studiu, platforma va permite împărțirea automată a studenților în semigrupe de dimensiuni corespunzătoare. Aceasta va elimina necesitatea editării manuale a semigrupelor de către admin, asigurând o distribuție echitabilă a studenților și evitând semigrupelor cu un număr prea mic de studenți.

Aceste îmbunătățiri sunt menite să optimizeze și mai mult funcționalitatea UniProject Manager, să sprijine o gestionare mai eficientă a proiectelor academice și să îmbunătățească experiența utilizatorilor.

UniProject Manager reprezintă un exemplu excelent de utilizare a tehnologiilor web moderne pentru a rezolva problemele reale din mediul academic. Prin integrarea eficientă a PHP, HTML, CSS și JavaScript, împreună cu instrumente precum VS Code, Git și XAMPP, am realizat o platformă care nu doar că simplifică gestionarea proiectelor și lucrărilor de licență, dar și îmbunătățește semnificativ experiența utilizatorilor. Platforma demonstrează cum tehnologia poate fi utilizată pentru a crea soluții eficiente și intuitive, sprijinind astfel procesele educaționale și administrative într-o manieră coerentă și eficientă.

# 6. Webliografie

[WWW01] CEITI (Centrul de Excelență în Informatică și Tehnologii Informaționale)- PHP/MySQL, preluat de pe https://web.ceiti.md/files/MySQLsiPHP.pdf

[WWW02] Code Institute- A Comprehensive Guide to PHP Programming: What You Need to Know, preluat de pe https://codeinstitute.net/global/blog/what-is-php-programming/

[WWW03] The IoT Academy- What is XAMPP Server, preluat de pe https://www.theiotacademy.co/blog/xampp-server/

[WWW04] W3Schools- HTML Tutorial, preluat de pe https://www.w3schools.com/html/

[WWW05] W3Schools- CSS Tutorial, preluat de pe https://www.w3schools.com/css/default.asp

[WWW06] Wikipedia - JavaScript, preluat de pe https://en.wikipedia.org/wiki/JavaScript