

# PathFinder - A Learning Management System

Suhas Kollur

*dept. of Electrical and Computer Engineering  
Rutgers University-New Brunswick  
New Brunswick, New Jersey  
sk2870@scarletmail.rutgers.edu*

Parth Kharkar

*dept. of Electrical and Computer Engineering  
Rutgers University-New Brunswick  
New Brunswick, New Jersey  
pk674@scarletmail.rutgers.edu*

**Abstract**—In response to the evolving landscape of higher education, this paper presents the design and implementation of a sophisticated educational management system tailored to the needs of academic institutions. By harnessing contemporary web technologies and adhering to best practices in software engineering, the system offers a comprehensive suite of features catering to the multifaceted requirements of professors, students, and administrative staff. Through meticulous analysis, iterative development cycles, and rigorous testing, the platform strives to optimize administrative workflows, facilitate seamless communication, and cultivate collaborative learning environments. This endeavor represents a pivotal advancement in educational technology, underscoring our commitment to innovation and excellence in academia.

**Index Terms**—component, formatting, style, styling, insert

## I. INTRODUCTION

In the realm of higher education, the effective management of academic resources and administrative processes is paramount to the success of institutions and the cultivation of a conducive learning environment. To address the diverse and evolving needs of academic stakeholders, from professors to students and administrative staff, the development of a robust educational management system is imperative. This paper delves into the conceptualization, design, and implementation of such a system, leveraging contemporary web technologies and software engineering principles to create a comprehensive solution tailored to the intricacies of academia.

At its core, the system aims to streamline administrative workflows, optimize resource allocation, and enhance communication channels within academic institutions. Grounded in the principles of user-centered design, the platform offers a seamless user experience characterized by intuitive interfaces, efficient navigation, and accessibility across devices. By harnessing the power of React.js for frontend development and Node.js for backend services, the system achieves a balance of performance, scalability, and maintainability, ensuring its viability in diverse educational contexts.

The genesis of the project stems from a thorough analysis of the pain points and inefficiencies prevalent in traditional academic management systems. Through extensive stakeholder consultations and iterative prototyping, the system's architecture emerged, emphasizing modularity, extensibility, and interoperability. From the professor's perspective, functionalities encompassing course management, profile setup, and announcement dissemination empower educators to focus on

delivering quality instruction while leveraging technology to augment administrative tasks.

Moreover, the system caters to student needs by providing seamless access to course materials, assignment submissions, and communication with peers and professors. Administrative features, including course creation, student enrollment management, and announcement dissemination, empower administrative staff to orchestrate academic operations efficiently. Through meticulous attention to security, data privacy, and compliance with regulatory standards, the system ensures the integrity and confidentiality of sensitive information.

Furthermore, the system's architecture facilitates extensibility and interoperability, enabling integration with existing institutional systems and future enhancements. By adhering to industry best practices in software development, including version control, continuous integration, and automated testing, the system embodies a commitment to reliability, scalability, and maintainability.

In summary, the development of this educational management system represents a significant stride forward in the realm of educational technology, poised to revolutionize the way academic institutions operate and engage with stakeholders. Through a synthesis of cutting-edge technologies, user-centric design principles, and rigorous software engineering practices, the system epitomizes the convergence of innovation and pragmatism in academia, laying the foundation for a future where technology empowers learning, collaboration, and institutional excellence.

### A. Problem Statement and Motivation

The landscape of higher education is characterized by a myriad of administrative challenges, ranging from inefficient course management systems to cumbersome communication channels between educators and students. Traditional methods of academic resource management often prove inadequate in meeting the dynamic demands of modern educational institutions, leading to bottlenecks, errors, and suboptimal user experiences. Recognizing these shortcomings, the project endeavors to address the pressing need for a comprehensive educational management system that streamlines administrative workflows, enhances collaboration, and improves accessibility for all stakeholders.

### B. Significance of the Project

The significance of the project lies in its potential to revolutionize the educational landscape by empowering educators and students with a robust, user-friendly platform that caters to their diverse needs. By centralizing course management, communication, and administrative tasks within a single, integrated system, the project aims to foster a more efficient, transparent, and engaging learning environment. Moreover, the project's emphasis on accessibility ensures that users of all backgrounds and abilities can benefit from its features, thereby promoting inclusivity and equity in education.

## II. LITERATURE REVIEW

In recent years, there has been a proliferation of educational platforms and online learning systems aimed at facilitating remote learning, improving course management, and enhancing professor-student interaction. A review of existing literature reveals a diverse array of systems, each offering unique features and functionalities tailored to the needs of educators and students. Platforms such as Moodle, Canvas, and Blackboard have gained widespread adoption in academic institutions worldwide, providing comprehensive course management tools, content delivery mechanisms, and assessment capabilities.

While these platforms have undoubtedly revolutionized the educational landscape, they are not without their limitations. One common critique is the complexity and learning curve associated with navigating these systems, particularly for novice users. Additionally, the rigid structure of some platforms may limit customization options and hinder adaptability to unique pedagogical approaches and institutional requirements. Moreover, concerns regarding data privacy, security vulnerabilities, and vendor lock-in have prompted educators and institutions to seek alternative solutions that offer greater control and transparency.

### A. Addressing Limitations and Proposing Solutions

The proposed project aims to address the limitations of existing educational platforms by providing a modern, user-centric system that prioritizes simplicity, flexibility, and security. By leveraging contemporary web technologies and best practices in user experience design, the project seeks to create an intuitive, accessible platform that empowers educators and students to engage effectively in the learning process.

One key strength of the proposed project is its emphasis on user-friendly interfaces and streamlined workflows, reducing the learning curve for both educators and students. By adopting a modular architecture and customizable templates, the system offers flexibility and adaptability to accommodate diverse teaching styles and institutional requirements. Furthermore, robust security measures, such as encryption, authentication, and access controls, ensure the privacy and integrity of user data, addressing concerns raised by previous systems.

Moreover, the project's focus on enhancing professor-student interaction through integrated communication tools, collaborative spaces, and personalized learning experiences

fosters a sense of community and engagement within the virtual classroom. By facilitating seamless communication, feedback exchange, and collaborative activities, the system promotes active learning and knowledge sharing, ultimately enriching the educational experience for all stakeholders.

### B. Front-End Components

The front-end of the platform is built using cutting-edge web technologies such as React.js, a popular JavaScript library for building user interfaces. React enables the creation of interactive and responsive UI components, facilitating a smooth and intuitive user experience. Additionally, HTML5 and CSS3 are used for structuring and styling the user interface, ensuring compatibility across different devices and browsers.

### C. Back-End Components

On the back end, the platform utilizes a robust and scalable architecture powered by Node.js, a lightweight and efficient JavaScript runtime environment. Node.js enables the development of server-side applications with non-blocking, event-driven I/O, making it ideal for handling concurrent requests and real-time interactions. Additionally, Express.js, a minimalist web framework for Node.js, is employed to streamline the development of RESTful APIs and manage server-side logic.

## III. USER AUTHENTICATION AND AUTHORIZATION

The educational platform implements robust user authentication and authorization mechanisms to ensure secure access to its features and data. This section discusses the methodologies employed for both student and professor roles, emphasizing role-based access control to regulate user privileges effectively.

### A. Authentication Middleware

The provided `authMiddleware.js` file contains middleware functions responsible for authenticating student and professor users based on JSON Web Tokens (JWT). Upon receiving a request, these middleware functions extract the JWT token from the request headers and attempt to verify its authenticity using a secret key stored in the server environment variables (`process.env.JWT.SECRET`).

1) *authenticateStudent*: This middleware function is designed to authenticate student users. It extracts the JWT token from the request headers, verifies its validity, and decodes the token to obtain student information. If the token is valid, the decoded student information is attached to the request object (`req.student`), allowing subsequent middleware or route handlers to access user details. If the token is invalid or missing, the middleware responds with a 401 Unauthorized status code and an appropriate error message.

2) *authenticateProfessor*: Similarly, this middleware function handles authentication for professor users. It follows the same procedure as *authenticateStudent*, extracting the JWT token from the request headers, verifying its authenticity, and decoding the token to retrieve professor information. Upon successful verification, the decoded professor details are

attached to the request object (req.professor). In case of an invalid or missing token, the middleware responds with a 401 Unauthorized status code and an error message.

#### B. Role-Based Access Control (RBAC)

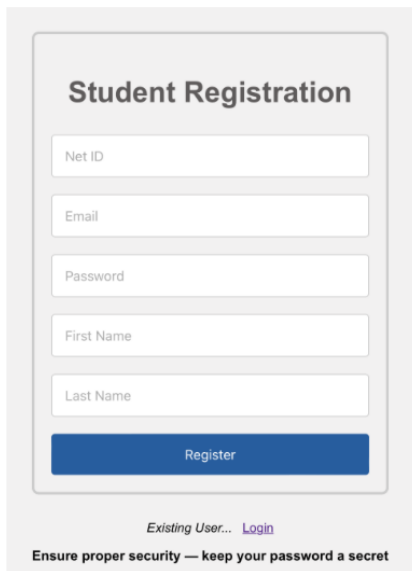
- Role-based access control is implemented to manage user privileges based on their roles (student or professor). Once authenticated, users are authorized to access specific features and data according to their roles. This ensures that students can only perform actions relevant to their role (e.g., submitting assignments, viewing course materials), while professors have access to additional functionalities (e.g., creating courses, posting announcements).
- By associating different permissions and restrictions with each role, the platform maintains data integrity, confidentiality, and security. Role-based access control prevents unauthorized users from accessing sensitive information or performing actions beyond their scope, thereby safeguarding the educational platform against potential security threats and unauthorized activities.

### IV. STUDENT FLOW OF OPERATIONS

The student operations domain is the cornerstone of the educational platform, encompassing a suite of functionalities meticulously designed to cater to the diverse needs of students and enhance their academic journey. From initial registration to ongoing engagement with course materials, these operations facilitate seamless interactions and empower students to make the most of their educational experience.

#### A. Student Registration and Login

The registerStudent function serves as the gateway for new users to join the platform. It prompts students to provide essential registration details, including their Net ID, email, password, first name, and last name.



The form is titled "Student Registration" and contains five input fields: "Net ID", "Email", "Password", "First Name", and "Last Name". Below these fields is a blue "Register" button. At the bottom, there is a link "Existing User... Login" and a security notice: "Ensure proper security — keep your password a secret".

Fig. 1. Student Registration

Upon submission, the platform verifies the uniqueness of the Net ID and proceeds to insert the student's information into the database, thus formalizing their presence within the platform.

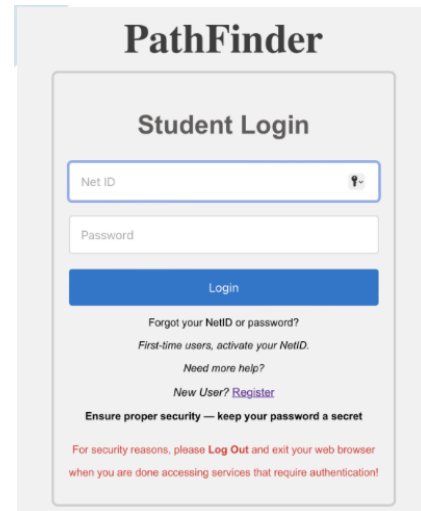
id	net_id	email	userPassword	first_name	last_name
1	sk2870	kollursuhas.us@gmail.com	Suhas123\$	Suhas	Kollur
2	pk674	parth.kharkar@gmail.com	Parth1810	Parth	Kharkar
3	ssk241	shreyashkalaled@gmail.com	ShreyashKalal	Shreyash	Kalal
4	dp1351	devanshipatel885@gmail.com	DevanshiPatel	Devanshi	Patel
5	sk2907	sarvesh.kharche@rutgers.edu	SarveshKharche	Sarvesh	Kharche
6	rk1108	rakshitha.kollur@rutgers.edu	RakshithaKollur	Rakshitha	Kollur
7	ag2384	ag2384@scarletmail.rutgers.edu	AdishGolechha	Adish	Golechha
NULL	NULL	NULL	NULL	NULL	NULL

students table

Fig. 2. Students Table

```
CREATE TABLE students (
  id INT AUTO_INCREMENT PRIMARY KEY,
  net_id VARCHAR(50) UNIQUE NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL,
  userPassword VARCHAR(100) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  last_name VARCHAR(50) NOT NULL
);
```

Conversely, the loginStudent function is pivotal in authenticating registered users. By validating the provided Net ID and password against existing records in the database, this function grants access to authenticated users, thereby enabling them to utilize platform features and services securely.



The form is titled "PathFinder" and "Student Login". It contains two input fields: "Net ID" and "Password". Below these fields is a blue "Login" button. Under the button, there are links: "Forgot your NetID or password?", "First-time users, activate your NetID.", "Need more help?", and "New User? Register". At the bottom, there is a security notice: "Ensure proper security — keep your password a secret" and "For security reasons, please Log Out and exit your web browser when you are done accessing services that require authentication!".

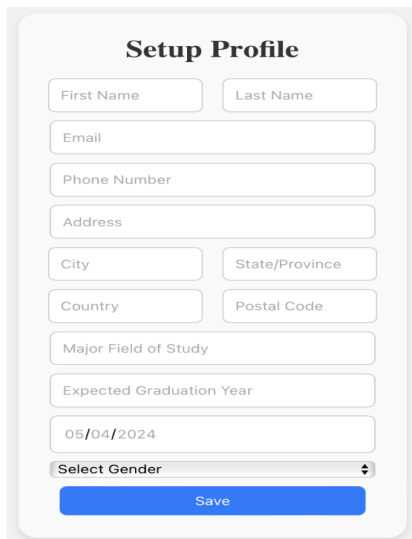
Fig. 3. Student Login

#### B. Profile Setup

A comprehensive profile is instrumental in personalizing the learning experience and facilitating effective communication between students and educators. The setupProfile function allows students to furnish their profiles with additional details such as phone number, address, city, state, country, postal

code, major field of study, expected graduation year, date of birth, and gender. These details enrich the student profile, enabling the platform to tailor content, notifications, and recommendations based on individual preferences and academic pursuits.

```
CREATE TABLE Profile (
    id INT AUTO_INCREMENT PRIMARY KEY,
    net_id VARCHAR(50),
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255),
    city VARCHAR(100),
    state VARCHAR(100),
    country VARCHAR(100),
    postal_code VARCHAR(20),
    major_field_of_study VARCHAR(255),
    expected_graduation_year INT,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other')
    ,FOREIGN KEY (net_id)
    REFERENCES students(net_id)
);
```



The form is titled "Setup Profile" and contains several input fields for student information. It includes fields for First Name, Last Name, Email, Phone Number, Address, City, State/Province, Country, and Postal Code. There are also fields for Major Field of Study, Expected Graduation Year (with a date picker set to 05/04/2024), and a dropdown menu for Select Gender. A blue "Save" button is at the bottom.

Fig. 4. Student Profile Setup

### C. Course Enrollment and Retrieval

Course enrollment is a pivotal aspect of student engagement, allowing students to select and enroll in courses aligned with their academic interests and requirements.

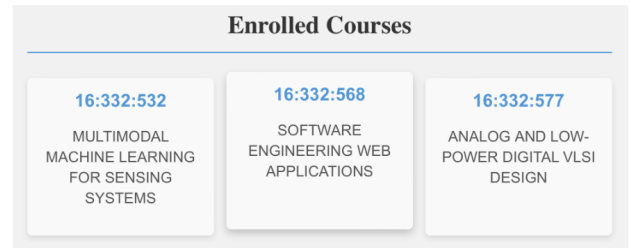
```
CREATE TABLE courses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    course_code VARCHAR(70) UNIQUE NOT NULL,
    course_name VARCHAR(100) NOT NULL,
    course_description TEXT NOT NULL
```

```
);
```

id	course_code	course_name	course_description
1	16:332:501	SYSTEMS ANALYSIS	Fundamentals of linear system concepts via solution of lin...
2	16:332:502	TECHNOLOGY ENTREPRENEURSHIP	Structure and framework of entrepreneurial endeavors, PH...
3	16:332:503	PROGRAMMING METHODOLOGY FOR NUMERICAL COMPUTING AND COMPUTATIONAL FINANCE	Fundamentals of object-oriented programming and C++ w...
4	16:332:504	SENSOR-BASED SYSTEMS AND APPLICATION	The course will develop skills in designing, programming, a...
5	16:332:505	CONTROL SYSTEM THEORY	Review of basic feedback concepts and basic controllers...
6	16:332:506	APPLIED CONTROLS	Review of state space techniques, transfer function metho...
7	16:332:507	SECURITY ENGINEERING	Essential principles, techniques, tools, and methods for sy...
8	16:332:508	DIGITAL CONTROL SYSTEMS	Review of linear discrete-time systems and the Z-transfor...
9	16:332:509	CONVEX OPTIMIZATION FOR ENGINEERING APPLICATIONS	The course develops the necessary theory, algorithms and...
10	16:332:510	OPTIMUM CONTROL SYSTEMS	Formulation of both deterministic and stochastic optimal co...

Fig. 5. Web Scrapped Courses

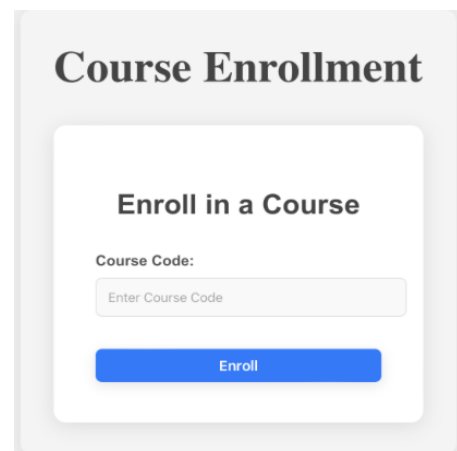
The enrollCourses function streamlines the enrollment process by verifying student authorization and inserting enrollment records into the database for selected courses.



The display is titled "Enrolled Courses" and shows three course cards. Each card contains a course code, course name, and a brief description. The courses are: 16:332:532 (MULTIMODAL MACHINE LEARNING FOR SENSING SYSTEMS), 16:332:568 (SOFTWARE ENGINEERING WEB APPLICATIONS), and 16:332:577 (ANALOG AND LOW-POWER DIGITAL VLSI DESIGN).

Fig. 6. Enrolled Courses

```
CREATE TABLE enrollments (
    id INT AUTO_INCREMENT PRIMARY KEY,
    net_id VARCHAR(50),
    course_id INT,
    course_code VARCHAR(70),
    course_name VARCHAR(100),
    FOREIGN KEY (net_id) REFERENCES
    students(net_id),
    FOREIGN KEY (course_id) REFERENCES
    courses(id)
);
```



The form is titled "Course Enrollment" and contains a section for "Enroll in a Course". It includes a label "Course Code:" and a text input field "Enter Course Code". A blue "Enroll" button is at the bottom.

Fig. 7. Course Enrollment

Furthermore, the getEnrolledCourses function empowers students to retrieve a comprehensive list of courses in which they are currently enrolled. This feature provides students with

visibility into their academic commitments, schedules, and progress, thereby facilitating effective course management and planning.

id	net_id	course_id	course_code	course_name
1	sk2870	11	16:332:512	NONLINEAR AND ADAPTIVE CONTROL THEORY
2	sk2870	3	16:332:503	PROGRAMMING METHODOLOGY FOR NUMERI...
3	sk2870	76	16:332:900	Cognitive Science
4	sk2870	77	16:332:901	Engineering Management
5	rk1108	76	16:332:900	Cognitive Science
6	ssk241	21	16:332:532	MULTIMODAL MACHINE LEARNING FOR SENSI...
7	ssk241	41	16:332:568	SOFTWARE ENGINEERING WEB APPLICATIONS
8	ssk241	48	16:332:577	ANALOG AND LOW-POWER DIGITAL VLSI DESI...
NULL	NULL	NULL	NULL	NULL

enrollments table

Fig. 8. Enrollment Table

### D. Accessing Course Announcements

Timely communication of course-related announcements is crucial for keeping students informed about important updates, deadlines, and events. The `getAnnouncements` function enables students to access course announcements specific to their enrolled courses.

```
CREATE TABLE announcements (
  id INT AUTO_INCREMENT PRIMARY KEY,
  course_id INT NOT NULL,
  title VARCHAR(255) NOT NULL,
  message TEXT NOT NULL,
  posted_on DATETIME DEFAULT
  CURRENT_TIMESTAMP,
  FOREIGN KEY (course_id) REFERENCES
  courses(id)
);
```

By querying the database for announcements associated with the student's Net ID and enrolled courses, this function delivers relevant information directly to the student, ensuring they stay abreast of course-related developments and initiatives. From assignment deadlines to course material updates, these announcements foster transparency, communication, and engagement within the learning community.

### E. Assignment Submission

The `submitAssignment` function facilitates the seamless submission of assignments, a core component of the learning process.

id	course_id	course_code	course_name	course_instruc...	assignment_ti...	assignment_description	assignment_deadl...
1	76	16:332:900	Cognitive Science	Suhas Kollur	Assignment 1	Basics of Cognitive Science	2024-05-16 00:00:00
2	76	16:332:900	Cognitive Science	Suhas Kollur	Assignment 2	Hands-on Cognitive Science	2024-05-21 00:00:00
3	77	16:332:901	Engineering Management	Suhas Kollur	Assignment 1	Introduction to Engineering Management	2024-05-23 00:00:00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

professor\_assignment table

Fig. 9. Assignments Table

```
CREATE TABLE assignments (
  id INT AUTO_INCREMENT PRIMARY KEY,
  course_id INT,
  title VARCHAR(100) NOT NULL,
  description TEXT,
```

```
deadline DATETIME,
FOREIGN KEY (course_id) REFERENCES
courses(id)
);
```

Students can submit their assignments by providing necessary details such as the assignment file, description, course code, and assignment title. Upon receiving the submission, the platform validates the input data, stores submission metadata in the database, and saves the assignment file to a designated location.

Fig. 10. Assignments Retrieval

```
Assignments for student: [
  {
    id: 1,
    course_id: 76,
    course_code: '16:332:900',
    course_name: 'Cognitive Science',
    course_instructor: 'Suhas Kollur',
    assignment_title: 'Assignment 1',
    assignment_description: 'Basics of Cognitive Science',
    assignment_deadline: 2024-05-16T04:00:00.000Z
  },
  {
    id: 2,
    course_id: 76,
    course_code: '16:332:900',
    course_name: 'Cognitive Science',
    course_instructor: 'Suhas Kollur',
    assignment_title: 'Assignment 2',
    assignment_description: 'Hands-on Cognitive Science',
    assignment_deadline: 2024-05-21T04:00:00.000Z
  },
  {
    id: 3,
    course_id: 77,
    course_code: '16:332:901',
    course_name: 'Engineering Management',
    course_instructor: 'Suhas Kollur',
    assignment_title: 'Assignment 1',
    assignment_description: 'Introduction to Engineering Management',
    assignment_deadline: 2024-05-23T04:00:00.000Z
  }
]
```

Fig. 11. Assignments

This streamlined process ensures that students can submit assignments efficiently while adhering to deadlines and fulfilling course requirements, thereby contributing to their academic success and progress.

```
CREATE TABLE submissions (
  id INT AUTO_INCREMENT PRIMARY KEY,
  assignment_id INT NOT NULL,
  student_id INT NOT NULL,
  file_path VARCHAR(255) NOT NULL,
  description TEXT,
  submission_date TIMESTAMP
  DEFAULT CURRENT_TIMESTAMP,
```

```
FOREIGN KEY (assignment_id)
REFERENCES assignments(id),
FOREIGN KEY (student_id)
REFERENCES students(id)
```

);

**Assignment Submission**

Select File:

Choose File Blockchain\_S...hreats-2.pdf

Selected File: Blockchain\_Security\_and\_Cryptocurrency\_Threats-2.pdf

Description:

Submitted by  
1. Suhas Kollur  
2. Parth Kharkar

I have attached a PDF file for your reference.

Submit

Assignment submitted successfully

Fig. 12. Submissions

#### F. Assignment Submission

In addition to the robust professor management and course administration features, the platform includes an assignment submission functionality that facilitates the seamless submission and evaluation of student assignments. This feature significantly enhances the teaching and learning process by providing a structured framework for assignment management. Below are the key aspects of the assignment submission functionality:

- **Student Submission Interface:** Students are provided with a user-friendly interface to submit their assignments electronically. The interface allows students to upload their assignment files, provide any necessary descriptions or comments, and submit their work within specified deadlines.

**Assignment Grades Dashboard**

**Grade an Assignment**

Post grades for students who have submitted assignments.

Post Grades

**View All Grades**

View the list of published grades for students from your course

View List of Graded Assignments

Fig. 13. Assignment Grades

- **File Management:** The platform supports various file formats for assignment submissions, including documents, presentations, spreadsheets, and multimedia files. It ensures compatibility with different types of assignments and enables students to showcase their work effectively.
- **Grading Interface:** Professors are provided with a dedicated grading interface to review and evaluate student assignments. The interface allows professors to provide

feedback, assign grades, and communicate with students regarding their performance.

Student Name	Submission Time	Description	Action
Akhil Bhargava	May 9, 2024 at 07:08:37 PM	Software Development Life Cycle (SDLC) is a systematic process used by software development teams to design, develop, and test high-quality software.	View
Akhil Bhargava	May 9, 2024 at 08:43:03 PM	The SDLC provides a structured framework that enables the production of software that meets or exceeds customer expectations. It encompasses entire time and cost estimates, and is efficient, reliable, and maintainable.	View
Akhil Bhargava	May 9, 2024 at 08:43:43 PM	The SDLC provides a structured framework that enables the production of software that meets or exceeds customer expectations. It encompasses entire time and cost estimates, and is efficient, reliable, and maintainable.	View

Fig. 14. Posting Grades

- **Integration with Course Management:** The assignment submission functionality seamlessly integrates with the broader course management system, allowing professors to correlate assignments with course objectives, learning outcomes, and assessment criteria. It provides a holistic view of student progress and performance within the course.

#### G. Logout Functionality

Secure logout functionality is essential for protecting user accounts and data from unauthorized access. The logoutStudent function ensures that authenticated students can terminate their sessions securely, thereby mitigating the risk of unauthorized access to their accounts. While the specific implementation may vary based on authentication mechanisms and session management protocols, the overarching goal remains consistent: to empower students with control over their account security and privacy, fostering trust and confidence in the platform.

### V. PROFESSOR FLOW OF OPERATIONS

#### A. Professor Registration and Authentication

The registerProfessor function is responsible for the registration of new professors within the educational platform. It begins by validating whether a professor with the provided Net ID already exists in the database. If no existing professor is found, the function proceeds to insert the details of the new professor into the professor's table.

id	net_id	email	userPassword	first_name	last_name
1	pk674	parthkharkar@gmail.com	Parth123	Parth	Kharkar
2	sk2870	suhaskollur@gmail.com	Suhas999	Suhas	Kollur
3	mm288	manasmaskar@gmail.com	mm456	Manas	Maskar
4	av860	amaanvora@gmail.com	av888	Amaan	Vora

professors table

Fig. 15. Professors Table

Upon successful registration, an appropriate success message is returned to the client. However, if a professor with the provided Net ID already exists, an error response indicating the duplication of records is returned.

```
CREATE TABLE professors (
  id INT AUTO_INCREMENT PRIMARY KEY,
  net_id VARCHAR(50) UNIQUE NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL,
  userPassword VARCHAR(100) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  last_name VARCHAR(50) NOT NULL
```



```
);
```

Conversely, the `loginProfessor` function handles the authentication process for professors attempting to access the platform. It validates the provided Net ID and password against existing records in the database. Upon successful authentication, the function generates a JSON Web Token (JWT), which serves as a secure authentication token granting access to protected routes within the platform. This token is then returned in response to the client, enabling them to access restricted resources securely. In case of failed authentication due to incorrect credentials or the absence of matching records, an error message indicating the invalidity of the provided credentials is returned to the client.

### B. Professor Profile Management

1) *Setup Professor Profile* (`setupProfessorProfile`): Professors can initialize their profiles by supplying essential details such as name, email, contact information, and personal preferences. This function validates the input data to ensure all required fields are provided. Upon successful validation, the profile information is inserted into the professor dashboard table. Finally, a success message is returned to confirm the completion of the profile setup process.

2) *Get Professor Profile* (`getProfessorProfile`): This feature enables professors to retrieve their profile information based on their unique Net ID. By querying the professor dashboard table, the system fetches the relevant profile data associated with the provided Net ID. The retrieved information includes details such as name, email, contact information, and other preferences. Subsequently, the profile data is returned in response to provide professors with access to their complete profile information.

3) *Update Professor Profile* (`updateProfessorProfile`): Professors have the flexibility to update their profile information as needed. This functionality validates the input data and allows modifications to specific fields within the professor's profile. Upon successful validation and update of the profile information in the professor dashboard table, a success message is returned to indicate the completion of the profile update process.

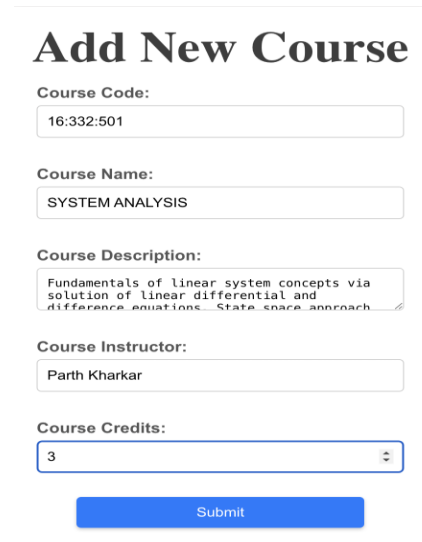
### C. Assignment Management

1) *Create Assignment* (`createAssignment`): This feature enables professors to create assignments tailored to specific courses. Upon receiving input data, including the assignment title, description, and deadline, the system validates the information and inserts the assignment details into the professor assignment table. Upon successful creation, a success message is returned, accompanied by the assignment ID for reference.

```
CREATE TABLE course_creation (
  id INT AUTO_INCREMENT PRIMARY KEY,
  course_id INT NOT NULL,
  course_instructor VARCHAR(100) NOT NULL,
  course_credits INT NOT NULL,
  FOREIGN KEY (course_id)
```

```
REFERENCES courses(id)
```

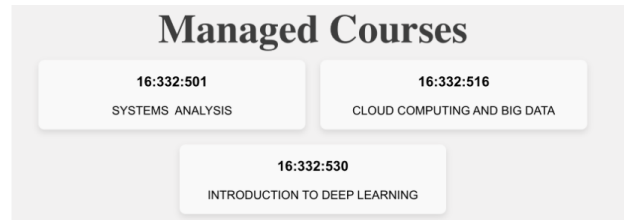
```
);
```



The form is titled "Add New Course" and contains several input fields: "Course Code:" with the value "16:332:501", "Course Name:" with the value "SYSTEM ANALYSIS", "Course Description:" with the text "Fundamentals of linear system concepts via solution of linear differential and difference equations. State space approach.", "Course Instructor:" with the value "Parth Kharkar", and "Course Credits:" with a dropdown menu showing the value "3". A blue "Submit" button is located at the bottom right of the form.

Fig. 16. Add New Course

2) *Get Assignments* (`getAssignments`): Professors can retrieve a list of assignments associated with a particular combined course ID. By querying the professor assignment table, the system fetches assignments linked to the specified course. The retrieved assignments include details such as title, description, and deadline, providing professors with comprehensive information about course-related tasks.



The dashboard is titled "Managed Courses" and displays three course cards. The first card shows "16:332:501" and "SYSTEMS ANALYSIS". The second card shows "16:332:516" and "CLOUD COMPUTING AND BIG DATA". The third card shows "16:332:530" and "INTRODUCTION TO DEEP LEARNING".

Fig. 17. Managed Courses

```
CREATE TABLE professor_assignment (
  id INT AUTO_INCREMENT PRIMARY KEY,
  course_id INT,
  course_code varchar(100) NOT NULL,
  course_name varchar(255) NOT NULL,
  course_instructor varchar(255) NOT NULL,
  assignment_title varchar(100) NOT NULL,
  assignment_description
  varchar(100) NOT NULL,
  assignment_deadline DATETIME,
  FOREIGN KEY (course_id)
  REFERENCES courses(id)
);
```

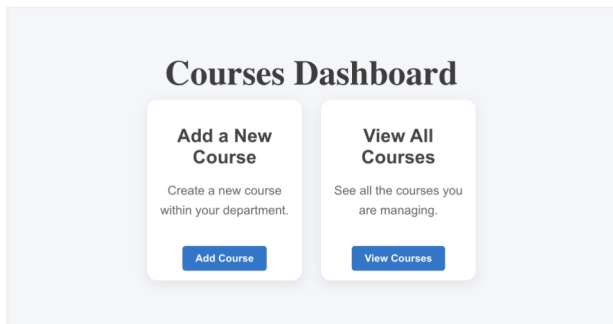


Fig. 18. Courses Dashboard

3) *Update Assignment (updateAssignment)*: This functionality empowers professors to modify assignment details, including the title, description, and deadline. After validating the input data, the system performs the necessary updates in the professor assignment table. Upon successful completion of the update process, a success message is returned to confirm the changes.

Fig. 19. Update Course Listing

```
CREATE TABLE combined_courses (
  id INT PRIMARY KEY AUTO_INCREMENT,
  course_id INT NOT NULL,
  course_code VARCHAR(255) NOT NULL,
  course_name VARCHAR(255) NOT NULL,
  course_description TEXT,
  course_instructor VARCHAR(255),
  course_credits INT
);
```

4) *Delete Assignment (deleteAssignment)*: Professors can remove assignments associated with a specific combined course ID and assignment ID. Upon validation of the input data, the system deletes the corresponding assignment from the professor assignment table. A success message is then returned to confirm the successful deletion of the assignment.

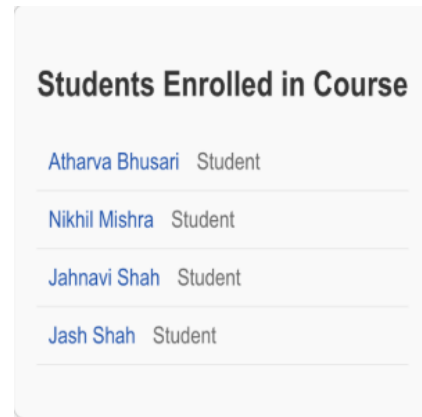


Fig. 20. Students Enrolled in the Course

#### D. Announcement Management

1) *Post Announcement (postAnnouncement)*: Professors can share important announcements with students by posting them for a specific course. This feature validates the input data, including the announcement title and message, and inserts the announcement details into the announcements table. Upon successful posting, a success message is returned along with the unique announcement ID for reference.

```
CREATE TABLE announcements (
  id INT AUTO_INCREMENT PRIMARY KEY,
  course_id INT NOT NULL,
  title VARCHAR(255) NOT NULL,
  message TEXT NOT NULL,
  posted_on DATETIME
  DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (course_id)
  REFERENCES courses(id)
);
```

2) *Get Announcements (getAnnouncements)*: This functionality enables professors to retrieve a list of announcements associated with a particular course ID. By querying the announcements table, the system fetches announcements linked to the specified course, providing professors with access to essential communication channels. The retrieved announcements include details such as title, message, and timestamp, ensuring comprehensive dissemination of course-related information.

3) *Update Announcement (updateAnnouncement)*: Professors have the ability to update announcement details, such as the title and message, to ensure relevance and accuracy. After validating the input data, the system performs the necessary updates in the announcements table, reflecting the latest information for students' reference. Upon successful completion of the update process, a success message is returned to confirm the changes, maintaining effective communication channels within the course environment.

#### E. Performing Unit Test

In addition to ensuring the functionality and reliability of the platform's features, it's essential to assess its performance





**Post Announcement**

Title:

Message:

**Post Announcement**

Fig. 21. Post Announcements

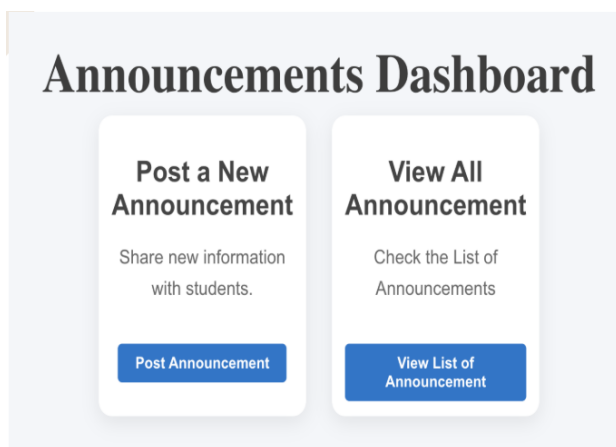


Fig. 22. Announcements Dashboard

to guarantee optimal user experience. One effective way to achieve this is by conducting unit tests using Lighthouse performance testing.

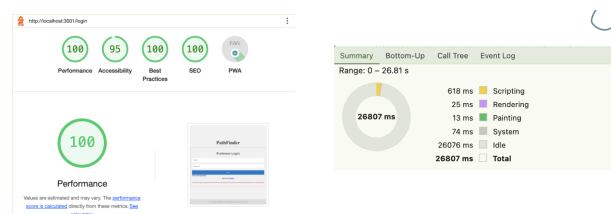


Fig. 23. Performance Unit Test

1) *Integration of Lighthouse into Unit Testing:* To incorporate Lighthouse performance testing into unit tests, developers can utilize Lighthouse CI, a set of commands and configurations designed specifically for continuous integration (CI) environments. By integrating Lighthouse CI into the unit testing pipeline, developers can automatically run performance audits on web pages or applications and generate reports as part of the testing process.

## CONCLUSION

In conclusion, the project has successfully developed a comprehensive platform for managing professor-related functionalities within an educational environment. Key findings and contributions include:

- **Efficient Professor Management:** The system offers seamless registration, authentication, and profile management capabilities for professors, enhancing administrative efficiency and user experience.
- **Effective Course Administration:** Through features like assignment creation, announcement posting, and student enrollment management, the platform streamlines course administration tasks, enabling professors to focus more on teaching and mentoring.
- **Secure Authentication:** Implementation of JWT-based authentication ensures secure access to protected routes, safeguarding professor accounts and sensitive data.
- **Enhanced Communication:** The system facilitates effective communication between professors and students through announcements, fostering an engaged learning community.

## FUTURE WORK

For future work, several enhancements and extensions can be considered:

- **Advanced Analytics:** Integrate analytics tools to provide insights into professor and student engagement, course performance, and learning outcomes, enabling data-driven decision-making and continuous improvement.
- **Collaborative Tools:** Incorporate collaborative features such as discussion forums, group assignments, and real-time chat to encourage peer interaction and collaborative learning experiences.
- **Integration with Learning Management Systems (LMS):** Enable seamless integration with existing LMS platforms to leverage their features and extend the system's functionality, offering a unified experience for both professors and students.
- **Accessibility and Inclusivity:** Ensure accessibility standards compliance and incorporate features to accommodate diverse learning needs, promoting inclusivity and equal access to education for all students.
- **Research and Development:** Explore avenues for research in educational technology, leveraging emerging technologies such as AI, machine learning, and natural language processing to innovate teaching methodologies, personalized learning experiences, and adaptive assessment strategies.