EduTech

Mini Project Report

Submitted by

Vikhnesh Sathyan

Reg. No.: AJC23MCA-2064

In Partial fulfillment for the Award of the Degree of

MASTER OF COMPUTER APPLICATIONS (MCA)

AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS

KANJIRAPPALLY

[Approved by AICTE, accredited by NAAC, Accredited by NBA.
Koovappally, KanjiraPally, Kottayam, Kerala – 6865182024-2025

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING AUTONOMOUS

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, "**EduTech**" is the Bonafide work of **Vikhnesh Sathyan (Regno: AJC23MCA-2064)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under **Amal Jyothi College of Engineering Autonomous, Kanjirappally** during the year 2024-25.

Mr. Jinson Devis
Internal Guide

Mr.Binumon Joseph
Coordinator

Rev. Fr. Dr. Rubin Thottupurathu Jose
Head of the Department

DECLARATION

I hereby declare that the project report **“EduTech”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from Amal Jyothi College of Engineering Autonomous during the academic year 2024-2025.

Date:	**Vikhnesh Sathyan**
KANJIRAPPALLY	**Reg: AJC23MCA-2064**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Director (Administration) **Rev. Fr. Dr. Roy Abraham Pazhayaparampil** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Mr**.**Binumon Joseph** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Mr. Jinson Devis** for his inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

Vikhnesh Sathyan

ABSTRACT

In today's dynamic educational environment, the need for personalized support to help students achieve their full potential is greater than ever. This paper presents the design and development of a multifaceted digital platform aimed at aiding students, particularly those who struggle academically, in their journey from basic proficiency to advanced levels of achievement. The platform leverages technology to provide an approach to student development, encompassing both academic excellence and extracurricular talent enhancement, including activities such as dance, music, and more.

MINI-PROJECT

STUDENTS:

- LOGIN AND SIGN-UP.
- Student Profile: Displays student's name, profile picture, and basic info.
- Notifications: Alerts for upcoming deadlines, new messages, and important updates.
- Academics: Grades, assignments, and course materials, quiz and other.
- Skill-Based Competitions and Challenges: Regular challenges or competitions related to different skills, such as coding, writing, or artistic performances.
- Resource Links: Embed widgets that link to relevant educational resources, such as online tutorials, articles, or videos.
- To-Do List: List of pending assignments and tests with deadlines.
- Study Tools: Interactive learning aids and resources.
- Flash Card

TEACHERS

●Profile Information: Displays teacher's name, profile picture, and basic info.
● Notifications: Alerts for student submissions, upcoming meetings, and important updates.
● Grades: Manage grading, feedback, and reports.
● Resources: Access teaching materials, lesson plans, and educational resources.
●Alerts: Notifications about students needing attention or support.
●Weekly Schedule: View and manage class tests, quiz, including changes or cancellations.
● Student List: View profiles of all students in each class.
● Parent Communication: Tools for communicating with parents and guardians.

PARENTS:

● Notifications: Alerts for important updates, messages from teachers, and upcoming events.
● Quick Access: Links to frequently used features such as student profile.

● Profile Information: Displays parent's name and profile picture.
● Communication: Messages and updates from teachers and school administration.
● Student Profile: Basic information about the child including name, grade, and class.
● Current Courses: List of subjects or courses the child is enrolled in.
● Recent Grades: Summary of recent grades or test scores

Admin:

● Student Profiles: Access to detailed student profiles
● Create and Manage Users: Add, edit, and delete user accounts including students, faculty, staff, and other administrators.
● Create and Update Courses: Add new courses, edit existing ones, and remove outdated courses.
● Upload and Manage Resources: Add, edit, and delete educational content such as lecture notes, videos, assignments, and other learning materials

CONTENT

List of Abbreviations

- AJAX - Asynchronous JavaScript and XML
- CSS - Cascading Style Sheets
- DBMS - Database Management System
- HTML - Hypertext Markup Language
- TS - JScript
- MySQL - My Structured Query Language
- PK - Primary Key

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

In today's dynamic educational environment, the need for personalized support to help students achieve their full potential is greater than ever. This paper presents the design and development of a multifaceted digital platform aimed at aiding students, particularly those who struggle academically, in their journey from basic proficiency to advanced levels of achievement. The platform leverages technology to provide an approach to student development, encompassing both academic excellence and extracurricular talent enhancement, including activities such as dance, music, and more.

1.2 PROJECT SPECIFICATION

1. EduTech

EduTech Platform: Improving Learning and Communication

2. Objective

To build a web-based platform that helps students, teachers, and parents manage academic activities like assignments, grades, and communication.

3. Scope

- **Students:** Track assignments, view grades, and receive notifications.
- **Teachers:** Manage assignments, grades, and provide feedback.
- **Parents:** Monitor student progress and communicate with teachers.
- **Admin:** Manage platform users and generate reports.

4. Key Features

- **Student Dashboard:** View assignments, performance, and reminders.
- **Teacher Dashboard:** Manage grades, assignments, and send alerts.
- **Parent Portal:** Track student progress and receive updates.

- **Admin Panel:** Manage users and platform data.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

In today's dynamic educational environment, the need for personalized support to help students achieve their full potential is greater than ever. This paper presents the design and development of a multifaceted digital platform aimed at aiding students, particularly those who struggle academically, in their journey from basic proficiency to advanced levels of achievement. The platform leverages technology to provide an approach to student development, encompassing both academic excellence and extracurricular talent enhancement, including activities such as dance, music, and more.

2.2 EXISTING SYSTEM

2.2.1 NATURAL SYSTEM STUDIED

The existing natural education system focuses on how traditional classroom environments operate to facilitate learning among students. This system emphasizes interaction, collaboration, and hands-on experiences, utilizing natural methods to enhance educational outcomes.

Key Features of the Natural System

1. Student-Centered Learning:
    - Emphasizes active participation and engagement from students.
    - Encourages students to take responsibility for their learning through exploration and inquiry.
2. Collaborative Learning:
    - Students work in groups to foster teamwork and communication skills.
    - Facilitates peer learning and sharing of knowledge among students.
3. Hands-On Experiences:
    - Incorporates practical activities, experiments, and real-world applications to reinforce theoretical knowledge.
    - Utilizes learning and field trips to connect students with their environment.
4. Flexible Learning Environment:
    - Adapts to the needs of students with varied learning styles and paces.
    - Provides spaces for individual study, group work, and creative projects.

2.2.2 DESIGNED SYSTEM STUDIED

Personalized Learning Dashboard:

- **Tailored Learning Paths**: Each student receives a customized dashboard reflecting the progress, strengths, and areas for improvement.
- **Adaptive Learning**: The system adjusts the difficulty of content based on the student's performance, providing a more effective learning experience.
- **Real-Time Progress Tracking**: Parents and teachers can view student performance metrics, engagement levels, and areas needing attention.

2.3 DRAWBACKS OF EXISTING SYSTEM

- Limited Personalization
- Inefficient Communication
- Resource Limitations
- Outdated Assessment Methods
- Inflexible Learning Environment

2.4 PROPOSED SYSTEM

The proposed EduTech platform is designed to transform the educational experience by integrating innovative technology, personalized learning approaches, and enhanced communication tools. It aims to create an engaging, adaptive, and collaborative learning environment that addresses the limitations of traditional education systems.

Objectives of the Proposed System

1. **Enhance Student Engagement**:
    - Utilize gamification and interactive content to make learning more enjoyable and motivating.
    - Foster a sense of community through collaboration and peer learning opportunities.
2. **Personalize Learning Experiences**:
    - Implement adaptive learning technologies that tailor educational content to individual student needs and preferences.

2.5 ADVANTAGES OF PROPOSED SYSTEM

- Improved Communication and Collaboration
- Reduced Student Anxiety and Stress
- Inclusive Learning for Special Needs Students
- Encourages Parent and Community Engagement
- Continuous Learning for Lifelong Education

CHAPTER 3

REQUIREMNT ANALYSIS

3.1 FEASIBILITY STUDY

The feasibility study evaluates the practicality of implementing the proposed EduTech platform by analyzing different aspects, including technical, economic, operational, legal, and schedule feasibility.

3.1.1 Economical Feasibility

Initial Investment: Costs include software development, cloud infrastructure, licenses.
Operational Costs: Ongoing expenses include maintenance, updates, cloud storage fees, and customer support.
Savings: Reduced costs for textbooks, printing, and administrative work
Revenue Opportunities: Subscription models, premium features for schools, and partnerships with educational institutions.
ROI Analysis: The platform will generate returns through improved learning outcomes

3.1.2 Technical Feasibility

1.Technology Stack: Choose a technology stack (e.g., Angular for frontend, Node.js for backend) that supports scalability and integration with educational tools.

2. System Architecture: Design an architecture that ensures performance, scalability, and security.

3. Security Measures: Implement robust security protocols to protect user data and comply with regulations like GDPR and FERPA. 4. Integration Capabilities: Ensure compatibility with existing Learning Management Systems (LMS) and other educational tools

3.1.2 Behavioral Feasibility

Behavioral Feasibility assesses whether the proposed system aligns with the behaviors, attitudes, and readiness of key stakeholders, including students, teachers, parents, and administrators. This section focuses on potential resistance to change, user adoption, and the psychological readiness to use the system.

3.1.4 Feasibility Study Questionnaire

1 Resource Links

- How often do you use external resources (e.g., online tutorials, articles, videos) for studying? (Often/Sometimes/Never)
- What type of resources do you find most helpful? (Text-based, Videos, Interactive tools)

2. To-Do List Management

- Do You Use Tools or apps to manage your assignments and deadlines? (Yes/No)
- Would You find a built-in to-do list feature helpful in your student portal? (Yes/No)

3.Student Interaction

- What Methods do you use to communicate with students outside of class? (Email/Messaging Apps/Online Platforms)
- Do you find alerts about students needing additional support useful? (Yes/No

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	-	12th Gen
RAM	-	16.0 GB
Hard disk	-	512GB

3.2.2 Software Specification

Front End	-	Angular
Back End	-	NodeJS
Database	-	MySQL
Client on PC	-	Windows 7 and above.
Technologies used	-	JS, HTML5, AJAX, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 Angular:

Angular is a powerful open-source front-end web application framework developed by Google. It is designed to simplify the development and testing of single-page applications (SPAs) by providing a structured framework for building dynamic web apps. Angular uses TypeScript, a superset of JavaScript, which enhances code quality and maintainability.

Key Features

- **Component-Based Architecture:** Applications are built using reusable components, making it easy to manage and maintain code.
- **Two-Way Data Binding:** Automatically synchronizes data between the model and the view, reducing boilerplate code.
- **Dependency Injection:** Facilitates code reusability and improves the organization of code through injectable services.
- **Routing:** Provides a powerful routing module to navigate between different views or components seamlessly.

3.3.2 MySQL

MySQL is a popular open-source relational database management system (RDBMS) that is widely used for web applications. It was first released in 1995 by Swedish software engineer Michael Widens and is now owned by Oracle Corporation.
MySQL uses Structured Query Language (SQL) as its primary language for managing data. It stores data in tables that are related to each other through keys and indexes. MySQL is a client-server system, meaning that it runs as a server and can be accessed by multiple clients simultaneously.

Some of the key features of MySQL include:

Scalability: MySQL is designed to handle large volumes of data and can scale to meet the needs of enterprise-level applications.

Security: MySQL provides robust security features, including user authentication, access control, and encryption of data in transit and at rest.

Speed: MySQL is optimized for fast data access and processing and can handle large volumes of queries simultaneously.

Portability: MySQL can run on multiple operating systems, including Windows, Linux, and macOS.

Flexibility: MySQL supports a wide range of data types, including integers, floating-point numbers, strings, and dates, and can be customized to meet the needs of specific applications.

Replication: MySQL supports replication, which allows data to be replicated across multiple servers for improved redundancy and fault tolerance.

CHAPTER 4
SYSTEM DESIGN

4.1 INTRODUCTION

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It is an important phase in the software development life cycle (SDLC) and involves translating the requirements identified in the system analysis phase into a detailed design that can be implemented by developers.

The system design phase typically involves creating a detailed technical design document that outlines the overall architecture of the system and the individual components that make up the system. This document should include detailed descriptions of the data structures, algorithms, and
interfaces that will be used in the system.

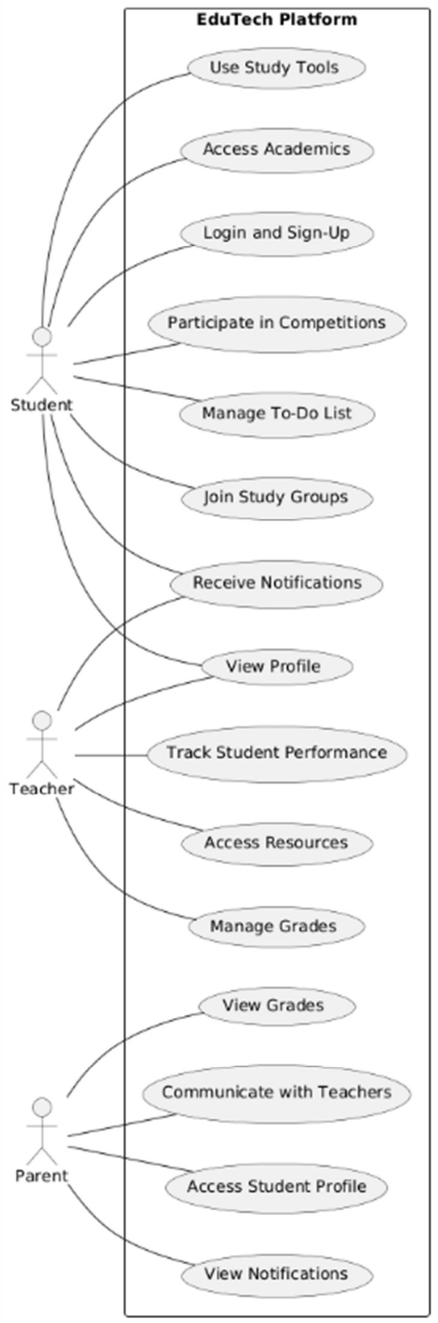
The goal of the system design phase is to create a blueprint for the development team that clearly defines how the system will be implemented and how it will interact with other systems and users.
It should also identify any potential technical challenges or limitations that may need to be addressed during the development process.

Overall, the system design phase is critical to the success of a software development project as it lays the foundation for the development team to build and test the system in a structured and organized manner. It also provides a clear roadmap for the project stakeholders to follow and can help ensure that the final product meets the requirements and expectations of the users.

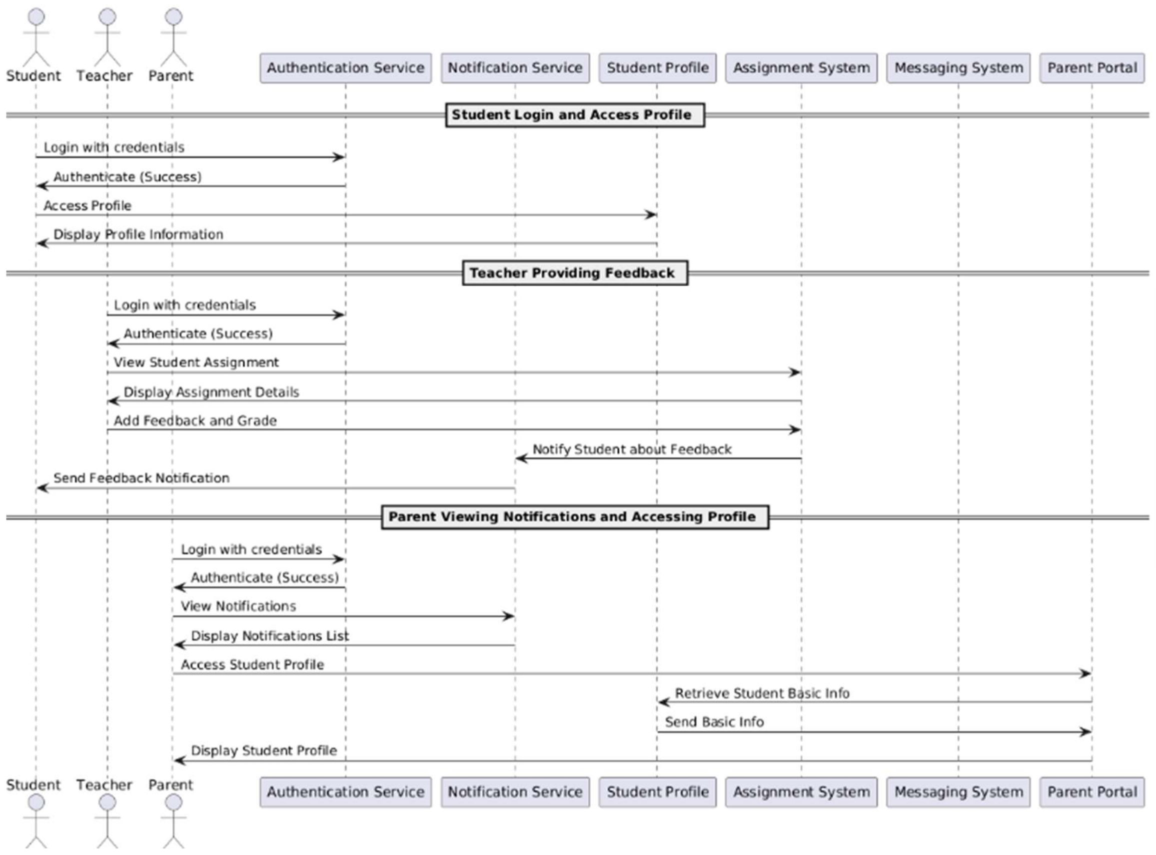
4.2UML DIAGRAM

Unified Modeling Language (UML) is a graphical language used to visualize, specify, construct, and document the artifacts of a software system. UML diagrams are used to represent various aspects of a system#39; s design, such as its structure, behavior, and interactions.

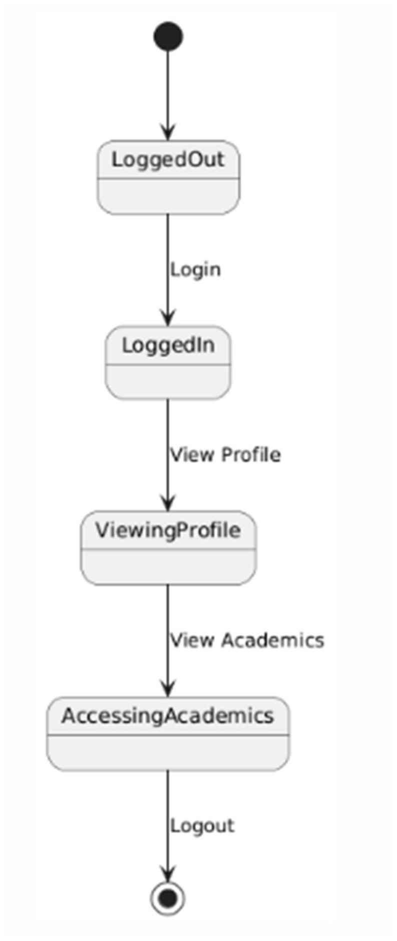
4.2.1 USE CASE DIAGRAM



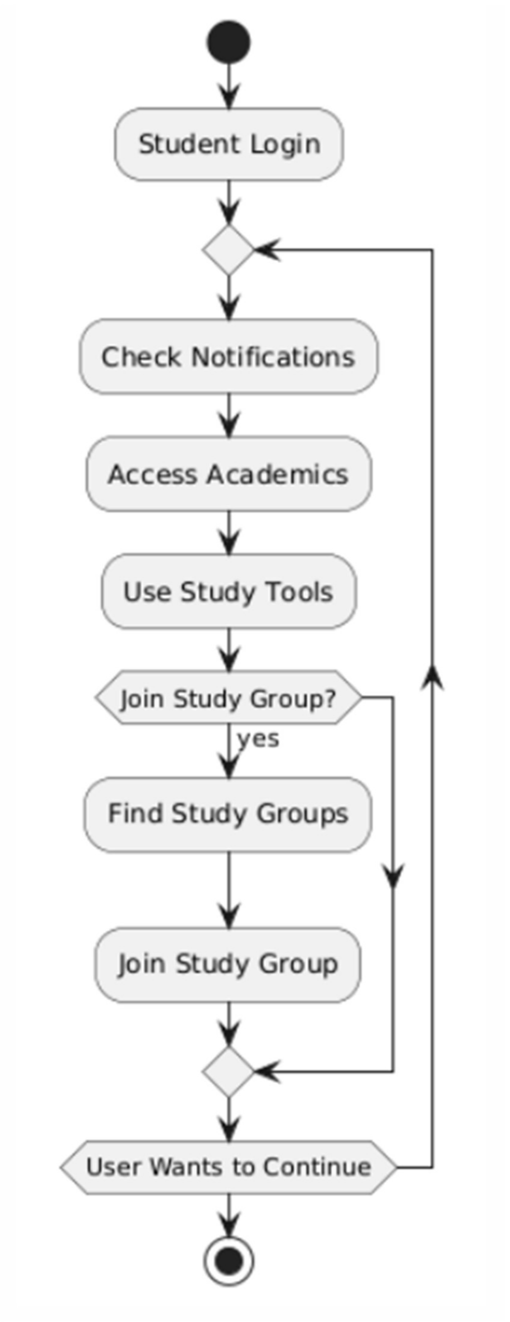
SEQUENCE DIAGRAM


Student
Teacher
Parent
Authentication Service
Notification Service
Student Profile
Assignment System
Messaging System
Parent Portal
Student Login and Access Profile
Login with credentials
Authenticate (Success)
Access Profile
Display Profile Information
Teacher Providing Feedback
Login with credentials
Authenticate (Success)
View Student Assignment
Display Assignment Details
Add Feedback and Grade
Notify Student about Feedback
Send Feedback Notification
Parent Viewing Notifications and Accessing Profile
Login with credentials
Authenticate (Success)
View Notifications
Display Notifications List
Access Student Profile
Retrieve Student Basic Info
Send Basic Info
Display Student Profile

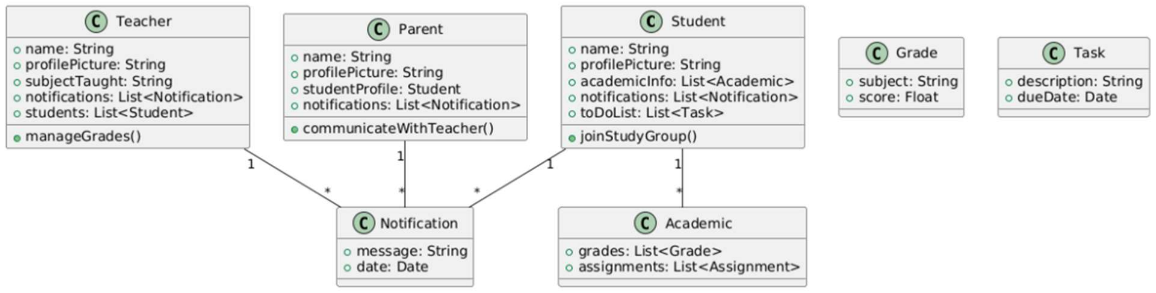
4.2.2 State Chart Diagram



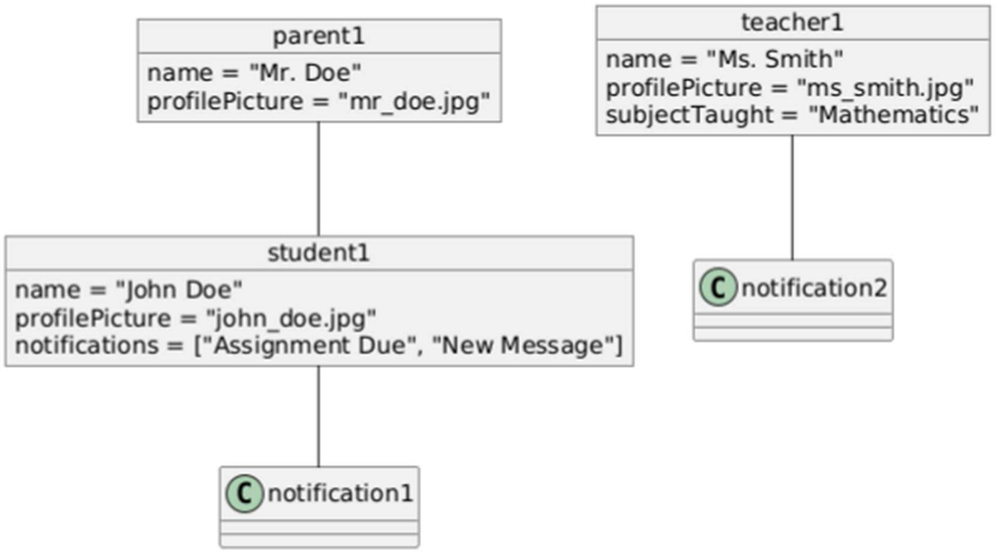
4.2.1 Activity Diagram



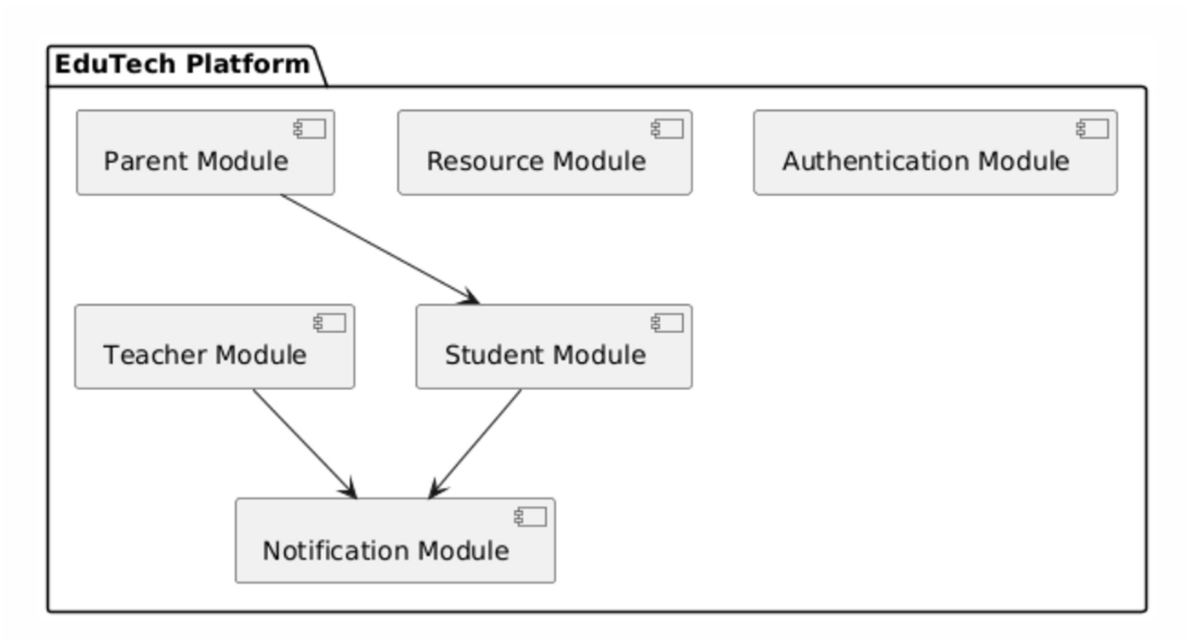
4.2.2 Class Diagram



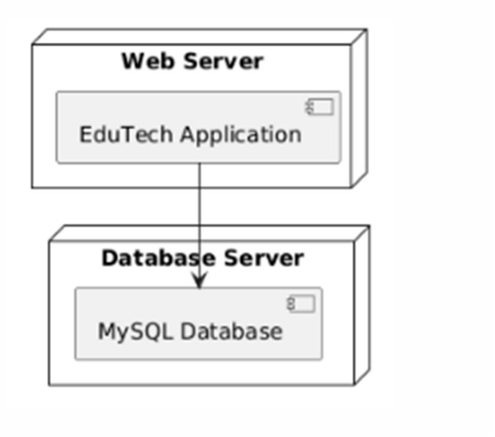
4.2.3 Object Diagram


parent1
name = "Mr. Doe"
profilePicture = "mr_doe.jpg"
teacher1
name = "Ms. Smith"
profilePicture = "ms_smith.jpg"
subjectTaught = "Mathematics"
student1
name = "John Doe"
profilePicture = "john_doe.jpg"
notifications = ["Assignment Due", "New Message"]
notification2
notification1

4.2.4 Component Diagram



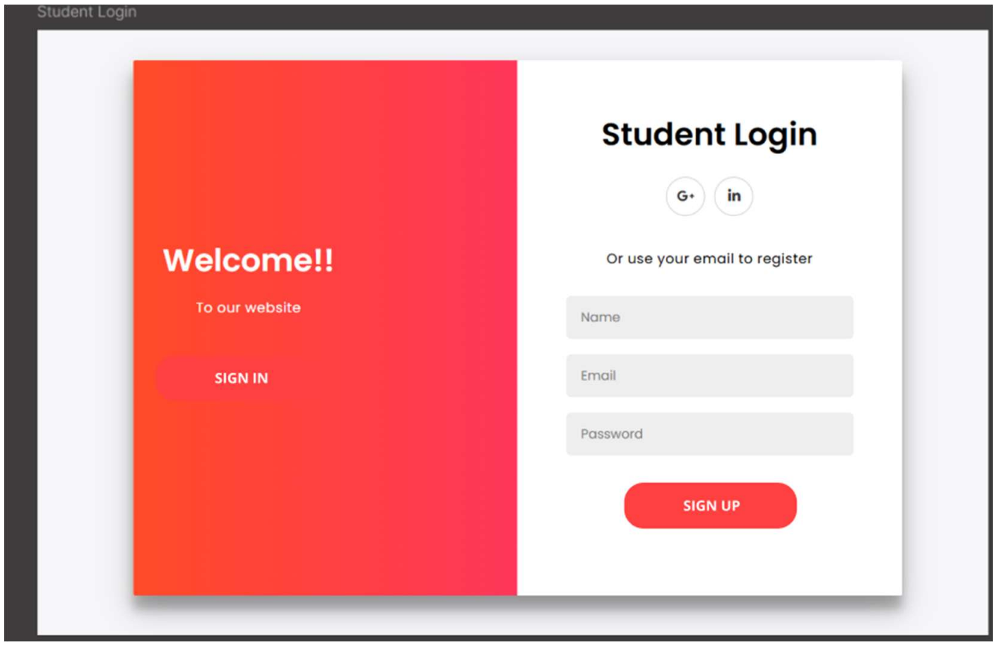
4.2.8 Deployment Diagram



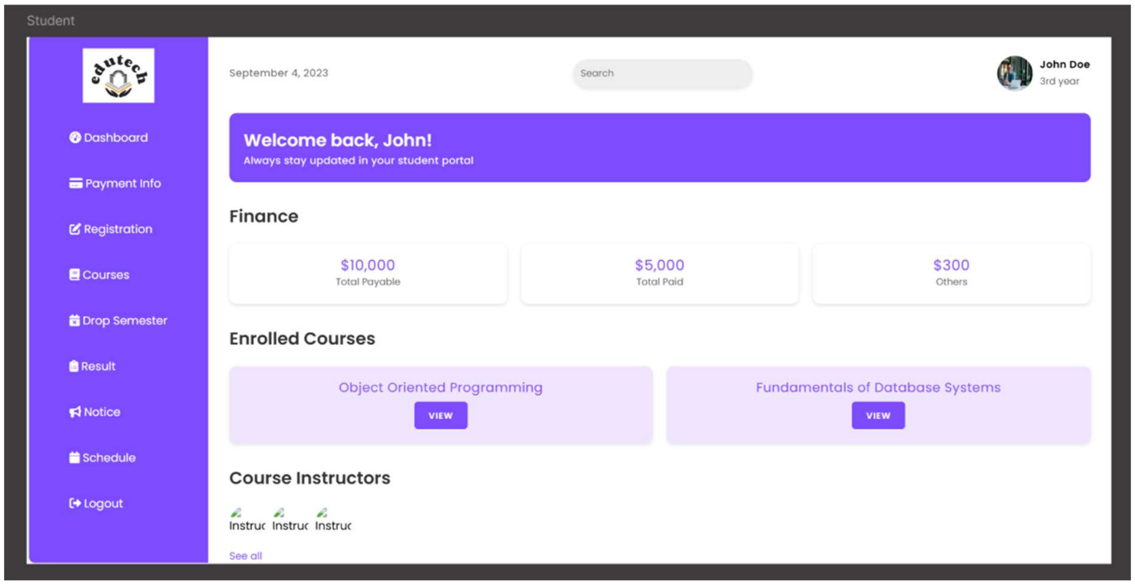
4.2.9 COLLABORATION DIAGRAM

4.3 USER INTERFACE DESIGN USING FIGMA

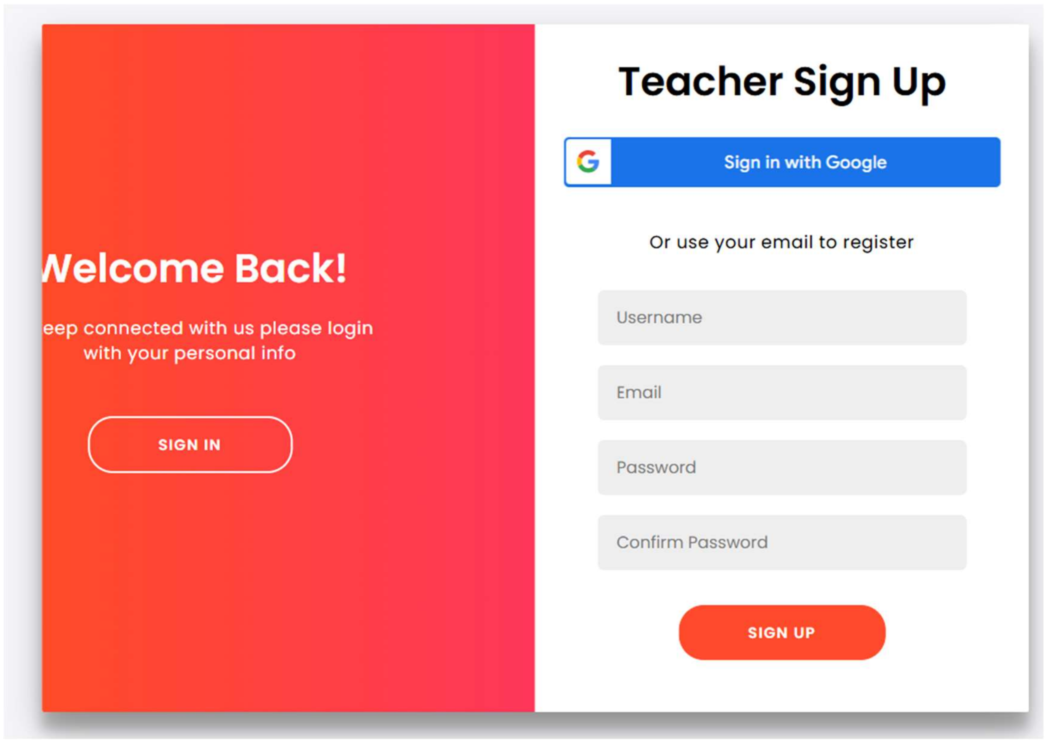
Form Name: Student-Login



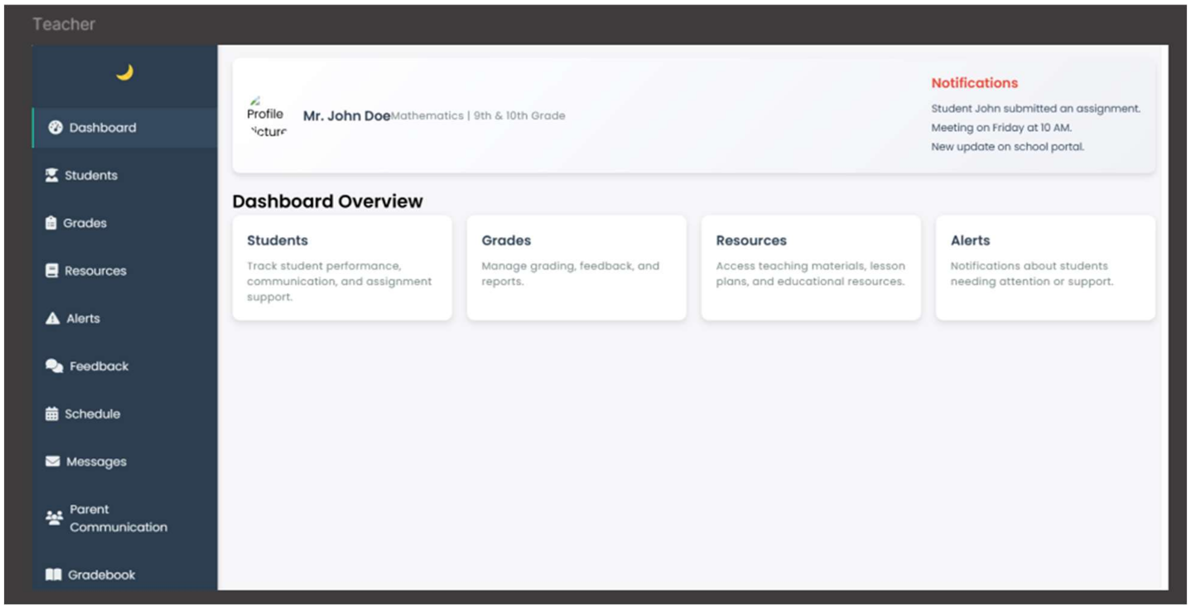
Form Name: Student Dashboard



Form Name: Teacher-Login



Form Name: Teacher Dashboard


Teacher

Dashboard
Students
Grades
Resources
Alerts
Feedback
Schedule
Messages
Parent Communication
Gradebook

Profile Picture Mr. John Doe Mathematics | 9th & 10th Grade

Notifications
Student John submitted an assignment.
Meeting on Friday at 10 AM.
New update on school portal.

Dashboard Overview

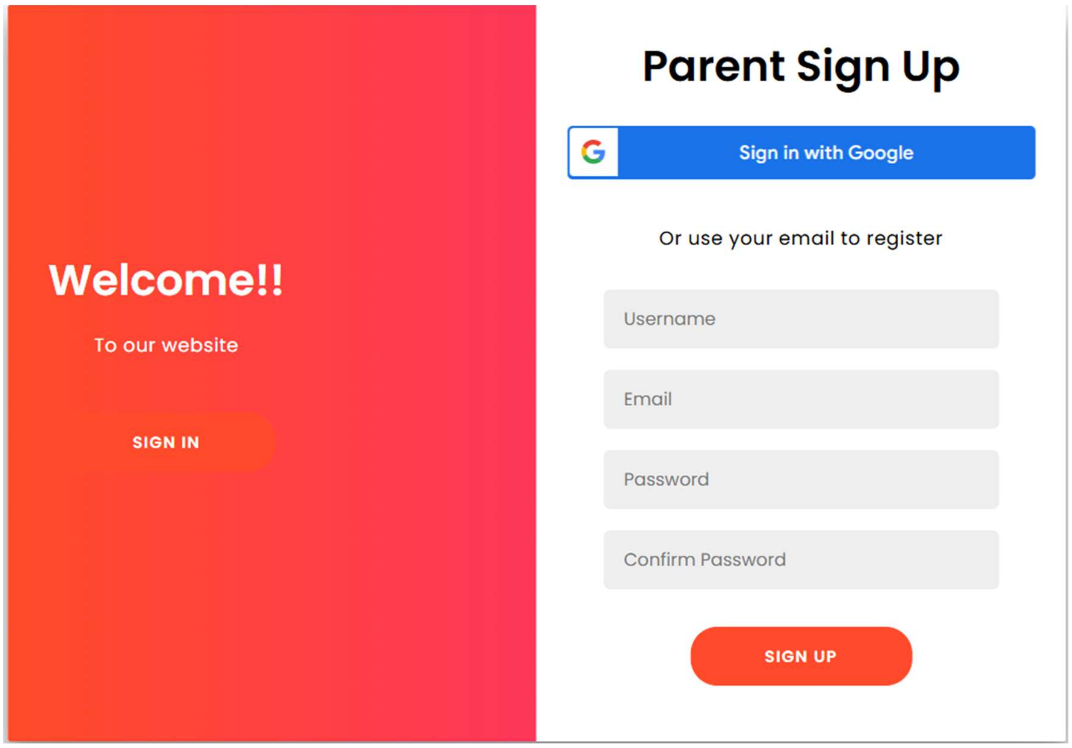
Students
Track student performance, communication, and assignment support.

Grades
Manage grading, feedback, and reports.

Resources
Access teaching materials, lesson plans, and educational resources.

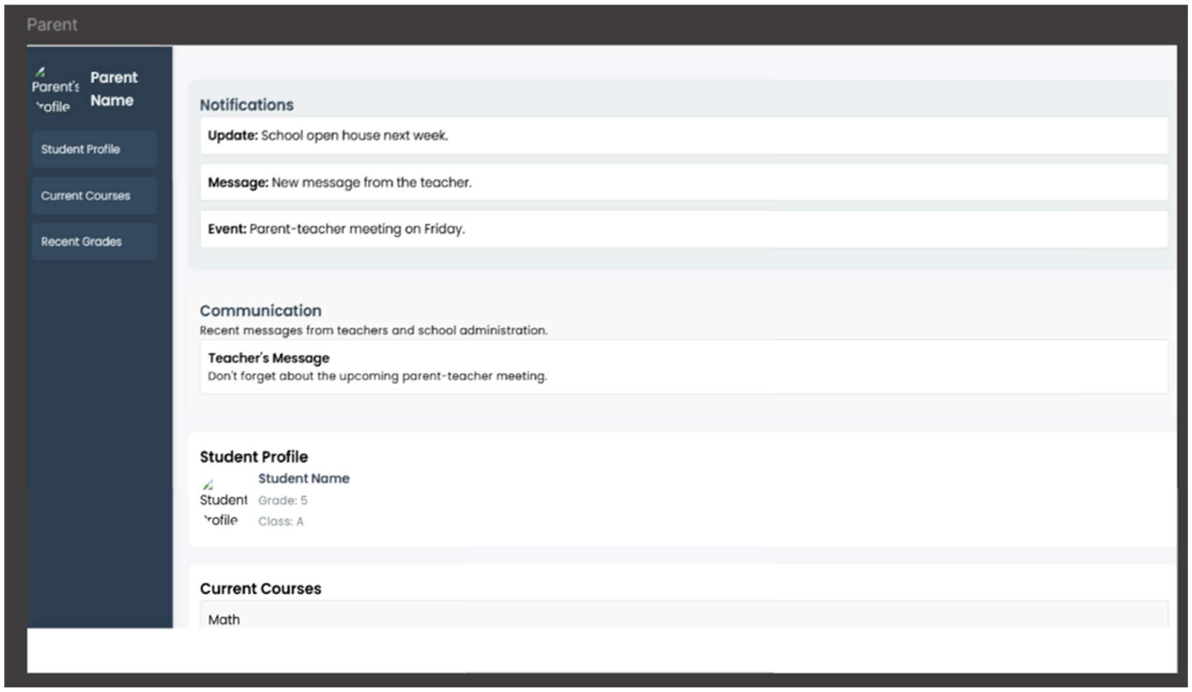
Alerts
Notifications about students needing attention or support.

Form Name: Parent Login


Welcome!!
To our website
SIGN IN

Parent Sign Up
Sign in with Google
Or use your email to register
Username
Email
Password
Confirm Password
SIGN UP

Form Name: Parent Dashboard



Parent

Notifications

Update: School open house next week.

Message: New message from the teacher.

Event: Parent-teacher meeting on Friday.

Communication

Recent messages from teachers and school administration.

Teacher's Message

Don't forget about the upcoming parent-teacher meeting.

Student Profile

Student Name

Grade: 5

Class: A

Current Courses

Math

4.4 DATABASE DESIGN

4.4.1 Relational Database Management System (RDBMS)

RDBMS stands for Relational Database Management System. It is a type of database management system that is based on the relational model, which organizes data into one or more tables (or &autocreation’s &quot;) of columns and rows.

In an RDBMS, data is organized into tables, with each table representing a specific entity or object in the system. Each row in a table represents a single instance of that entity, and each column represents a specific attribute or characteristic of the entity. The relationships between tables are defined through the use of keys, which are columns that are used to identify unique instances of an entity in the system.

RDBMSs are widely used in many different applications and industries, as they provide a reliable, scalable, and flexible way to store and manage large amounts of data. Some popular RDBMSs include MySQL, Oracle Database, Microsoft SQL Server, and PostgreSQL.

4.4.2 Normalization

Normalization is the process of organizing data in a relational database to reduce redundancy and improve data integrity. The goal of normalization is to ensure that each piece of data is stored in only one place and that it is easy to maintain and update.

Normalization is achieved by applying a series of rules to the database design, called normal forms. The most commonly used normal forms are first normal form (1NF), second normal form (2NF), third normal form (3NF), and so on.

Normalization can improve the efficiency and performance of a database, as well as make it easier to maintain and update. However, excessive normalization can also lead to increased complexity and decreased performance, so it is important to strike a balance between normalization and practical considerations for the particular database and its intended use

4.4.4 Indexing

4.5 TABLE DESIGN

1.Tbl_users_login

Eg.Primary key: **login id**

Eg.Foreign key: **login id** references table **Tbl_users_login**

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	user_id		PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each user
2	username	VARCHAR (50)	UNIQUE	User's login username
3	password	VARCHAR (255)		User's encrypted password
4	email	VARCHAR (100)	UNIQUE	User's email address

2.Students_table:

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	student_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each student
2	username	Varchar	UNIQUE	Student's login username
3	password	Varchar		Student's encrypted password

3.Tbl_Resource

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	Resource_id	INT (PK) (50)	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each user
2	resource_name	VARCHAR (50)	UNIQUE	Name of the resource
3	resource_ type	VARCHAR (255)		User's encrypted password
4	link	VARCHAR (100)		URL link to the resource

4.Tbl_To-do

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	student_id	INT(30)	PRIMARY KEY, AUTO_INCREMENT	References student_id from Student Profile
2	task name	VARCHAR (50)	UNIQUE	Name of the task or assignment
3	due date	DATETIME		Deadline for the task
4	email	ENUM	UNIQUE	'pending', 'completed'

5.Tbl_Parent

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	parent_id	INT(30)	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each parent
2	name	VARCHAR (50)		Full name of the parent
3	student_id	INT	FOREIGN KEY	References the student associated with this parent
4	email	ENUM	UNIQUE	'pending', 'completed'

6.Tbl_message

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	message_id	INT(30)	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each message
2	message_content	TEXT		ID of the user who received the message
3	subject	VARCHAR (100)		Subject or title of the message
4	timestamp	DATETIME	DEFAULT	Date and time the message was sent

7.Tbl_question

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	question_id	INT(30)	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each question
2	question_content	TEXT		The text content of the question
3	subject	VARCHAR (100)		Subject or title of the message
4	timestamp	DATETIME	DEFAULT	Date and time the message was sent

CHAPTER 5
SYSTEM TESTING

5.1 INTRODUCTION

System testing is a process of evaluating the functionality and performance of a software or application as a whole, to ensure that it meets the specified requirements and is fit for purpose. The goal of system testing is to identify and correct any defects or issues that may affect the system#39;s functionality, usability, reliability, security, and other quality attributes.

System testing is typically conducted after the completion of integration testing, which tests the interactions between different components or modules of the system. System testing involves testing the entire system in a realistic environment, using real-world data and scenarios, to simulate the end users#39;s experience.

5.2 TEST PLAN

A test plan is a document that outlines the objectives, scope, approach, and resources required for testing a software system or application. The purpose of a test plan is to guide the testing process and ensure that all necessary tests are conducted and documented.

Here are some of the key components of a test plan:

Test objectives: This section outlines the overall goals and objectives of the testing effort, and how the testing will help to achieve the project#39; s overall goals.

Test scope: This section defines the scope of the testing effort, including the areas of the system to be tested, the types of testing to be conducted, and any limitations or constraints on the testing effort.

Test approach: This section outlines the overall approach to be taken for testing the system, including the testing methodology, tools and techniques to be used, and any specific testing strategies or tactics.

Test resources: This section lists the resources required for the testing effort, including personnel, hardware, software, and any other equipment or facilities needed for testing.

Test deliverables: This section lists the deliverables that will be produced during the testing effort, such as test plans, test cases, test reports, and any other documentation.

Test risks: This section identifies the potential risks and issues that may arise during the testing effort, and outlines strategies for mitigating or addressing them.

5.2.1 Unit Testing

Unit testing is a type of software testing that involves testing individual units or components of asoftware application in isolation from the rest of the system. The purpose of unit testing to ensure that each unit of code, such as a function or method, works correctly and meets its functionality:

Unit testing can help catch bugs early in the development cycle, before they have a chance to cause larger problems in the system. It can also help developers to identify and fix issues moreIn order to perform unit testing, developers use specialized testing frameworks and tools that help automate the process and provide feedback on the success or failure of individual tests. Common.

5.2.2 Integration Testing

Integration testing is a type of software testing that aims to test how different components or modules of a software application work together. The purpose of integration testing is to verify that these components can communicate and exchange data with each other as expected, and that they function correctly as a unified system.

Integration testing typically occurs after unit testing has been completed, and it may involve testing different modules or components of the application together in pairs or in larger groups. This can be done in a variety of ways, including using stubs or mock objects to simulate the behavior of missing components, or by testing against a live system or database.

5.2.3 Validation Testing or System Testing

Validation testing, also known as system testing, is a type of software testing that aims to ensure that a software system meets its intended purpose and satisfies the requirements of the stakeholders. It is typically performed after integration testing has been completed and involves testing the entire system as a whole, rather than individual components.

The purpose of validation testing is to verify that the system functions correctly in the context of the larger environment in which it will be used. This includes testing the system#39;s functionality, usability, performance, reliability, and security, among other factors.

Validation testing may involve a combination of manual and automated testing techniques, and may be conducted using various testing methods, such as:

Functional testing: This involves testing the system#39;s functionality to ensure that it meets the requirements of the stakeholders.

Usability testing: This involves testing the system#39;s ease of use and user interface to ensure that it is intuitive and user-friendly.

Performance testing: This involves testing the system#39;s speed, scalability, and resource usage to ensure that it can handle the expected workload.

Security testing: This involves testing the system#39;s security features and vulnerabilities to ensure that it is protected against unauthorized access or malicious attacks.

5.2.4 Output Testing or User Acceptance Testing

Output testing, also known as user acceptance testing (UAT), is a type of software testing that involves testing the software system from the perspective of the end-user. The purpose of user acceptance testing is to verify that the system meets the requirements and expectations of the stakeholders, and that it functions correctly in the context of the user's workflow and environment. User acceptance testing is typically performed by a group of users or stakeholders who are representative of the target audience for the software system. These users are given specific scenarios or tasks to perform, and they provide feedback on their experience using the system. The focus of user acceptance testing is on the output or results of the system, rather than the individual components or technical details.

The main objectives of user acceptance testing are to:

- Verify that the system meets the requirements and expectations of the stakeholders.
- Validate that the system is easy to use and understand.
- Confirm that the system is reliable, accurate, and produces the expected output.
- Ensure that the system integrates seamlessly with other systems or tools used by the users.
- Identify any remaining defects or issues that need to be addressed before the system is released.

User acceptance testing is important because it provides a final check to ensure that the software system is ready for deployment and meets the needs of the end-users. It helps to validate that the system will be accepted and adopted by the users, which can ultimately determine the success of the project.

5.2.5 Automation Testing

Automation testing is a type of software testing that involves the use of tools and software to automate the execution of test cases and the comparison of actual results with expected results. The purpose of automation testing is to improve the efficiency and effectiveness of the testing process by reducing the time and effort required to execute tests and analyze results.

Automation testing can be applied to different types of testing, including unit testing, integration testing, and system testing. It involves the use of specialized tools and frameworks that can automate the testing process, such as:

- Test automation frameworks: These provide a structured approach to automate the testing process, including test case management, test data management, and reporting.

- Test scripting tools: These enable the creation and execution of automated test scripts, which can simulate user interactions with the software system.

However, automation testing also has some limitations, such as the need for specialized skills and resources to develop and maintain automated tests, as well as the inability to test certain aspects of the system that require human intuition or judgment.

5.2.6 Selenium Testing

- Cross-browser testing: Selenium testing allows for automated testing of web applications across different browsers, including Chrome, Firefox, Safari, and Internet Explorer.
- Record and playback: Selenium IDE, a Selenium testing tool, allows testers to record and playback interactions with the web application, which can be useful for creating test scripts.

Example:

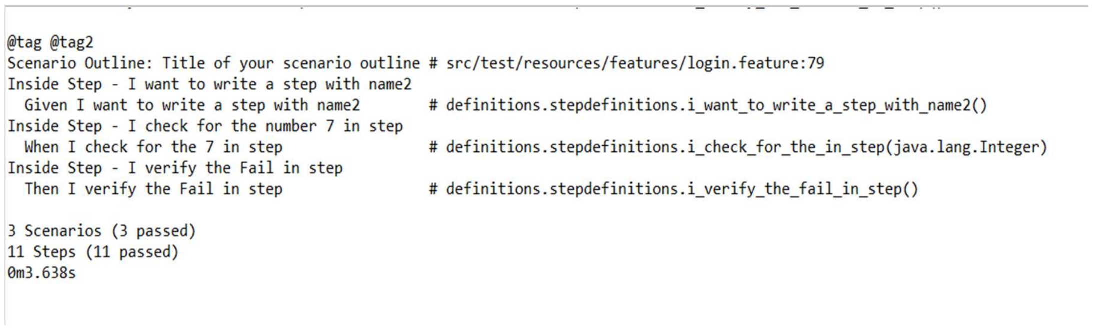
Test Case 1

Code

```
package definitions; // Ensure that this matches your package structure
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.cucumber.java.en.*;
public class stepdefinitions {
    WebDriver driver;
    // Step to open the browser
    @Given("browser is open")
    public void browser_is_open() {
        System.out.println("Inside Step - browser is open");
        // Set the path to your ChromeDriver or other browser driver
        System.setProperty("webdriver.chrome.driver",
"C:\\Users\\vikhn\\eclipse-workspace\\2\\src\\test\\resources\\features\\chromedriver.exe");
        driver = new ChromeDriver();
        driver.manage().window().maximize();
    }
    // Step to navigate to the login page
    @And("user is on login page")
    public void user_is_on_login_page() {
        System.out.println("Inside Step - user is on login page");
        driver.navigate().to("http://localhost:4200/teacher-login");
    }
    // Step to enter username and password
    @When("user enters username and password")
    public void user_enters_username_and_password() {
        System.out.println("Inside Step - user enters username and password");
      // Example locators for the username and password fields
        driver.findElement(By.id("email")).sendKeys("vikky1@gmail.com");
        driver.findElement(By.id("password")).sendKeys("12345678");
    }
    // Step for clicking the login button
    @And("user clicks on login")
    public void user_clicks_on_login() {
        System.out.println("Inside Step - user clicks on login");
        // Example locator for the login button
        driver.findElement(By.id("signIn")).click();
    }
    // Step to verify navigation to the home page
    @Then("user is navigate to the home page")
    public void user_is_navigate_to_the_home_page() {
        System.out.println("Inside Step - user is navigate to the home page");
        // Example verification step (you can add actual verification code here)
        String expectedUrl = "http://localhost:4200/teacher-login/t-dashboard";
        String actualUrl = driver.getCurrentUrl();
        if (actualUrl.equals(expectedUrl)) {
            System.out.println("Login successful, user is on home page.");
        } else {
            System.out.println("Login failed, user is not on home page.");
        }
        // Close the browser
        driver.quit()
    }
    // Step for scenario outline example
    @Given("I want to write a step with name1")
    public void i_want_to_write_a_step_with_name1() {
        System.out.println("Inside Step - I want to write a step with name1");
    }
    @When("I check for the {int} in step")
    public void i_check_for_the_in_step(Integer int1) {
        System.out.println("Inside Step - I check for the number " + int1 + " in step");
    }
```

```
    @Then("I verify the success in step")
    public void i_verify_the_success_in_step() {
        System.out.println("Inside Step - I verify the success in step");
    }
    @Given("I want to write a step with name2")
    public void i_want_to_write_a_step_with_name2() {
        System.out.println("Inside Step - I want to write a step with name2");
    }
    @Then("I verify the Fail in step")
    public void i_verify_the_fail_in_step() {
        System.out.println("Inside Step - I verify the Fail in step");
    }
}
```

Screenshot



```
@tag @tag2
Scenario Outline: Title of your scenario outline # src/test/resources/features/login.feature:79
Inside Step - I want to write a step with name2
  Given I want to write a step with name2       # definitions.stepdefinitions.i_want_to_write_a_step_with_name2()
Inside Step - I check for the number 7 in step
  When I check for the 7 in step                # definitions.stepdefinitions.i_check_for_the_in_step(java.lang.Integer)
Inside Step - I verify the Fail in step
  Then I verify the Fail in step                # definitions.stepdefinitions.i_verify_the_fail_in_step()

3 Scenarios (3 passed)
11 Steps (11 passed)
0m3.638s
```

Eg. Test Report

Test Case 1

Project Name: EduTech	
Login Test Case	
Test Case ID: Test_1	**Test Designed By: Vikhnesh Sathyan**
Test Priority(Low/Medium/High):	**Test Designed Date: 22/10/24**
Module Name: Login	**Test Executed By : Mr.Jinson Devis**
Test Title : Login Test case	**Test Execution Date: 22/10/24**
Description:	
Pre-Condition : User has valid username and password	

Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fai l)**
1	Navigate to login page	Vikhneshsathyan@gmail.com	Username and password field are successfully filled	Username and password field are successfully filled	Pass
2	Submit the login form	N/A	Browser window is successfully closed	Browser window is successfully closed	Pass
3					
4	Close the browser window	N/A	Browser window is successfully closed	Browser window is successfully closed	Pass
5					
6					
7					
Post-Condition:					

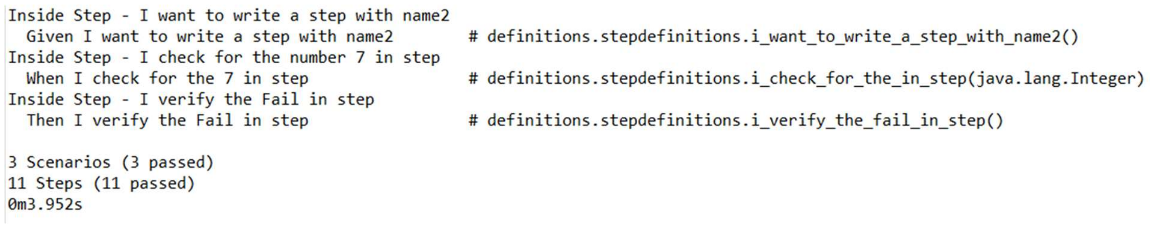
Test Case 2:

```
package definitions; // Ensure that this matches your package structure
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.cucumber.java.en.*;
public class stepdefinitions {
    WebDriver driver;
    // Step to open the browser
    @Given("browser is open")
    public void browser_is_open() {
        System.out.println("Inside Step - browser is open")
        // Set the path to your ChromeDriver or other browser driver
        System.setProperty("webdriver.chrome.driver",           "C:\\Users\\vikhn\\eclipse-
workspace\\2\\src\\test\\resources\\features\\chromedriver.exe");
        driver = new ChromeDriver();
        driver.manage().window().maximize();
    // Step to navigate to the login page
    @And("user is on login page")
    public void user_is_on_login_page() {
        System.out.println("Inside Step - user is on login page");
        driver.navigate().to("http://localhost:4200/messages");
    }
    // Step to enter username and password
    @When("user enters username and password")
    public void user_enters_username_and_password() {
        System.out.println("Inside Step - user enters username and password");
        // Example locators for the username and password fields
        driver.findElement(By.id("message")).sendKeys("notes")
    }
    // Step for clicking the login button
    @And("user clicks on login")
    public void user_clicks_on_login() {
```

```
    System.out.println("Inside Step - user clicks on login");
    // Example locator for the login button
    driver.findElement(By.id("messages")).click();
}
// Step to verify navigation to the home page
@Then("user is navigate to the home page")
public void user_is_navigate_to_the_home_page() {
    System.out.println("Inside Step - user is navigate to the home page")
    // Example verification step (you can add actual verification code here)
    String expectedUrl = "http://localhost:4200/student-messages";
    String actualUrl = driver.getCurrentUrl();
    if (actualUrl.equals(expectedUrl)) {
        System.out.println("Login successful, user is on home page.");
    } else {
        System.out.println("Login failed, user is not on home page.");
    }
    // Close the browser
    driver.quit();
}
// Step for scenario outline example
@Given("I want to write a step with name1")
public void i_want_to_write_a_step_with_name1() {
    System.out.println("Inside Step - I want to write a step with name1");
}
@When("I check for the {int} in step")
public void i_check_for_the_in_step(Integer int1) {
    System.out.println("Inside Step - I check for the number " + int1 + " in step");
}
@Then("I verify the success in step")
public void i_verify_the_success_in_step() {
    System.out.println("Inside Step - I verify the success in step");
}
```

```
    @Given("I want to write a step with name2")
    public void i_want_to_write_a_step_with_name2() {
        System.out.println("Inside Step - I want to write a step with name2");
    }
    @Then("I verify the Fail in step")
    public void i_verify_the_fail_in_step() {
        System.out.println("Inside Step - I verify the Fail in step");
    }
}
```

Screenshot



```
Inside Step - I want to write a step with name2
  Given I want to write a step with name2       # definitions.stepdefinitions.i_want_to_write_a_step_with_name2()
Inside Step - I check for the number 7 in step
  When I check for the 7 in step                # definitions.stepdefinitions.i_check_for_the_in_step(java.lang.Integer)
Inside Step - I verify the Fail in step
  Then I verify the Fail in step                # definitions.stepdefinitions.i_verify_the_fail_in_step()

3 Scenarios (3 passed)
11 Steps (11 passed)
0m3.952s
```

Test report

Test Case 2

Project Name: EduTech					
Messaging Test Case					
Test Case ID: Test_2			**Test Designed By: Vikhnesh Sathyan**		
Test Priority(Low/Medium/High):			**Test Designed Date: 22/10/24**		
Module Name: Message			**Test Executed By : Jinson Devis**		
Test Title : Message Send to students			**Test Execution Date: 22/10/24**		
Description:					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actua l Resul t**	**Status(Pass/ Fai l)**
1	Navigate to the teacher Dashboard	Make a message	Successfully send the message	Successful ly send	Pass
2	Navigate to the student dashboard	Message has been received	Successf ully message has been received	Successfully message has been received	Pass
3					
4					
5					
6					
7					
Post-Condition:					

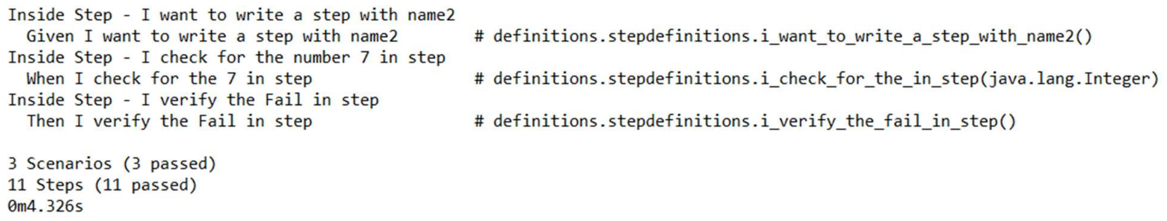
Test Case 3

Code

```
package definitions; // Ensure that this matches your package structure
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.cucumber.java.en.*;
public class stepdefinitions {
    WebDriver driver;
    // Step to open the browser
    @Given("browser is open")
    public void browser_is_open() {
        System.out.println("Inside Step - browser is open");
        // Set the path to your ChromeDriver or other browser driver
        System.setProperty("webdriver.chrome.driver",            "C:\\Users\\vikhn\\eclipse-
workspace\\2\\src\\test\\resources\\features\\chromedriver.exe");
        driver = new ChromeDriver();
        driver.manage().window().maximize();
    }
    // Step to navigate to the login page
    @And("user is on login page")
    public void user_is_on_login_page() {
        System.out.println("Inside Step - user is on login page");
        driver.navigate().to("http://localhost:4200/study-tips");
    }
    // Step to enter username and password
    @When("user enters username and password")
    public void user_enters_username_and_password() {
        System.out.println("Inside Step - user enters username and password");
        // Example locators for the username and password fields
        driver.findElement(By.id("classSelect")).sendKeys("class 9");
        driver.findElement(By.id("subject")).sendKeys("notes");
        driver.findElement(By.id("topic")).sendKeys("notes");
        driver.findElement(By.id("text")).sendKeys("notes");
    }
    // Step for clicking the login button
    @And("user clicks on login")
    public void user_clicks_on_login() {
        System.out.println("Inside Step - user clicks on login");
        // Example locator for the login button
        driver.findElement(By.id("tip")).click();
    }
    // Step to verify navigation to the home page
    @Then("user is navigate to the home page")
    public void user_is_navigate_to_the_home_page() {
        System.out.println("Inside Step - user is navigate to the home page");
        // Example verification step (you can add actual verification code here)
        String expectedUrl = "http://localhost:4200/tips";
        String actualUrl = driver.getCurrentUrl();
        if (actualUrl.equals(expectedUrl)) {
            System.out.println("Login successful, user is on home page.");
        } else {
            System.out.println("Login failed, user is not on home page.");
        }
        // Close the browser
        driver.quit();
    }
    // Step for scenario outline example
    @Given("I want to write a step with name1")
    public void i_want_to_write_a_step_with_name1() {
        System.out.println("Inside Step - I want to write a step with name1");
    }
    @When("I check for the {int} in step")
    public void i_check_for_the_in_step(Integer int1) {
        System.out.println("Inside Step - I check for the number " + int1 + " in step");
    }
```

```
    @Then("I verify the success in step")
    public void i_verify_the_success_in_step() {
        System.out.println("Inside Step - I verify the success in step");
    }
    @Given("I want to write a step with name2")
    public void i_want_to_write_a_step_with_name2() {
        System.out.println("Inside Step - I want to write a step with name2");
    }
    @Then("I verify the Fail in step")
    public void i_verify_the_fail_in_step() {
        System.out.println("Inside Step - I verify the Fail in step");
    }
}
```

Screenshot



```
Inside Step - I want to write a step with name2
  Given I want to write a step with name2       # definitions.stepdefinitions.i_want_to_write_a_step_with_name2()
Inside Step - I check for the number 7 in step
  When I check for the 7 in step                # definitions.stepdefinitions.i_check_for_the_in_step(java.lang.Integer)
Inside Step - I verify the Fail in step
  Then I verify the Fail in step                # definitions.stepdefinitions.i_verify_the_fail_in_step()

3 Scenarios (3 passed)
11 Steps (11 passed)
0m4.326s
```

Test Case 3

Project Name: EduTech	
Tips Test Case	
Test Case ID: Test_3	**Test Designed By:Vikhnesh Sathyan**
Test Priority(Low/Medium/High):	**Test Designed Date: 22/10/24**
Module Name: Tips	**Test Executed By : Jinson Devis**
Test Title : Tips	**Test Execution Date: 22/10/24**
Description:	
Pre-Condition : User has valid username and password	

Step	**Test Step**	**Test Data**	**Expected Result**	**Actua l Resul t**	**Status(Pass/ Fai l)**
1	Teacher Login	Login Success	Successfully Login	Success	Pass
2	Arrange a Tip with Subject	Successfull y Tip is arranged	Success	Success	Pass
3	Student Login	Successfully Login	Succes	Success	Pass
4	Tips section	Entered in the tip section	Success	Success	Pass
5					
6					
7					

Post-Condition:

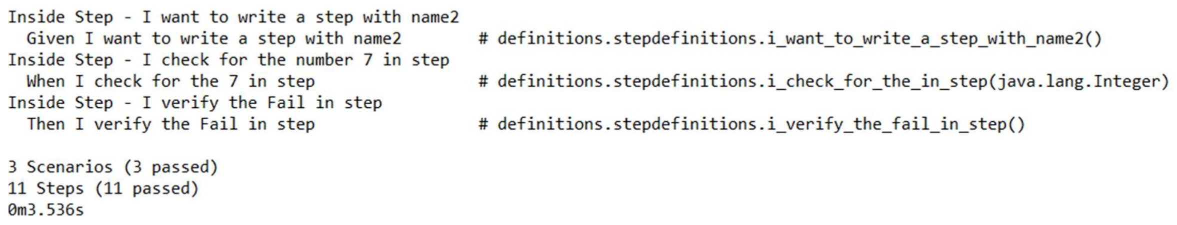
Test Case 4:

Code:

```
package definitions; // Ensure that this matches your package structure
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.cucumber.java.en.*;
public class stepdefinitions {
  WebDriver driver;
  // Step to open the browser
  @Given("browser is open")
  public void browser_is_open() {
    System.out.println("Inside Step - browser is open");
    // Set the path to your ChromeDriver or other browser driver
    System.setProperty("webdriver.chrome.driver",          "C:\\Users\\vikhn\\eclipse-
workspace\\2\\src\\test\\resources\\features\\chromedriver.exe");
    driver = new ChromeDriver();
    driver.manage().window().maximize();
  }
  // Step to navigate to the login page
  @And("user is on login page")
  public void user_is_on_login_page() {
    System.out.println("Inside Step - user is on login page");
    driver.navigate().to("http://localhost:4200/student-submission");
  // Step to enter username and password
  @When("user enters username and password")
  public void user_enters_username_and_password() {
    System.out.println("Inside Step - user enters username and password")
    // Example locators for the username and password fields
    driver.findElement(By.id("name")).sendKeys("anoop");
    driver.findElement(By.id("class")).sendKeys("9");
    driver.findElement(By.id("submission")).sendKeys("tcl commands");

  }
```

```
// Step for clicking the login button
@And("user clicks on login")
public void user_clicks_on_login() {
    System.out.println("Inside Step - user clicks on login");
    // Example locator for the login button
    driver.findElement(By.id("submit")).click();
}
// Step to verify navigation to the home page
@Then("user is navigate to the home page")
public void user_is_navigate_to_the_home_page() {
    System.out.println("Inside Step - user is navigate to the home page");
    // Example verification step (you can add actual verification code here)
    String expectedUrl = "http://localhost:4200/teacher-review";
    String actualUrl = driver.getCurrentUrl();
    if (actualUrl.equals(expectedUrl)) {
        System.out.println("Login successful, user is on home page.");
    } else {
        System.out.println("Login failed, user is not on home page.");
    }
    // Close the browser
    driver.quit();
}
// Step for scenario outline example
@Given("I want to write a step with name1")
public void i_want_to_write_a_step_with_name1() {
    System.out.println("Inside Step - I want to write a step with name1");
}
@When("I check for the {int} in step")
public void i_check_for_the_in_step(Integer int1) {
    System.out.println("Inside Step - I check for the number " + int1 + " in step");
}
@Then("I verify the success in step")
public void i_verify_the_success_in_step() {
    System.out.println("Inside Step - I verify the success in step");
    }
    @Given("I want to write a step with name2")
    public void i_want_to_write_a_step_with_name2() {
        System.out.println("Inside Step - I want to write a step with name2");
    }
    @Then("I verify the Fail in step")
    public void i_verify_the_fail_in_step() {
        System.out.println("Inside Step - I verify the Fail in step");
    }
}
```

Screenshot



```
Inside Step - I want to write a step with name2
  Given I want to write a step with name2       # definitions.stepdefinitions.i_want_to_write_a_step_with_name2()
Inside Step - I check for the number 7 in step
  When I check for the 7 in step                # definitions.stepdefinitions.i_check_for_the_in_step(java.lang.Integer)
Inside Step - I verify the Fail in step
  Then I verify the Fail in step                # definitions.stepdefinitions.i_verify_the_fail_in_step()

3 Scenarios (3 passed)
11 Steps (11 passed)
0m3.536s
```

Test report

Test Case 1

Project Name: EduTech					
Login Test Case					
Test Case ID: Test_4			**Test Designed By: Vikhnesh Sathyan**		
Test Priority(Low/Medium/High):			**Test Designed Date: 22/10/24**		
Module Name: Sending Questions			**Test Executed By : Jinson Devis**		
Test Title : Question Sending			**Test Execution Date: 22/10/24**		
Description:					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actua l Resul t**	**Status(Pass/ Fai l)**
1	Teacher Login	Successfully login	Success	Success	Pass
2	Teacher select a question for subject	Success	Success	Success	Pass
3	Student login	Success	Success	Success	Pass
4	Question section	Success	Success	Success	Pass
5					
6					
7					
Post-Condition:					

CHAPTER 6
IMPLEMENTATION

6.1INTRODUCTION

Implementation is the process of translating software designs and specifications into a functional software system. It involves the actual coding, testing, and integration of software components to create a working software product.

The implementation phase is a critical step in the software development life cycle (SDLC) because it involves turning the abstract ideas and plans into tangible software products that can be used by end-users. The implementation process involves several steps, including:

- Coding: Writing code in the programming language specified in the design phase.
- Testing: Ensuring that the code functions correctly and meets the design specifications. Testing can include unit testing, integration testing, and system testing.
- Debugging: Identifying and fixing any errors or defects in the code.
- Integration: Combining the individual software components into a cohesive system that functions as intended.
- Deployment: Installing the software on the target hardware and making it available to end-users.

During the implementation phase, it is important to follow coding and testing standards to ensure the software is reliable, maintainable, and scalable. It is also important to document the code and maintain version control to facilitate future maintenance and updates. Collaboration between developers and other stakeholders, such as testers, project managers, and end-users, can help to ensure that the software meets the needs and expectations of all stakeholders.

6.2 IMPLEMENTATION PROCEDURES

The implementation phase of the software development life cycle (SDLC) involves a series of procedures that must be followed to ensure that the software product is developed according to the design specifications and meets the quality and performance standards. Below are some of the typical procedures involved in the implementation phase:

- Testing: Testing is a critical part of the implementation phase, and it involves testing the software at different levels, including unit testing, integration testing, and system testing. Testers must develop test cases and test scripts based on the design specifications to ensure that the software functions correctly.
- Debugging: During testing, issues and bugs may be identified in the software code. Developers must identify and fix these errors or defects to ensure that the software functions as intended.
- Integration: After individual software components have been tested and debugged, they must be integrated into a cohesive system. The integration process must be carefully managed to ensure that the software functions as intended and that there are no conflicts or errors.

6.2.1 User Training

User training is a critical component of any software development project. It involves providing end-users with the necessary knowledge and skills to effectively use the software product. The goal of user training is to ensure that end-users can use the software product efficiently and effectively to meet their business needs.

Below are some common steps involved in user training:

- Identify training needs: Before training can begin, it is essential to identify the training needs of the end-users. This can be done by conducting a needs analysis, which involves assessing the current skills and knowledge of the end-users and identifying any gaps that need to be addressed.
- Develop training materials: Once the training needs have been identified, training materials can be developed. This can include user manuals, online tutorials, and training videos. The training materials should be designed to be easy to understand and follow, and should cover all the necessary features and functionalities of the software product.
- Conduct training sessions: Training sessions can be conducted in-person or online, depending on the needs of the end-users. The training sessions should be interactive

and engaging, and should provide opportunities for end-users to ask questions and practice using the software product.

6.2.2 Training on the Application Software

Training on application software is an essential component of software implementation. It is the process of providing end-users with the knowledge and skills necessary to operate and use the application software effectively. The goal of training is to ensure that end-users are able to utilize all the features and functionalities of the software to perform their tasks and achieve their goals.

Below are some best practices for providing effective training on application software:

Identify the target audience: Before developing training materials, it is essential to identify the target audience and their specific needs. Different groups of end-users may require different levels of training or different types of training materials.

Develop training materials: Training materials can include user manuals, online tutorials, videos, and live training sessions. The training materials should be designed to be easy to understand and follow, and should cover all the necessary features and functionalities of the software product.

6.2.3 System Maintenance

System maintenance is the process of keeping a software system up-to-date and functioning properly after it has been deployed. It involves a variety of activities that are designed to ensure that the system remains stable, secure, and efficient over time. Some of the key activities involved in system maintenance include:

- Updates and patches: One of the most important aspects of system maintenance is keeping the system up-to-date with the latest software updates and security patches. These updates and patches are released periodically by software vendors to fix bugs, add new features, and address security vulnerabilities. Applying these updates in a timely manner is essential to ensure that the system remains secure and reliable.
- Performance monitoring: Regular performance monitoring is another key aspect of system maintenance. This involves tracking system performance metrics such as CPU usage, memory utilization, and network traffic to identify potential issues and optimize system performance.
- Backup and recovery: Backing up system data and ensuring that it can be recovered in the event of a system failure or data loss is an important part of system maintenance. This involves creating regular backups of system data and testing the recovery process

to ensure that it works as expected.

6.2.4 Hosting

Hosting refers to the process of storing and serving website files on a remote server, which can be accessed by visitors over the internet. When you create a website, you need to have it hosted on a server so that it can be available for others to view. There are various types of hosting options available such as shared hosting, dedicated hosting, VPS hosting, cloud hosting, etc. The choice of hosting depends on the size of the website, its traffic volume, and the level of control and flexibility required by the owner

VERCEL

Vercel is a cloud platform that allows developers to easily deploy, host, and manage web applications. It specializes in hosting front-end frameworks and static sites, with a particular focus on Next.js, although it supports other frameworks like React, Vue, Svelte, and more. Vercel simplifies the deployment process by integrating with Git repositories, automatically building and deploying applications when code is pushed.

Procedure for hosting a website on 000Webhost:

Step 1: Create a Vercel Account

Step 2: Connect Your Project

Step 4: Configure Project Settings

Step 5: Deploy Your Project

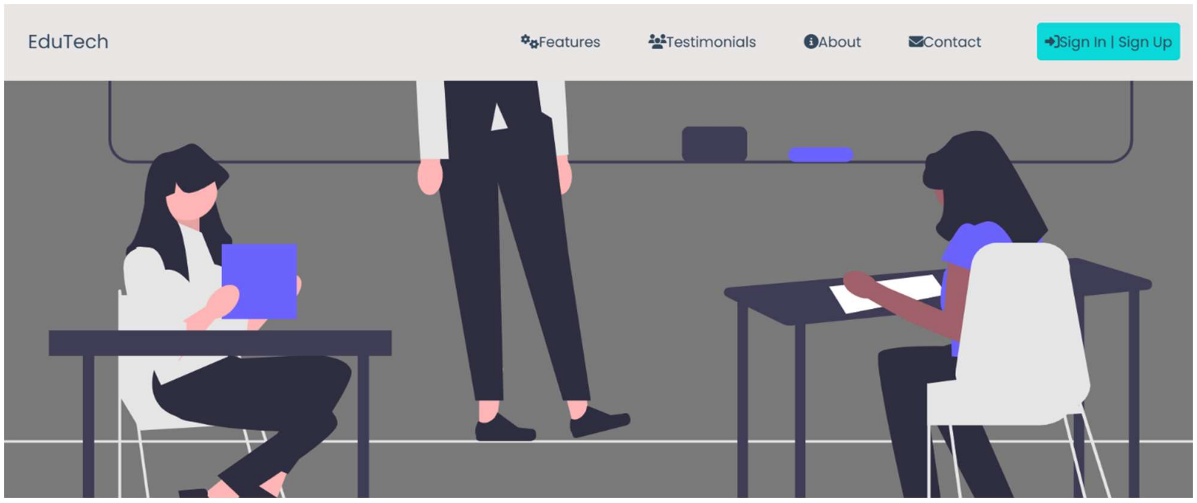
Step 6: Configure Automatic Deployments

Step 7: Manage Your Deployment

Hosted Website:

HostedLink:edu-tech-ummc.vercel.app

Screenshot



CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

Your project successfully demonstrates the integration of key features that enhance user experience in educational dashboards, catering to the needs of students, teachers, and parents. Through thoughtful design, intuitive navigation, and functionalities like assignment reminders, grade tracking, and student talent support, this platform fosters a supportive educational environment. The teacher dashboard, with real-time notifications, student performance metrics, and quick-access tools, provides educators with a streamlined way to monitor and support students. Likewise, the parent dashboard and other sections add value by promoting family engagement in the academic process. The consistent use of Angular, CSS, and innovative styling choices has given the application a professional look, making it both functional and visually engaging

7.2 FUTURE SCOPE

Advanced Analytics and Insights: Adding more in-depth analytics could allow teachers and parents to identify trends in student performance, engagement, and attendance. Machine learning algorithms can provide predictive insights to support students at risk of falling behind.

Gamification and Engagement Features: Incorporating gamified elements such as badges, levels, or rewards for completing assignments or achieving goals can enhance student motivation and make learning more interactive.

Mobile App Version: Developing a mobile application version would make the platform accessible on-the-go, allowing users to receive real-time notifications, access resources, and track progress anytime, anywhere.

Enhanced Communication Tools: Adding a message center that supports multimedia sharing and real-time chat could improve communication among students, teachers, and parents, especially for remote or hybrid learning environments.

CHAPTER 8
BIBLIOGRAPHY

REFERENCES:

- Angular - The Complete Guide by Maximilian Schwarzmüller (Udemy)
- Official Angular Documentation
- Official Node.js Documentation
- MEAN Stack Guide on Angular

WEBSITES:

- https://angular.io/
- https://www.coursera.org/courses?query=angular
- nodejs.org
- nodeschool.io

CHAPTER 9
APPENDIX

9.1 Sample Code

LOGIN-STUDENT

```
        <div class="container" id="main">
  <div class="sign-up">
    <form (ngSubmit)="onSubmit($event, 'signUp')" [formGroup]="signupForm">

      <h1>Student Sign Up</h1>
      <div class="social-container">
        <div id="google-btn"></div> <!-- Google Sign-In Button -->
      </div>
      <p>Or use your email to register</p>
      <! -- Username input field -->
      <input
        autocomplete="off"
        type="text"
        placeholder="Username"
        formControlName="username"
        aria-label="Username"
      >
      <!-- Error messages for username -->
      <div *ngIf="signupForm.get('username')?.invalid && (signupForm.get('username')?.dirty ||
signupForm.get('username')?.touched)">
        <small *ngIf="signupForm.get('username')?.errors?.['required']">Username is
required.</small>
        <small *ngIf="signupForm.get('username')?.errors?.['minlength']">Username must be at
least 3 characters long.</small>
        <small *ngIf="signupForm.get('username')?.errors?.['maxlength']">Username cannot exceed
15 characters.</small>
        <small *ngIf="signupForm.get('username')?.errors?.['pattern']">Username can only contain
letters, numbers, and underscores.</small>
        <small *ngIf="signupForm.get('username')?.errors?.['noSpaces']">Username must not
contain spaces.</small>
        <small *ngIf="signupForm.get('username')?.errors?.['startsWithSpecial']">Username must
not start with a special character.</small>
```

```
    <small *ngIf="signupForm.get('username')?.errors?.['usernameTaken']">Username is already
taken.</small>
   </div>
   <!-- Email input field for signup -->
   <input
    id="signupEmail"
    formControlName="email"
    type="email"
    placeholder="Email"
    aria-label="Email"
   >
   <div *ngIf="signupForm.get('email')?.invalid && (signupForm.get('email')?.touched ||
signupForm.get('email')?.dirty)">
    <div *ngIf="signupForm.get('email')?.errors?.['required']">Email is required.</div>
    <div *ngIf="signupForm.get('email')?.errors?.['email']">Invalid email format.</div>
    <div *ngIf="signupForm.get('email')?.errors?.['invalidDomain']">Email domain is not
allowed.</div>
   </div>
   <!-- Password input field -->
   <input
    autocomplete="off"
    type="password"
    placeholder="Password"
    formControlName="password"
    minlength="8"
    aria-label="Password"
   >
   <!-- Error messages for password -->
   <div *ngIf="signupForm.get('password')?.invalid && (signupForm.get('password')?.dirty ||
signupForm.get('password')?.touched)">
    <small *ngIf="signupForm.get('password')?.errors?.['required']">Password is
required.</small>
    <small *ngIf="signupForm.get('password')?.errors?.['minlength']">Password must be at least
8 characters long.</small>
```

```
      </div>
      <input
        autocomplete="off"
        type="password"
        placeholder="Confirm Password"
        formControlName="confirmPassword"
        required
        aria-label="Confirm Password"
      >
      <button type="submit" class="btn" [disabled]="signupForm.invalid">Sign Up</button>
    </form>
  </div>
  <div class="sign-in">
    <form [formGroup]="signinForm" (ngSubmit)="($event)">

      <h1>Sign In</h1>
      <div class="social-container">
        <div id="google-btn"></div> <!-- Google Sign-In Button -->
        <a href="#" class="social"><i class="fab fa-linkedin-in"></i></a>
      </div>
      <p>or use your account</p>
      <input
        id="signinEmail"
        formControlName="email"
        type="email"
        placeholder="Email"
        aria-label="Email"
      >
      <div *ngIf="signinForm.get('email')?.invalid && (signinForm.get('email')?.touched ||
signinForm.get('email')?.dirty)">
        <div *ngIf="signinForm.get('email')?.errors?.['required']">Email is required.</div>
        <div *ngIf="signinForm.get('email')?.errors?.['email']">Invalid email format.</div>
        <div *ngIf="signinForm.get('email')?.errors?.['invalidDomain']">Email domain is not
allowed.</div>
```

```
      </div>
      <input
        autocomplete="off"
        type="password"
        placeholder="Password"
        formControlName="password"
        minlength="8"
        aria-label="Password"
      >
      <div *ngIf="signinForm.get('password')?.invalid && (signinForm.get('password')?.dirty ||
signinForm.get('password')?.touched)">
        <small *ngIf="signinForm.get('password')?.errors?.['required']">Password is
required.</small>
        <small *ngIf="signinForm.get('password')?.errors?.['minlength']">Password must be at least
8 characters long.</small>
      </div>
      <a href="#">Forgot your password?</a>
      <button type="submit" class="btn">Sign In</button>
    </form>
  </div>
  <div class="overlay-container">
    <div class="overlay">
      <div class="overlay-left">
        <h1>Welcome!!</h1>
        <p>To our website</p>
        <button class="btn" (click)="togglePanel('signIn')">Sign In</button>
      </div>
      <div class="overlay-right">
        <h1>Hello</h1>
        <p>Enter your personal details and start your journey with us</p>
        <button class="btn" (click)="togglePanel('signUp')">Sign Up</button>
      </div>
    </div>
  </div>
```

```
</div>
```

CODE EDITOR

```
import { CommonModule } from '@angular/common';
import { HttpClientModule } from '@angular/common/http';
import { Component, ViewChild, ElementRef } from '@angular/core';
import { FormsModule } from '@angular/forms';
@Component({
  selector: 'app-code-editor',
  standalone: true,
  imports: [CommonModule, HttpClientModule, FormsModule],
  templateUrl: './code-editor.component.html',
  styleUrls: ['./code-editor.component.css'],
})
export class CodeEditorComponent {
  language: string = 'html'; // Default selected language
  code: string = '<h1>Hello, World!</h1>'; // Default code for demonstration
  errorMessage: string = ''; // Property for storing error messages
  @ViewChild('iframeOutput') iframeOutput!: ElementRef<HTMLIFrameElement>;
  // Method to run the code in the iframe
  runCode(): void {
    this.errorMessage = ''; // Clear previous error messages
    const iframe = this.iframeOutput.nativeElement;
    const iframeWindow = iframe.contentWindow || iframe.contentDocument?.defaultView;
    if (iframeWindow) {
      try {
        iframeWindow.document.open();
        iframeWindow.document.write(this.getCodeWithDoctype());
        iframeWindow.document.close();
      } catch (error: any) {  // Assert error as any
        this.errorMessage = 'Error executing code: ' + error.message; // Capture any errors
      }
    } else {
      this.errorMessage = 'Unable to access iframe window.'; // Handle iframe access error
    }
  }
  // Generates code with appropriate doctype
  private getCodeWithDoctype(): string {
    switch (this.language) {
      case 'html':
        return `<!DOCTYPE html><html><head><meta charset="UTF-
8"><title>Output</title></head><body>${this.code}</body></html>`;
      case 'css':
        return `<style>${this.code}</style>`;
      case 'javascript':
        return `<script>${this.code}</script>`;
      default:
        return '';
    }
  }
}
```

MESSAGES

```
import { Component } from '@angular/core';
import { MessagesService } from '../message.service'; // Adjust the path as necessary
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
@Component({
  selector: 'app-messages',
  standalone: true,
  imports: [CommonModule, FormsModule, HttpClientModule],
  templateUrl: './messages.component.html',
  styleUrls: ['./messages.component.css'], // Change 'styleUrl' to 'styleUrls'
  providers: [MessagesService]
})
export class MessagesComponent {
  messageContent: string = '';
  recipient: string = 'student';  // Replace with actual student ID
```

```
 sender: string = 'teacher';      // Replace with actual teacher ID


 constructor(private messagesService: MessagesService) {}
 sendMessage() {
  if (!this.messageContent.trim()) {
    console.error('Message content cannot be empty.');
    return;
  }
  this.messagesService.sendMessage(this.sender, this.recipient, this.messageContent).subscribe(
    (response: any) => {
      console.log('Message sent:', response);
      this.messageContent = ''; // Clear the input field
    },
    (error: any) => {
      console.error('Failed to send message:', error);
    }
  );
}
}
```

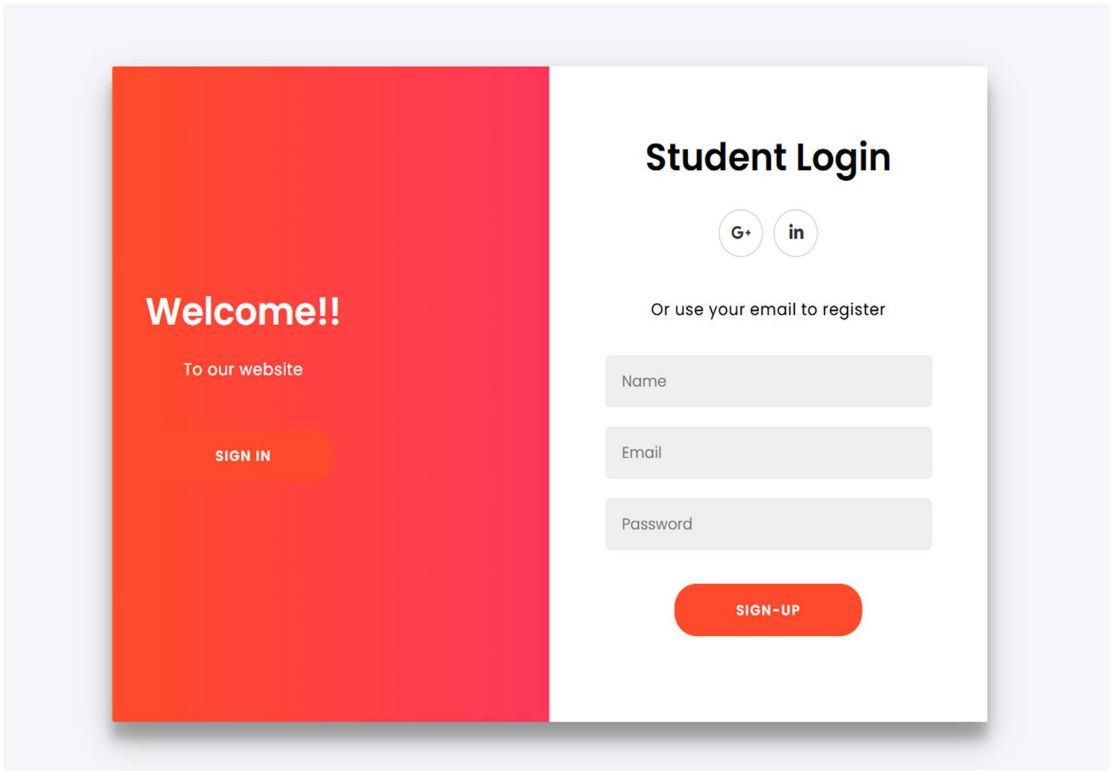
FLASH CARD

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';  // Import Router
import { FlashcardService } from '../flashcard.service'; // Adjust path if necessary
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { CommonModule } from '@angular/common';
import { RouterModule } from '@angular/router'; // Import RouterModule

@Component({
 selector: 'app-flashcard',
 standalone: true,
 imports: [CommonModule, FormsModule, HttpClientModule, RouterModule],  // Include
RouterModule
 providers: [FlashcardService],
 templateUrl: './flashcard.component.html',
 styleUrls: ['./flashcard.component.css']
})
export class FlashcardComponent {
 flashcard: any = { question: '', answer: '', tags: '', deck: '' };
```

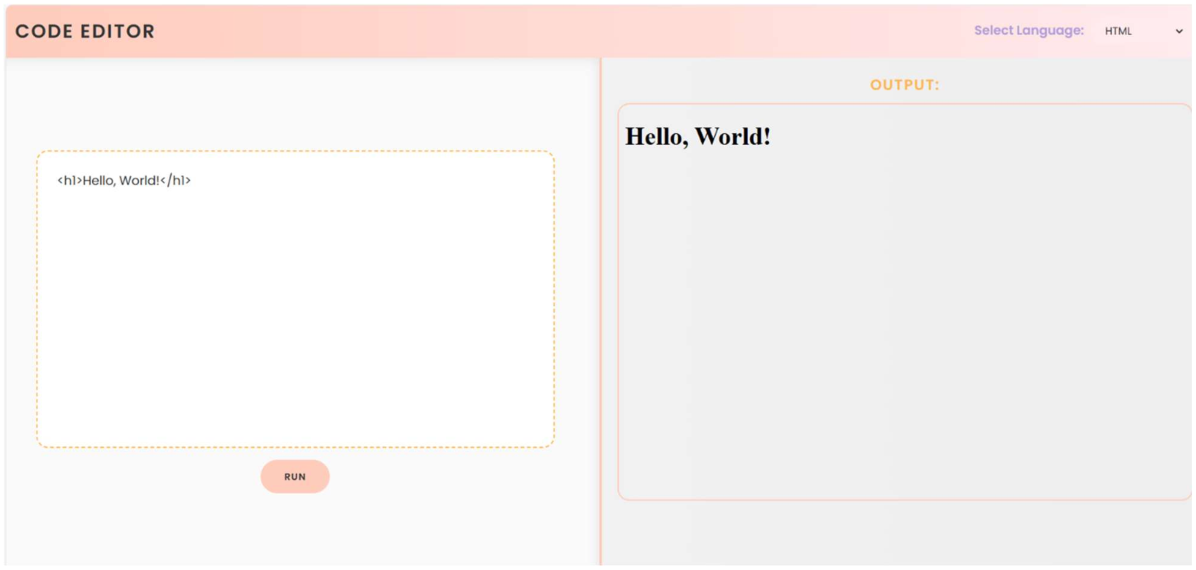
```
constructor(private router: Router) {}  // Inject Router

onSubmit(): void {
  let flashcards = JSON.parse(localStorage.getItem('flashcards') || '[]');
  flashcards.push(this.flashcard);
  localStorage.setItem('flashcards', JSON.stringify(flashcards));
  this.resetForm();
}

resetForm(): void {
  this.flashcard = { question: '', answer: '', tags: '', deck: '' };
}

goToFlashcardList(): void {
  this.router.navigate(['flashcard-list']);  // Navigate to flashcards list
}
```

9.2 Screen Shots

STUDENT-LOGIN



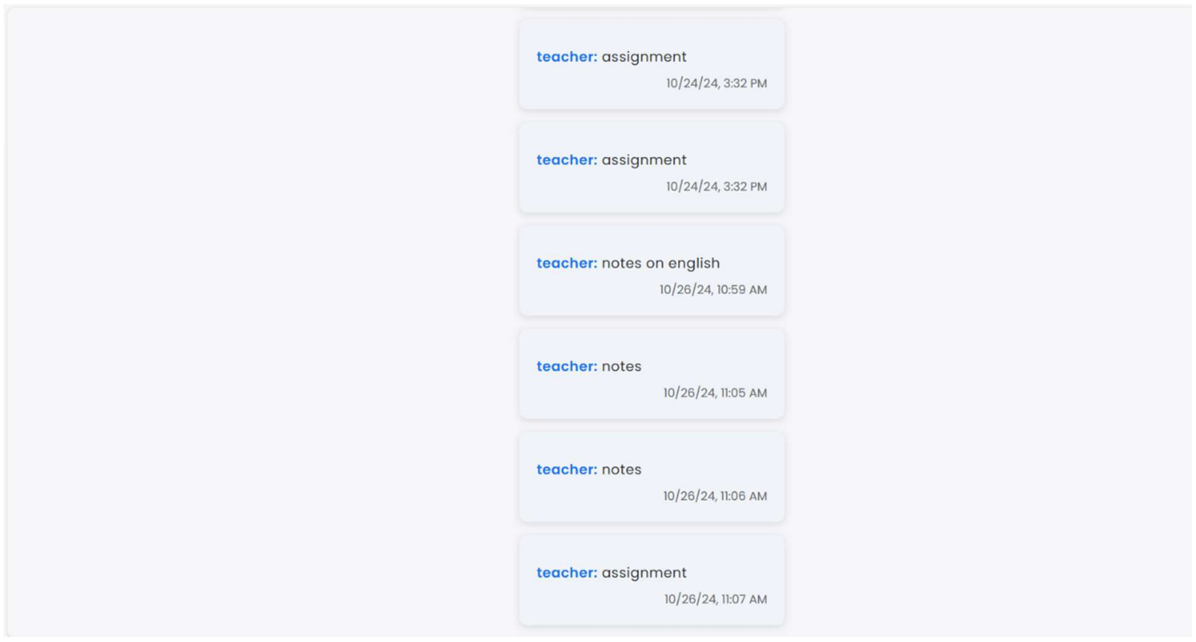
CODE-EDITOR



```
<h1>Hello, World!</h1>
```

```
Hello, World!
```

MESSAGE



teacher: assignment
10/24/24, 3:32 PM

teacher: assignment
10/24/24, 3:32 PM

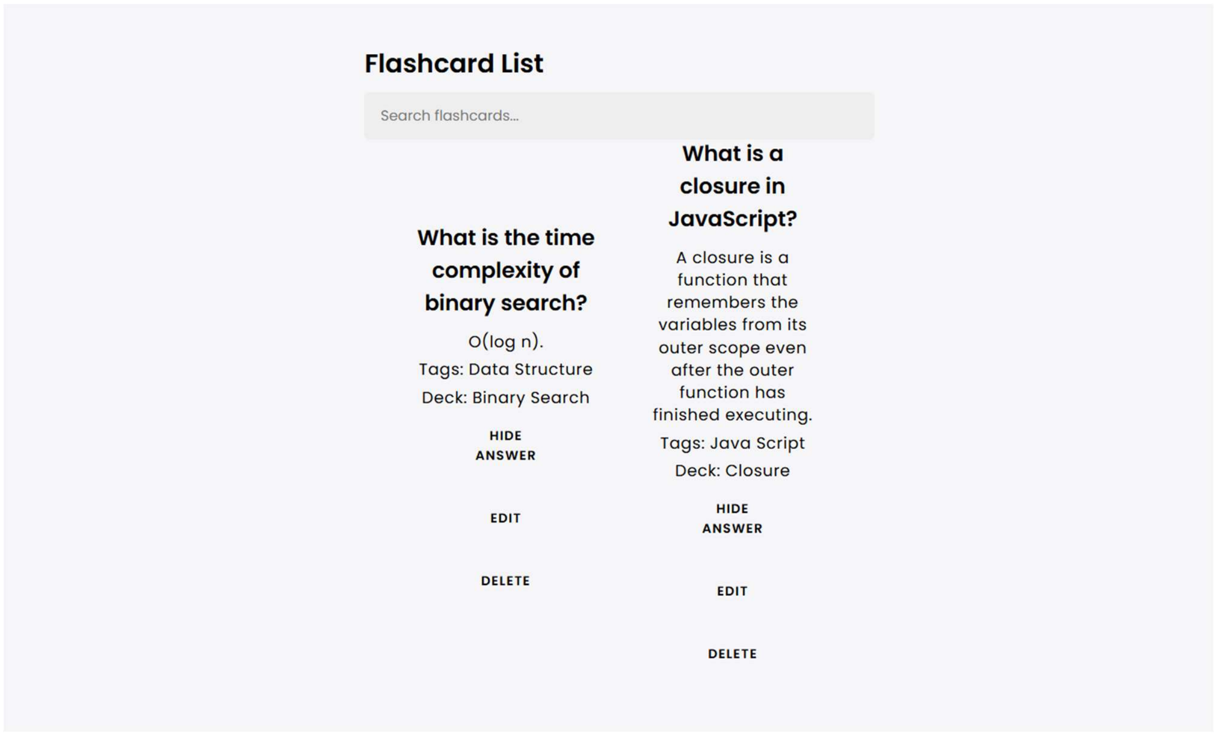
teacher: notes on english
10/26/24, 10:59 AM

teacher: notes
10/26/24, 11:05 AM

teacher: notes
10/26/24, 11:06 AM

teacher: assignment
10/26/24, 11:07 AM

FLASH-CARD



Flashcard List

Search flashcards...

What is the time complexity of binary search?

O(log n).

Tags: Data Structure

Deck: Binary Search

What is a closure in JavaScript?

A closure is a function that remembers the variables from its outer scope even after the outer function has finished executing.

Tags: Java Script

Deck: Closure