**Project Report**

**ON**

***E-learning Streaming Platform***

**BCA-5th Semester**

**(Session 2022-2025)**



**Submitted By:**

Name: **Anmol**

Inst. Roll No: **223067**

**Under Guidance of:**

Name: **Ms. Meenakshi Sharma**

Designation: **Assistant Professor**

Department: **Computer Applications**

**Maharaja Agrasen Institute of Management and Technology**

(Affiliated to Kurukshetra University, Kurukshetra & Approved by AICTE)

Old Saharanpur Road, Near Agrasen Chowk, Jagadhri-135003 (Haryana)

**INDEX**

1. **Introduction 1**
2. **Overall description 3**
3. **System features 5**
4. **External Interface Requirements 7**
5. **System Architecture and Design 8**
6. **Non-functional Requirements 18**

**7. Other requirements 19**

**Project Report**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to provide a detailed Software Requirements Specification (SRS) for the StudyBuddy e-learning platform. This document outlines the system's functional and non-functional requirements, use cases, and design constraints. It serves as a guide for developers, testers, and stakeholders to ensure that the system is built according to the defined requirements.

**1.2 Scope**

StudyBuddy is an e-learning platform designed to offer a variety of IT-related courses, both paid and free. It includes features for user registration, course management, video streaming, and payment processing. The platform is a web-based application developed using Node.js, Express.js, Sequelize, MySQL, JavaScript, HTML, and CSS.

**1.3 Definitions, Acronyms, and Abbreviations**

* **Node.js**: A JavaScript runtime environment that allows server-side scripting.
* **Express.js**: A web application framework for Node.js.
* **MySQL**: A relational database management system.
* **Sequelize**: A Node.js ORM for SQL-based databases.
* **HTML**: Hypertext Markup Language, used for creating web pages.
* **CSS**: Cascading Style Sheets, used for styling web pages.

**1.4 References**

* Node.js documentation: [Node.js Documentation](https://nodejs.org/en/docs/)
* Express.js documentation: [Express.js Documentation](https://expressjs.com/)
* MySQL documentation: [MySQL Documentation](https://dev.mysql.com/doc/)
* Sequelize documentation: [Sequelize Documentation](https://sequelize.org/master/manual/)

**1.5 Overview**

This SRS document is organized as follows:

* Section 2: Overall Description
* Section 3: System Features
* Section 4: External Interface Requirements
* Section 5: System Architecture and Design
* Section 6: Non-functional Requirements
* Section 7: Other Requirements

**2. Overall Description**

**2.1 Product Perspective**

StudyBuddy is a standalone web-based application that provides a platform for users to enroll in courses, watch educational videos, and obtain certificates. It will allow administrators to manage categories, courses (both free and paid), users, and video content efficiently.

**2.2 Product Functions**

The system provides the following functionalities:

* User authentication and authorization.
* Course management including adding and deleting courses (both free and paid).
* Video management, including uploading and streaming course videos.
* User enrollment in courses.
* Payment processing for paid courses.
* Review and rating system for courses.

**2.3 User Classes and Characteristics**

* **Administrator**: Has full access to all system features, including managing categories, courses (both free and paid).
* **User**: Can browse courses, enroll in courses, view course content, leave reviews.

**2.4 Operating Environment**

* **Server**: The system will be hosted on a server running Node.js and MySQL.
* **Client**: The system will be accessed via a web browser on devices with internet access.
* **Database**: MySQL will be used to manage and store all data.

**2.5 Design and Implementation Constraints**

* The system will be developed using Node.js, Express.js, Sequelize, MySQL, HTML, CSS, and JavaScript.

**2.6 Assumptions and Dependencies**

* Users will have a basic understanding of how to operate a web browser and interact with web applications.
* The system will rely on the availability of a MySQL database server.
* Internet connectivity is required for accessing the system.

**3. System Features**

**3.1 User Authentication**

**3.1.1 Description**  
Users must authenticate to access the system. Administrators have full access, while users have limited access.

**3.1.2 Functional Requirements**

* The system shall provide a login page where users can enter their credentials.
* The system shall validate user credentials against the database.
* The system shall restrict access to features based on user roles (Administrator, User).

**3.2 Course Management**

**3.2.1 Description**  
Administrators can manage courses, including adding and deleting both free and paid courses.

**3.2.2 Functional Requirements**

* The system shall allow administrators to add new courses (free or paid).
* The system shall allow administrators to remove courses.
* The system shall link each course to a specific category.
* The system shall allow administrators to set a course as free or paid.

**3.3 Video Management**

**3.3.1 Description**  
Administrators can manage videos associated with courses.

**3.3.2 Functional Requirements**

* The system shall allow administrators to upload videos for courses.
* The system shall allow administrators to delete videos from the system.

**3.4 Enrollment Management**

**3.4.1 Description**  
Users can enroll in available courses, whether free or paid.

**3.4.2 Functional Requirements**

* The system shall allow users to enroll in courses.
* The system shall record the enrollment in the database.
* The system shall provide users access to course content upon enrollment.
* The system shall allow users to access free courses without payment.

**3.5 Payment Processing**

**3.5.1 Description**  
The system will handle payments for paid courses.

**3.5.2 Functional Requirements**

* The system shall allow users to make payments for paid courses.
* The system shall track payment status (pending, completed, failed).
* The system shall grant access to the course upon successful payment.

**3.6 Review and Rating System**

**3.6.1 Description**  
Users can leave reviews and ratings for courses they have completed.

**3.6.2 Functional Requirements**

* The system shall allow users to submit ratings (1-5) for courses.
* The system shall allow users to write and submit reviews for courses.
* The system shall display ratings and reviews for each course.

**4. External Interface Requirements**

**4.1 User Interfaces**

* The system will have a web-based user interface accessible through a standard web browser.
* The UI will be responsive and support modern browsers like Chrome, Firefox, and Edge.

**4.2 Hardware Interfaces**

* The system will run on standard server hardware with sufficient resources to handle database operations and web requests.

**4.3 Software Interfaces**

* **Database**: The system will interact with a MySQL database for data storage and retrieval.
* **Payment Gateway**: The system will integrate with a payment gateway for processing payments.

**4.4 Communications Interfaces**

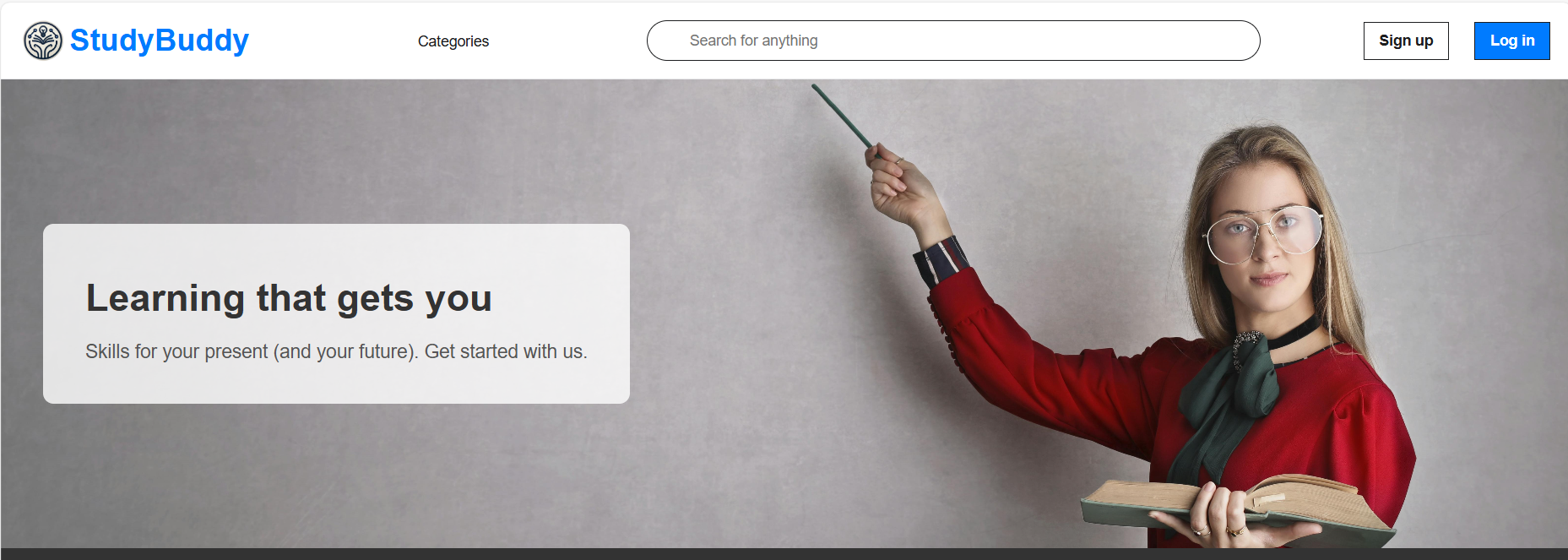
* The system will use HTTP/HTTPS for communication between the client and server.

**5. System Architecture and Design**

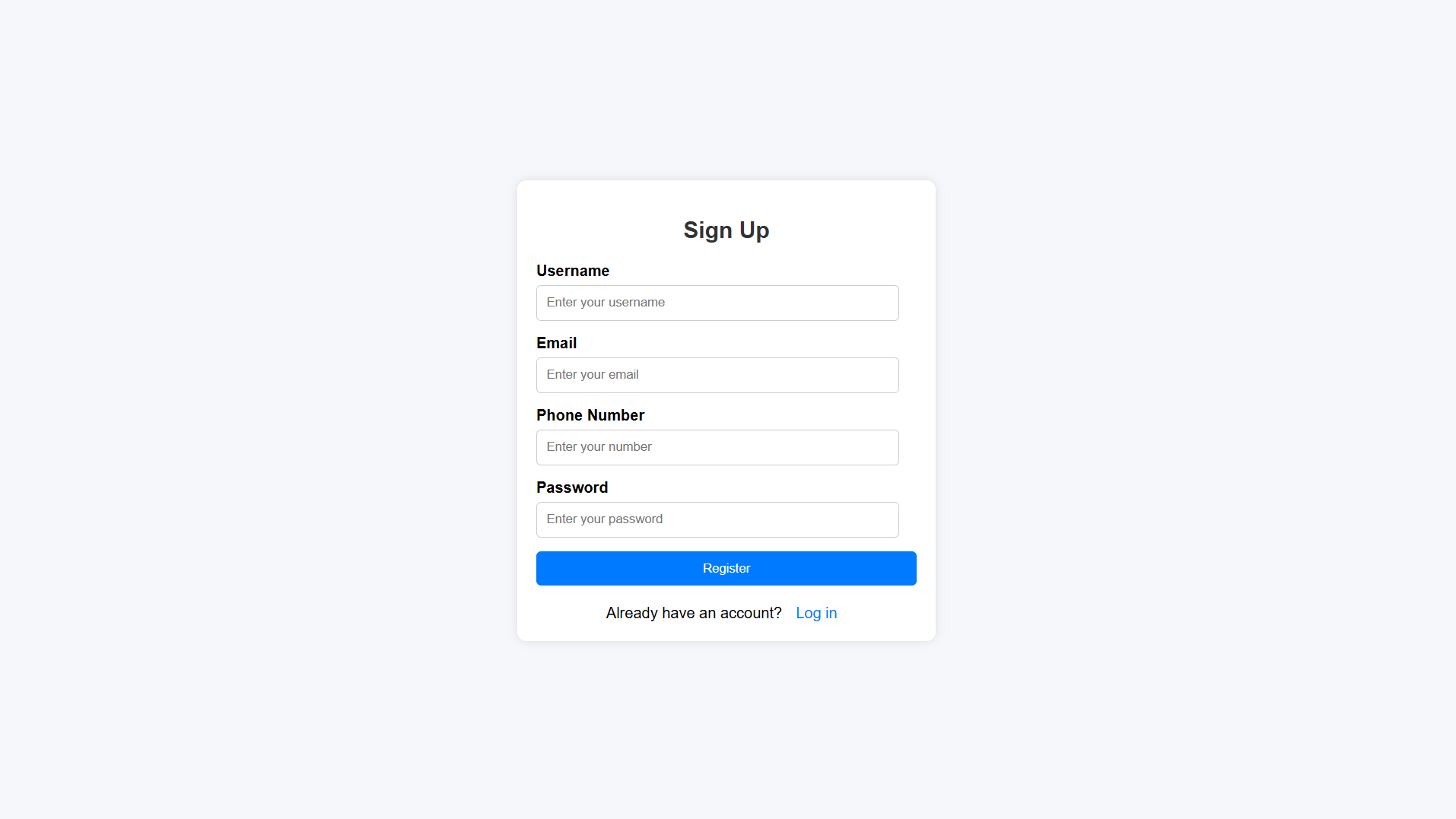
**5.1 System Architecture**

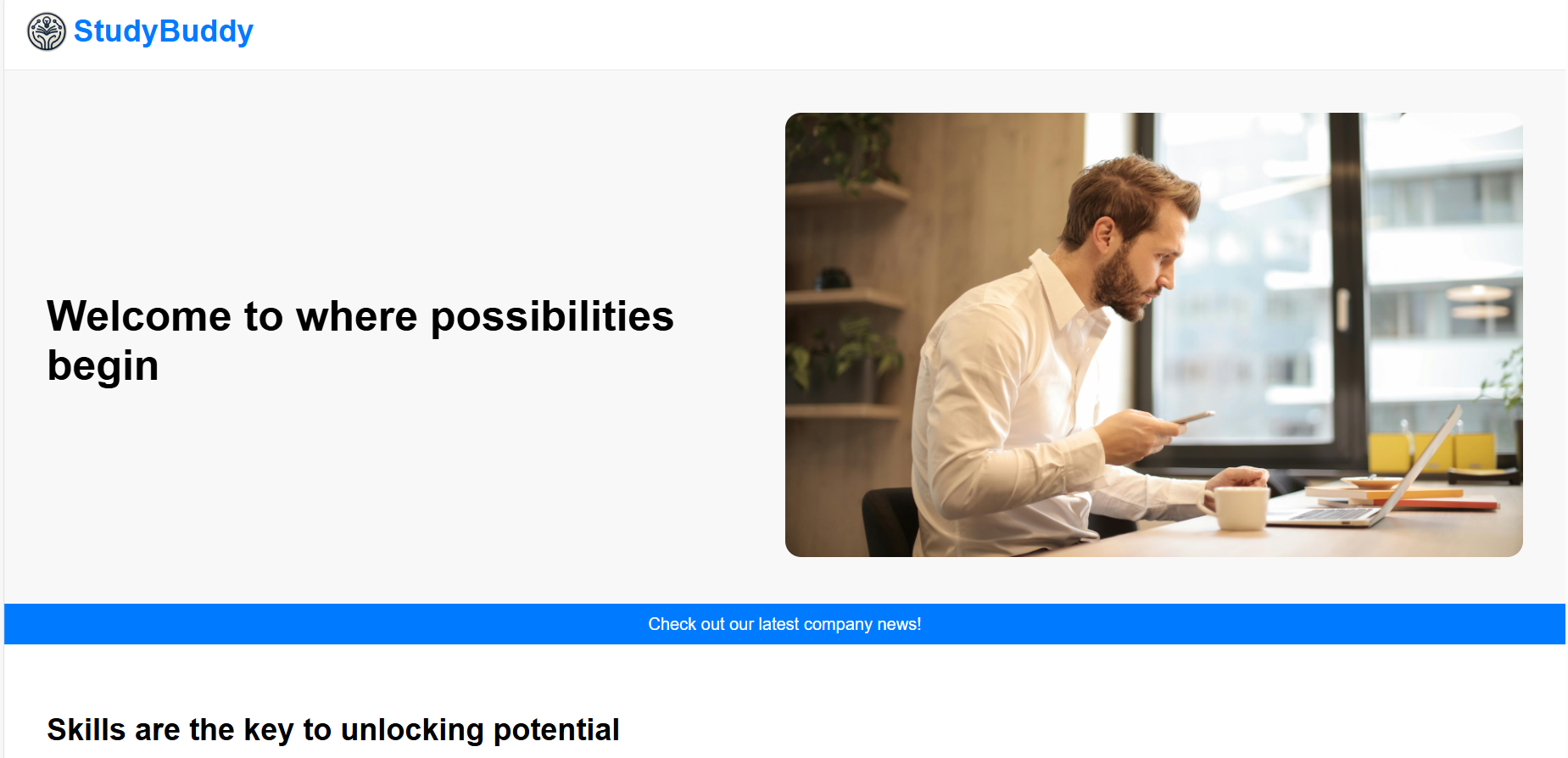
* The system will follow a client-server architecture.
* The backend will be built using Node.js and Express.js.
* The frontend will use HTML, CSS, and JavaScript for the user interface.
* Sequelize will be used to manage interactions with the MySQL database.

HOME PAGE:

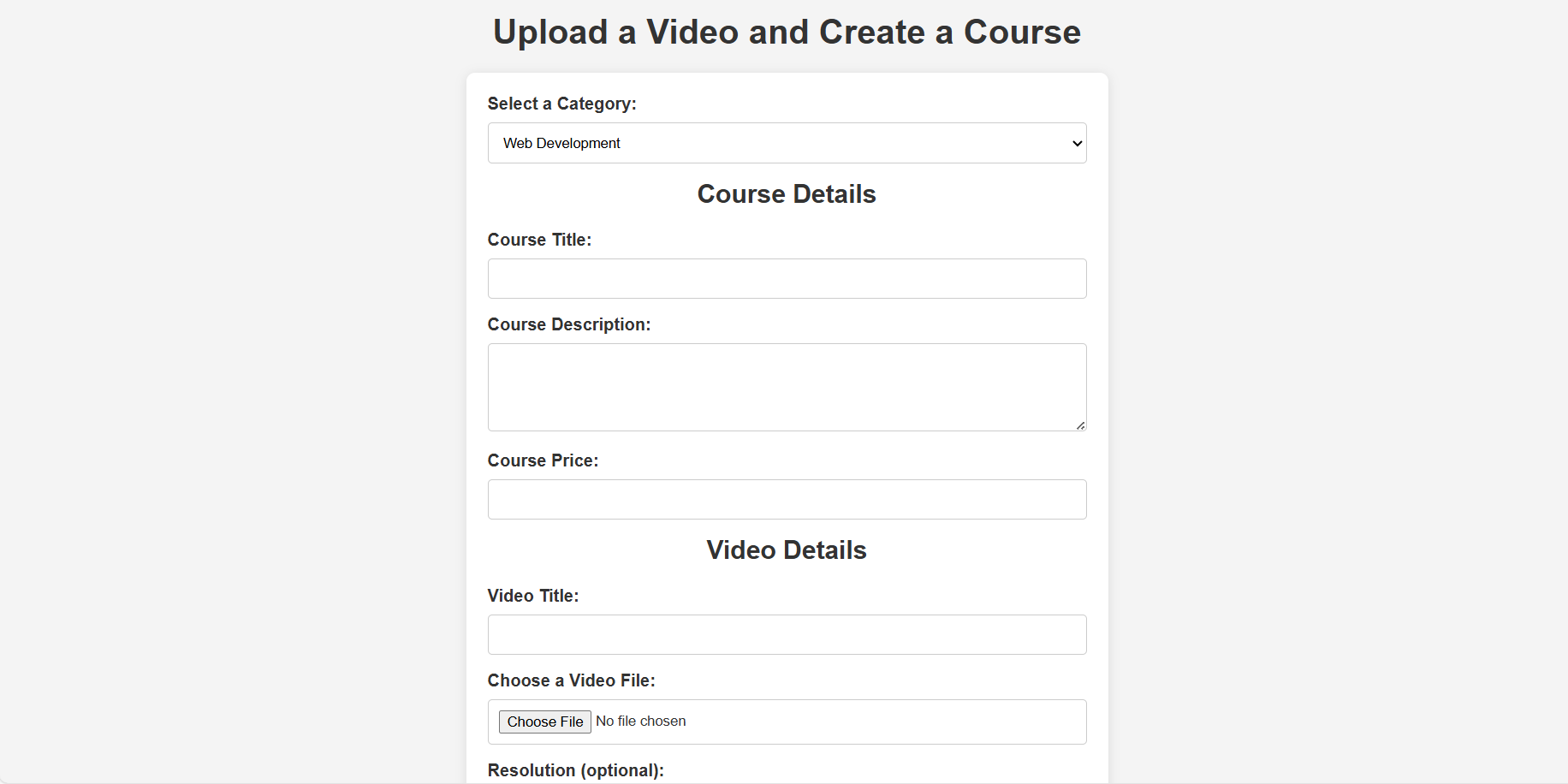


SIGN UP:



ABOUT US: 

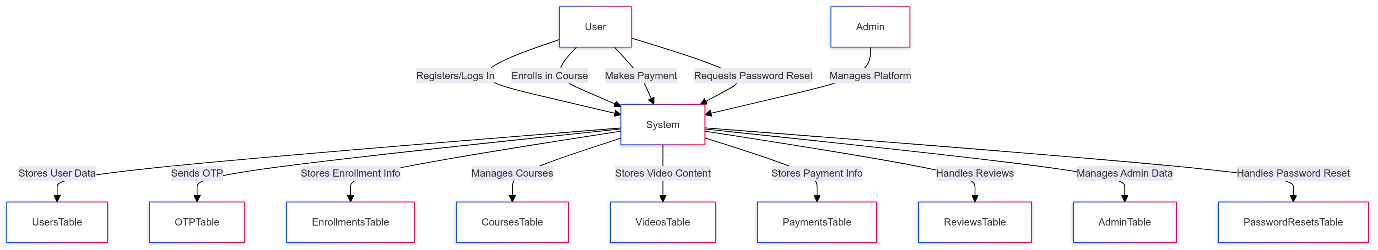
UPLOAD PAGE:



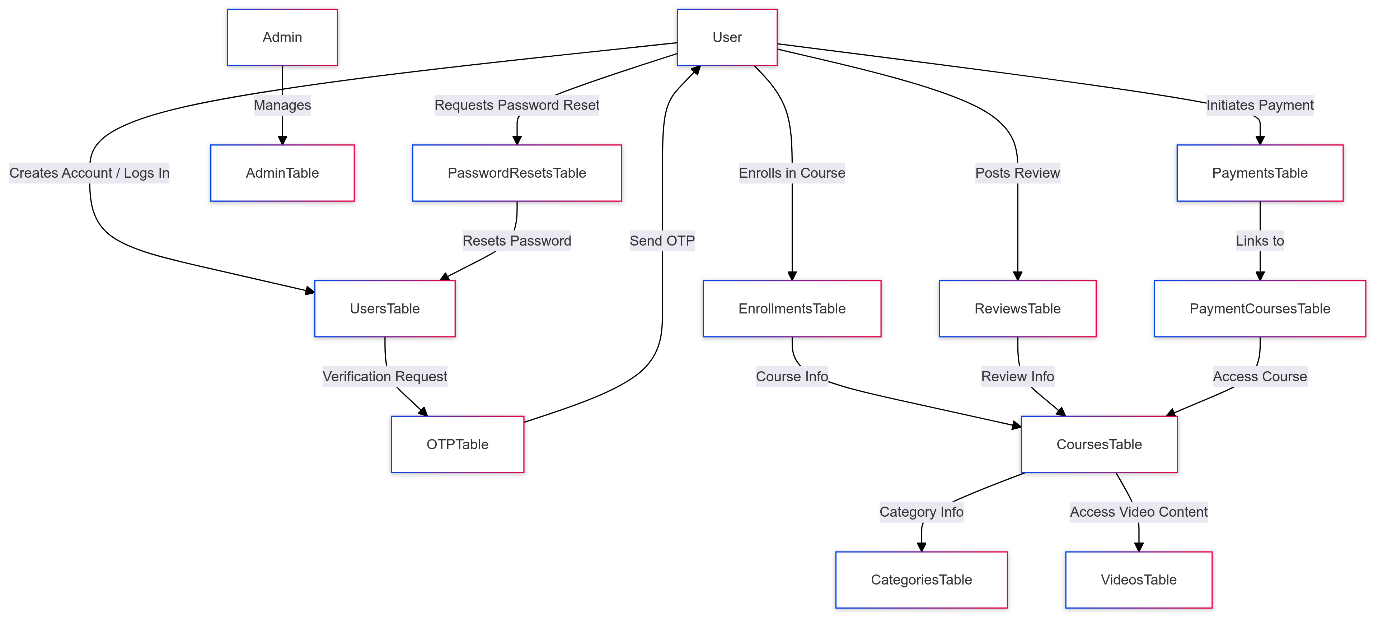
**5.2 Database Design**

**5.3 Data Flow Diagram**

Level – 0

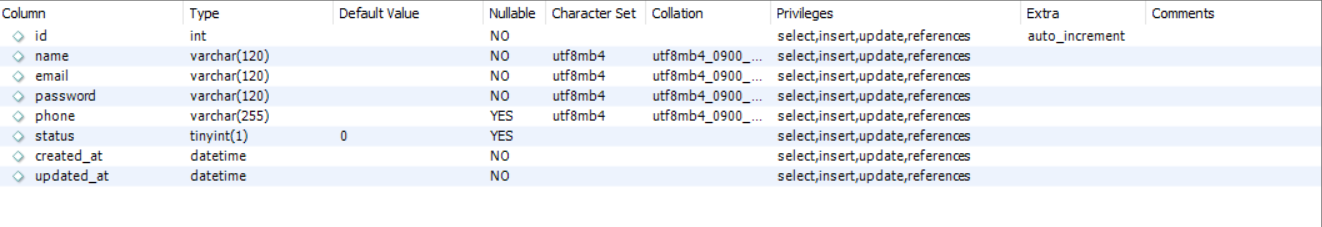


Level – 1

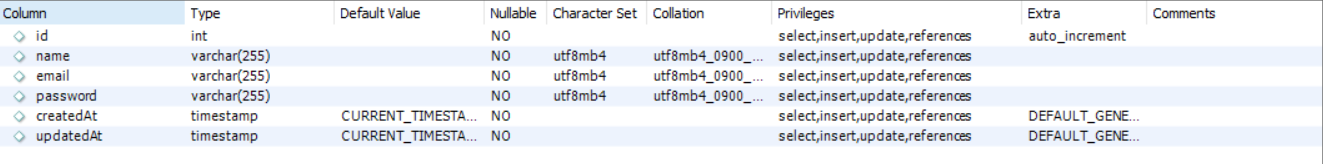
****

**5.4 Database Design**

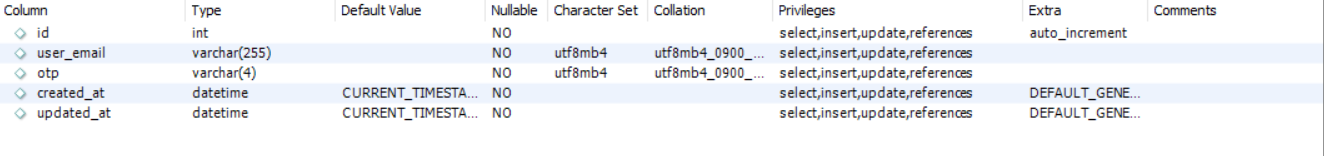
* **User table**

****

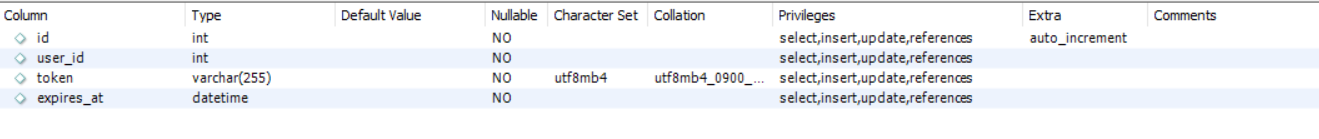
* **Admin table**

****

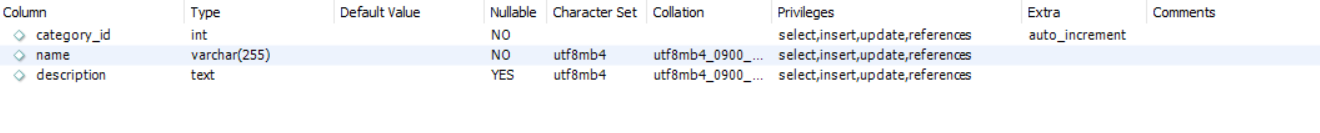
* **Otp table**

****

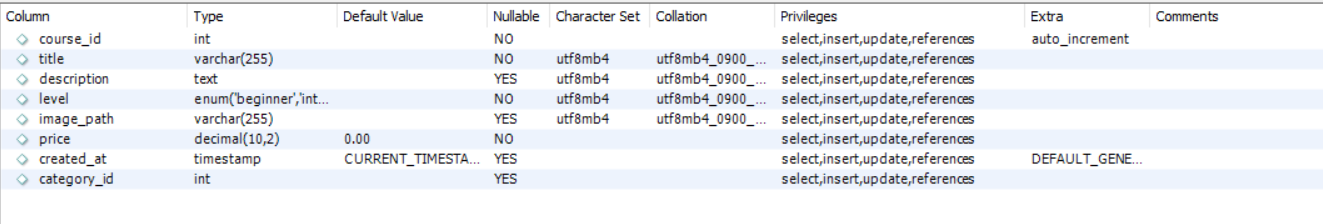
* **PasswordReset table**

****

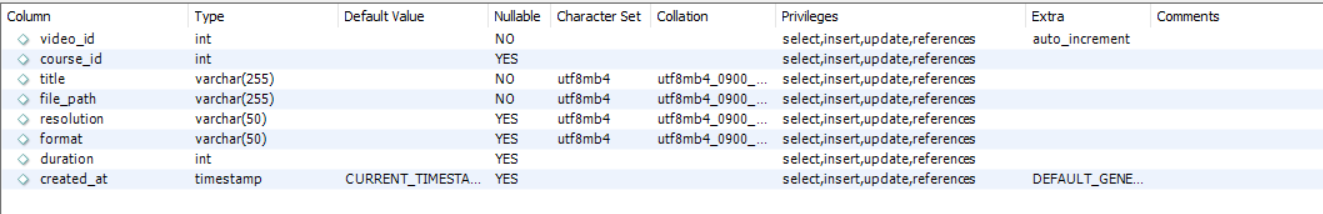
* **Category** **table**



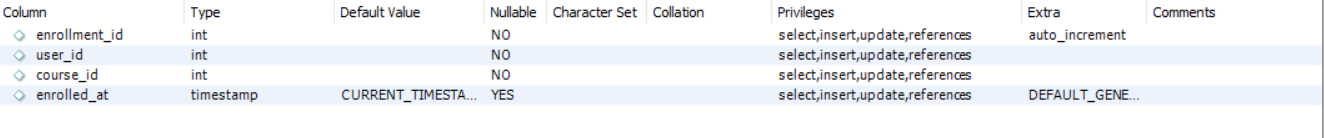
* **Courses** **table**



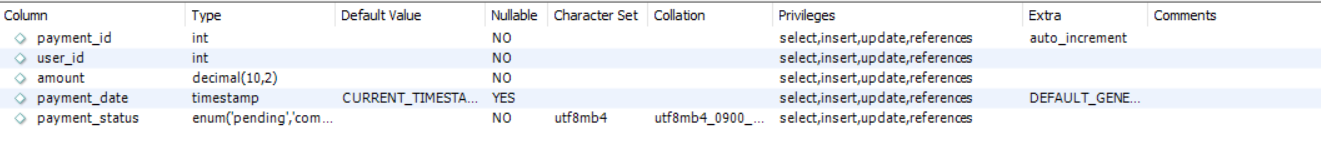
* **Video** **table**



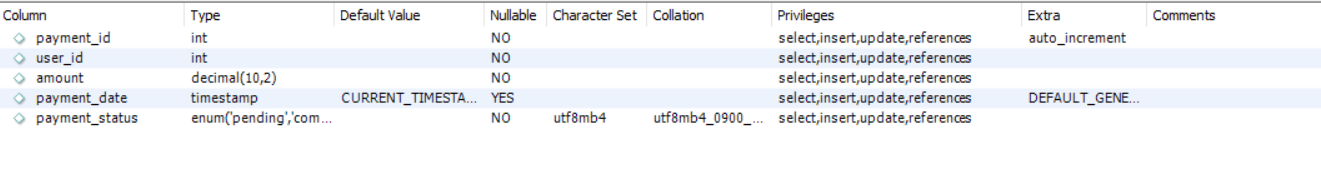
* **Enrollment** **table**



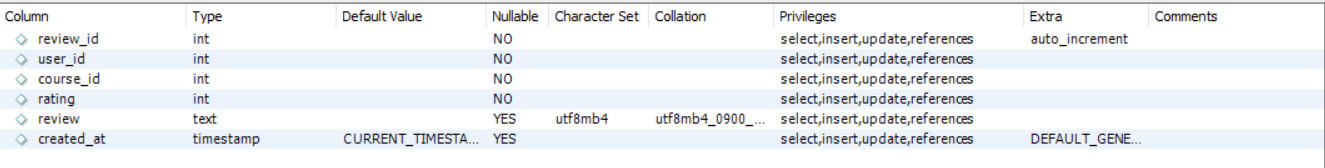
* **Payment** **table**



* **PaymentCourses table**



* **Review** **table**



**5.5 Security**

* User passwords will be hashed and stored securely in the database.
* Role-based access control will restrict access to certain features.
* HTTPS will be used for secure communication between client and server.

**5.5 TESTING**

**Features to be Tested:**

**This section outlines all the features that will be tested in the StudyBuddy e-learning platform:**

**Type of User**

**Administrator**

* Feature Identifier: Admin-1
  + Description: Add Course
* Feature Identifier: Admin-2
  + Description: Upload Video
* Feature Identifier: Admin-3
  + Description: Manage Categories

**User**

* Feature Identifier: User-1
  + Description: Enroll in Course
* Feature Identifier: User-2
  + Description: View Course Content
* Feature Identifier: User-3
  + Description: Submit Review and Rating

**CASE 1: Add Course (Admin-1)**

Purpose: Test that the admin can successfully add a new course with all required fields.

1. Navigate to the Add Course page.
2. Enter all required course details (Title, Description, Category, Price, etc.).
3. Click the "Add Course" button.

| S/N | Input Data | Expected Results | Actual Results | Pass/Fail | Remarks |
| --- | --- | --- | --- | --- | --- |
| 1 | Empty value for Course Title | Display error message "Course Title is required." | **✓** | Pass |  |
| 2 | Invalid price format | Display error message "Invalid price format." | **✓** | Pass |  |
| 3 | Correct data in all fields | Course is added and redirected to the course list. | **✓** | Pass |  |

**CASE 2: Upload Video (Admin-3)**

Purpose: Test that the admin can upload a video for a course.

1. Navigate to the Upload Video page.
2. Select a course and upload a video file.
3. Enter video details (Title, Duration, Format).
4. Click the "Upload Video" button.

| S/N | Input Data | Expected Results | **Actual Results** | **Pass/Fail** | **Remarks** |
| --- | --- | --- | --- | --- | --- |
| 1 | No video file selected | Display error message "Please select a video file." | **✓** | Pass |  |
| 2 | Invalid video format | Display error message "Invalid video format." | **✓** | Pass |  |
| 3 | Correct video file and details | Video is uploaded and linked to the selected course. | **✓** | Pass |  |

**CASE 3: Manage Categories (Admin-4)**

Purpose: Test that the admin can add, edit, or delete categories.

1. Navigate to the Manage Categories page.
2. Add a new category, edit an existing category, or delete a category.

| S/N | Input Data | Expected Results | Actual Results | Pass/Fail | Remarks |
| --- | --- | --- | --- | --- | --- |
| 1 | Enter an existing category name | Display error message "Category already exists." | **✓** | Pass |  |
| 2 | Enter a new category name | Category is added successfully. | **✓** | Pass |  |
| 3 | Select a category and delete it | Category is deleted from the system. | **✓** | Pass |  |

**CASE 4: Enroll in Course (User-1)**

Purpose: Test that a user can enroll in a course.

1. Log in as a user.
2. Browse available courses and select one to enroll in.
3. Click the "Enroll" button.

| S/N | Input Data | Expected Results | Actual Results | Pass/Fail | Remarks |
| --- | --- | --- | --- | --- | --- |
| 1 | Course not available | Display error message "Course is not available." | **✓** | Pass |  |
| 2 | Successfully enroll in course | Enrollment is recorded and user gains course access. | **✓** | Pass |  |

**CASE 5: View Course Content (User-2)**

Purpose: Test that a user can view course content.

1. Log in as a user.
2. Navigate to the enrolled course.
3. Access course materials.

| S/N | Input Data | Expected Results | Actual Results | Pass/Fail | Remarks |
| --- | --- | --- | --- | --- | --- |
| 1 | Attempt to access without enrollment | Display error message "You need to enroll first." | **✓** | Pass |  |
| 2 | Enrolled in the course | Course content is displayed and accessible. | **✓** | Pass |  |

**CASE 6: Submit Review and Rating (User-3)**

Purpose: Test that a user can submit a review and rating for a course.

1. Log in as a user.
2. Navigate to the course review section.
3. Submit a rating (1-5) and review text.

| S/N | Input Data | Expected Results | Actual Results | Pass/Fail | Remarks |
| --- | --- | --- | --- | --- | --- |
| 1 | No rating or review text | Display error message "Rating and review required." | **✓** | Pass |  |
| 2 | Correct rating and review | Review and rating are submitted and visible. | **✓** | Pass |  |

**6. Non-functional Requirements**

**6.1 Performance**

* The system should handle up to 500 concurrent users without significant degradation in performance.

**6.2 Reliability**

* The system should be available 99.9% of the time.

**6.3 Usability**

* The user interface should be intuitive and easy to use for both administrators and users.

**6.4 Maintainability**

* The system should be easy to maintain and update, with well-documented code.

**6.5 Portability**

* The system should be deployable on any server that supports Node.js, Express.js, Sequelize, and MySQL.

**7. Other Requirements**

**7.1 Legal and Regulatory Requirements**

* The system must comply with data protection regulations such as GDPR.

**7.2 Documentation**

* User manuals and technical documentation should be provided to help administrators and developers.