LEARNFINITY

MAJOR PROJECT-II

(20SSP65)

Submitted By

MOHANA G

20MSS026

Under the Guidance of

Dr. G. S. KARTHICK

Assistant Professor

Department of Software Systems

In Partial Fulfillment of the Requirements for the Award of the Degree of

MASTER OF SCIENCE IN SOFTWARE SYSTEMS

(Five-Years Integrated Programme)

of Bharathiar University

DEPARTMENT OF SOFTWARE SYSTEMS

PSG COLLEGE OF ARTS & SCIENCE

College with Potential for Excellence (Status Awarded by the UGC)

An Autonomous College - Affiliated to Bharathiar University

Accredited with 'A++'Grade by NAAC (4th Cycle)

Star College Status Awarded by DBT - MST

An ISO 9001:2015 Certified Institution

Coimbatore - 641 014

APRIL 2025

DEPARTMENT OF SOFTWARE SYSTEMS

PSG COLLEGE OF ARTS & SCIENCE

College with Potential for Excellence (Status Awarded by the UGC)

An Autonomous College - Affiliated to Bharathiar University

Accredited with 'A++'Grade by NAAC (4th Cycle)

Star College Status Awarded by DBT - MST

An ISO 9001:2015 Certified Institution

Coimbatore - 641 014

CERTIFICATE

This is to certify that this project work entitled **"LEARNFINITY"** is a bonafide record of work done by **MOHANA G (20MSS026)** in partial fulfillment of the requirements for the award of Degree of **Master of Science in Software Systems** (Five years Integrated Course) of Bharathiar University.

_____ _____

Faculty Guide **Head of the Department**

Submitted for Viva-Voce Examination held on _____

_____ _____

Internal Examiner **External Examiner**

DECLARATION

I, **MOHANA G (20MSS026)**, hereby declare that this project work entitled **"LEARNFINITY"** is submitted to PSG College of Arts &Science, Coimbatore in partial fulfillment of the requirements for the award of the degree of Master of Science in Software Systems, is a record of original work done by me under the supervision and guidance of **Dr. G. S. KARTHICK**, Assistant Professor, Department of Software Systems, PSG College of Arts &Science, Coimbatore.

This report has not been submitted by me for the award of any other Degree/ Diploma/ Associate ship/ Fellowship or any other similar degree to any other university.

PLACE: Coimbatore **MOHANA G**

DATE : **(20MSS026)**

ACKNOWLEDGEMENT

My venture stands imperfect without dedicating my gratitude to a few people who have contributed a lot towards the victorious completion for my project work.

I would like to thank **Shri L. Gopalakrishnan, Managing Trustee, PSG & Sons Charities**, for providing me prospect and surroundings that made the work possible.

I take this opportunity to express my deep sense of gratitude to our Secretary, **Dr. T. Kannaian, M.Tech., Ph.D.,** PSG College of Arts & Science, Coimbatore for permitting and doing the needful towards the successful completion of this project.

I express my deep sense of gratitude and sincere thanks to our Principal i/c **Dr. M. Senguttuvan, M.Sc., M.Phil., B.Ed., Ph.D.**, for his valuable advice and concern on students.

I am very thankful **Dr. M. Umarani, M.B.A., M.Phil., Ph.D.**, Vice Principal (SF), **Dr. M. Jayanthi M.Com., MBA., M.Phil., Ph.D.,** Vice Principal (Aided), for her support towards my project.

I sincerely thank **Dr. K. V. Rukmani., MCA., M.E., NET., Ph.D.,** Head of the Department, Department of Software Systems for her whole hearted help to complete this project successfully by giving valuable suggestions.

I convey my heartiest and passionate sense of thankfulness to my project guide **Dr. G. S. Karthick,** Assistant Professor, Department of Software Systems, for his timely suggestion which enabled me in completing the project successfully.

I am grateful to **Arul Frances Santhanam, B.E., Project Lead (DX)**, **Digital Transformation**, External Guide, Automation Department at **Hirotec India Private Limited** for giving me an expert's insight into the project and for sparing time in his hectic agenda, for planning and giving me invaluable guidance and suggestions

This note of acknowledgement will be incomplete without paying my heartful devotion to my parents, my friends and other people, for their blessings, encouragement, financial support and the patience, without which it would have been impossible for me to complete the job.

MOHANA G

20MSS026

INTERNSHIP OFFER LETTER

HIROTEC INDIA Private Limited
No.7/147, Powerhouse Road, Keeranatham P.O., Saravanampatti, Coimbatore – 641 035, INDIA
Phone: 91 (422) 666 6777
E-mail:info@hirotecindia.com Website: www.hirotecindia.com CIN – U72200TZ2005PTC011837

Dec 13, 2024

To

The Department Head
Department of Software Systems
PSG College of Arts & Science
Coimbatore - 641 014.

Respected Sir,

Sub: Internship Confirmation letter

This is to inform that **Ms. Mohana G (20MSS026),** student of your institute pursuing her M.Sc Software Systems has been confirmed to do Internship at HIROTEC India Private Limited from 9th Dec 2024 to 31st Mar 2025.

Thanking You,

Kavitha S
General Manager - HR & Admin

Branch
Office No. 401 & 402, Tated Horizon - 4th Floor
S.No. 41/2/1A/1, Akurdi, Haveli, PCMC,
Pune-411035

INTERNSHIP COMPLETION LETTER

HIROTEC India Private Limited

No. 7/147, Powerhouse Road, Keeranatham Post, Saravanampatti, Coimbatore - 641 035. INDIA
Phone : 91 (422) 666 6777
E-mail : info@hirotecindia.com Website : www.hirotecindia.com CIN- U72200TZ2005PTC011837

Apr 03, 2025

CERTIFICATE

This is to certify that **Ms. Mohana G** of PSG College of Arts & Science has undergone Internship training in Automation department in our company from Dec 09, 2024 to Mar 31, 2025.

During this period her code and conduct was good.

For Hirotec India Private Limited

Kavitha S
GM- HR & Administration

HIROTEC INDIA/HRD/PI/2025/4

Branch Office :
Office NO - 401 & 402, Tated Horizon - 4th Floor
S.No. 41/2/1A/1, Akurdi, Haveli, PCMC,
Pune - 411035. INDIA

SYNOPSIS

The project entitled as "**Learnfinity**" is developed as a web application for an advanced e-learning platform, designed to offer a seamless and secure learning experience.

The objective of this project is to develop Learnfinity, a MERN-based e-learning platform that ensures secure, role-based access while providing an interactive and scalable learning experience with real-time updates and JWT authentication.

The system incorporates role-based access control for students, teachers, and administrators, ensuring structured user interactions. Students can enroll in courses, take quizzes, track their progress, and earn completion certificates. Teachers can create and manage courses, quizzes, and lessons, while administrators oversee platform management, user roles, and course approvals.

Learnfinity is designed for educational institutions, online course providers, and independent instructors to facilitate interactive digital learning. The platform supports multimedia lessons, assessments, and performance tracking, making education accessible and engaging.

The web application is built using React.js for the frontend, Node.js and Express.js for the backend, and MongoDB as the database. This combination ensures a modern UI design, scalable backend infrastructure, and secure data management, resulting in efficient functionality and seamless user interactions.

TABLE OF CONTENTS

CHAPTER - 1

INTRODUCTION

1.1 COMPANY PROFILE

HIROTEC is a global automotive industry leader specializing in the production of body-in-white closures, exhaust systems, and closure manufacturing equipment. We deliver complete turn-key solution and we offer a fully-integrated production system that supports our customers from product design, tool development, through mass production. The flawless execution of our Full Vertical Approach enables us to achieve short vehicle development timeframes with exceptional quality.

Hirotec team has many decades of production experience, directly contributing to their ability to deliver stamping dies that consistently produce high quality detail panels. Individual high quality detail panels that produce the best gap and flush/fit and finish conditions which contribute to improved vehicle styling. Stamping dies that combine HIROTEC's global expertise with the use of our low-cost supply base, resulting in the highest quality at the lowest possible cost.

Automation Team delivers time saving and efficient cost saving solutions to reduce Manual efforts and to automate repetitive tasks. We are specialized in CAD Automation, Software, Hardware Automation, Internet of Things (IoT), Cloud, Mobile Apps, Web Apps and digitalization. We do lots of R&D and Proof of Concepts on modern technologies.

COMPANY DETAILS

Address: 7/147, Power House Road, Keeranatham, Coimbatore, India 641035

Email: sales@hirotecindia.com

T: +91 422 666 6777

1.2 PROJECT OVERVIEW

Learnfinity is an advanced e-learning web application designed to offer a seamless, secure, and interactive learning experience. The platform ensures structured role-based access for students, teachers, and administrators, optimizing course management and user interactions.

The core objective of Learnfinity is to develop a MERN-based e-learning platform that integrates real-time updates and JWT authentication, ensuring a scalable and engaging educational environment.

- Students can enroll in courses, watch course videos, participate in quizzes, track progress, provide reviews, and earn completion certificates. They can also edit their personal details anytime.
- Teachers can create, edit, and manage courses, add, update, or delete chapters, lessons, and quizzes. They can also update their personal details whenever needed.
- Administrators oversee platform operations, including blocking/deleting users, approving/rejecting teachers, creating/deleting courses, and sending emails to students based on their enrolled courses.

Learnfinity leverages modern web technologies to ensure security, efficiency, and accessibility. Real-time updates enhance user engagement, while JWT authentication safeguards data and user interactions. The platform's structured role management optimizes educational workflows, making it an ideal solution for educational institutions, online course providers, and independent instructors.

The system is built using the MERN stack, with React.js for the frontend, Node.js and Express.js for the backend, and MongoDB for data storage. This technology stack ensures a scalable, responsive, and dynamic user experience. The integration of secure authentication, real-time updates, and a comprehensive course management system makes Learnfinity a powerful and modern e-learning solution.

1.3 MODULE DESCRIPTION

Modules are the building blocks of a software project. The modules are separate part of a program which has many functions which are used all over the project. The modules work together to run a software. The modules make the programmer to write less code for a repetitive function and the updates can be added easily since the modules are separate part of a program.

The modules in this project are:

1. Home Module

- The main landing page providing access to Courses, About, FAQ, Signup, and Sign In.
- Serves as the entry point for all users.

2. Course Module

- Allows users to browse available courses and view detailed information.
- Students can enroll in courses, study lessons, attempt quizzes, and earn completion certificates.

3. Authentication Module

- **3.1 Sign Up**: Enables students to register, while teachers register through a separate signup page.
- **3.2 Sign In**: Supports login for students, teachers, and administrators, ensuring secure access based on user roles.
- **3.3 Logout**: Allows users to sign out, securely ending their session.

4. Student Module

4.1 Student Dashboard

- Provides an overview of enrolled courses.
- Enables students to explore and enroll in new courses.

4.2 Course Interaction

- Students can study course materials, watch course videos, attempt quizzes, and earn completion certificates.

4.3 Student Profile Management

- Students can edit and update their personal details anytime.

5. Teacher Module

5.1 Teacher Dashboard

- Provides teachers with a course management interface.
- Allows teachers to create, edit, and delete courses.
- Supports adding, updating, and deleting chapters, lessons, and quizzes.
- Enables teachers to view and manage their courses.

5.2 Teacher Profile Management

- Teachers can edit and update their personal details whenever needed.

6. Admin Module

6.1 Admin Dashboard

- Administrators can manage users, including blocking and deleting students.
- Admins have the authority to approve or decline teacher registration requests.
- They can create and delete courses, ensuring a well-maintained platform.

6.2 Admin Email Module

- Admins can send notifications or emails to students based on their enrolled courses.

CHAPTER - 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The existing system lacks proper role-based access and secure authentication for different user types (students, teachers, and admins). Course and user management tasks are handled manually, making the process inefficient and time-consuming. Additionally, there are limited interactive features, restricting effective communication between users.

DISADVANTAGES OF EXISTING SYSTEM

- Managing courses, users, and content is complex and unstructured.
- No proper authentication system, increasing security risks.
- Manual course approvals and user management slow down operations.
- Limited student-teacher interaction, reducing engagement.

2.2 PROPOSED SYSTEM

The proposed system, Learnfinity, is a MERN-based e-learning platform that provides structured role-based access for students, teachers, and admins. It enhances security, automation, and user engagement through modern web technologies.

ADVANTAGES OF PROPOSED SYSTEM

- Implements JWT-based authentication for secure role-specific access.
- Personalized dashboards for students, teachers, and admins to manage courses, quizzes, and lessons.
- Automated course approvals and user management streamline administrative tasks.
- Admins can send email notifications to students for better communication.
- Interactive features like course comments, quizzes, progress tracking, and certificate downloads improve user engagement.

CHAPTER - 3

SYSTEM CONFIGURATION

3.1 HARDWARE SPECIFICATION

COMPONENTS	MINIMUM REQUIREMENTS
PROCESSOR	: Intel core i3
RAM	: 4 GB
STORAGE	: 256 SSD

3.2 SOFTWARE SPECIFICATION

COMPONENTS	MINIMUM REQUIREMENTS
OPERATION SYSTEM	: Windows 10 or above
BROWSER	: Google Chrome or Microsoft Edge
FRONT END	: React.js
BACK END	: Node.js, Express.js
DATABASE	: MongoDB

CHAPTER - 4

SOFTWARE DESCRIPTION

4.1 FRONT END

REACT.JS

React.js is a powerful JavaScript library used for building dynamic and responsive user interfaces. It enables efficient rendering and seamless user interactions through a component-based architecture.

- Component-based architecture for reusable and maintainable UI elements.
- Virtual DOM for efficient rendering and improved performance.
- State management using React hooks or Redux for dynamic data handling.
- SEO-friendly with server-side rendering (SSR) capabilities using Next.js.

HYPERTEXT MARKUP LANGUAGE

HTML provides the fundamental structure for web pages, defining the elements and layout of content.

- Consists of various elements that define headings, paragraphs, links, images, and other web components.
- Ensures clear and structured content presentation for web browsers.
- Works in conjunction with CSS and JavaScript for interactive and dynamic web applications.
- Standard markup language for structuring web content.
- Semantic elements enhance accessibility and SEO.
- Supports media embedding, forms, and cross-browser compatibility.

CASCADING STYLE SHEETS

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media

- CSS stands for Cascading Style Sheets

- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External stylesheets are stored in CSS files

CSS has the potential of handling the design of all web pages at once. It enables the developers to make sure that style elements are applied consistently across the web pages. The accessibility of the website becomes more while using CSS.

EMAILJS

EmailJS is a third-party service that enables the application to send automated emails without requiring a backend email server. It facilitates seamless communication for notifications, password resets, and course updates, ensuring an efficient user experience.

- Allows automated email delivery without a backend mail server.
- Supports notifications, password resets, and course-related updates.
- Connects directly with email providers for quick and reliable email transmission.

4.2 BACK END

JAVASCRIPT

JavaScript is a versatile programming language used for creating interactive and dynamic web pages. It runs directly in the browser, enabling real-time updates, user interaction, and seamless integration with web APIs. Its event-driven nature and extensive ecosystem make it essential for modern web development.

- Executes in the browser for dynamic web interactions.
- Supports cross-browser compatibility and responsive UI.
- Enables DOM manipulation for real-time content updates.
- Event-driven and asynchronous for smooth user experiences.
- Integrates with web APIs and external services.

NODE.JS

Node.js is a server-side runtime environment for JavaScript. It allows you to run JavaScript code on the server, separate from the user's browser environment. Node.js is known for its event-driven, non-blocking I/O model, which makes it efficient at handling concurrent operations.

In the Node.js environment, you can create web servers, APIs, and various backend applications. It excels in real-time and scalable applications, making it ideal for tasks like processing user requests, interacting with databases, and integrating with external services.

Node.js leverages the Node Package Manager (NPM), providing access to a vast ecosystem of open-source packages and libraries for building server-side applications. It's cross-platform, compatible with different operating systems, and suitable for a wide range of applications beyond web development.

EXPRESS.JS

Express.js is a minimal and flexible Node.js web framework designed to simplify backend development. It provides powerful routing capabilities, middleware integration, and an intuitive API structure for building scalable web applications and RESTful APIs.

- A lightweight and flexible web framework for Node.js that simplifies backend development.
- Provides a robust routing mechanism for handling HTTP requests and responses efficiently.
- Supports middleware for implementing authentication, logging, error handling, and data processing.
- Ideal for building RESTful APIs and dynamic web applications.
- Offers a minimalistic yet extensible structure, allowing developers to integrate additional modules as needed.
- Enhances code maintainability and development speed with its modular architecture.

MONGO DB

MongoDB is a widely used NoSQL database that stores data in a flexible, JSON-like format, making it highly scalable and efficient for handling large datasets. It seamlessly integrates with Mongoose, an Object Data Modeling (ODM) library, to enable schema-based data structuring and validation, ensuring efficient database management.

- Uses a document-based structure for flexible data storage.
- Handles large datasets efficiently with horizontal scaling.
- Provides schema-based modeling for structured data management.
- Supports Create, Read, Update, and Delete for seamless data handling.
- Optimized indexing for quick access to stored information.

JWT AUTHENTICATION

JSON Web Token (JWT) is implemented for secure authentication and authorization across different user roles (students, teachers, and admins). It ensures safe login mechanisms and access control by assigning role-based permissions.

- Ensures authentication by generating secure tokens for users.
- Assigns role-based permissions to control access levels.
- Tokens verify users during API requests, preventing unauthorized access.

CHAPTER - 5

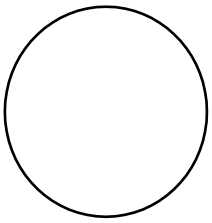
SYSTEM DESIGN

5.1 DATA FLOW DIAGRAM

A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design). On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process. The purpose of a DFD is:

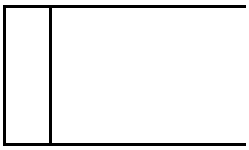
- To show the scope and boundaries of a system
- To show that the whole system has been considered
- May be used as a communications tool between a systems analyst and any person who plays a part in the system
- To act as the starting point for redesigning a system.

The representations used in order to frame a data flow diagram are:

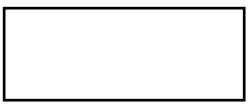


The circle represents the process. It shows the pair of the process that transforms input to output.

The arrow represents the graphical flow into or out of a process.



The store is used to model a collection of data packets at rest.



The terminator represents external entities with system Communication.

Level 0

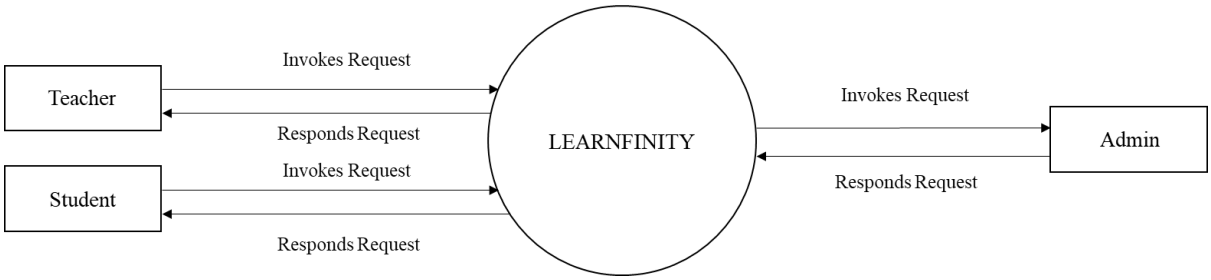


Figure 5.1.1 Level 0 DFD

Level 1 - Admin

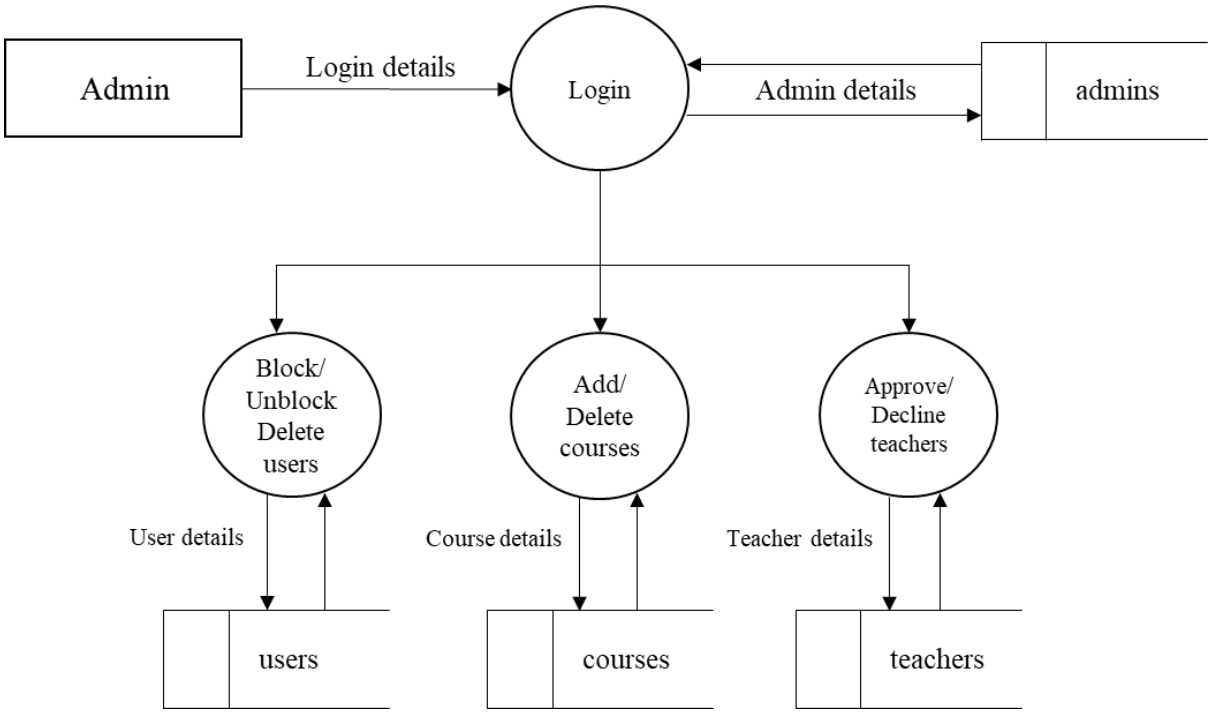


Figure 5.1.2 Level 1 DFD

Level 1 - Teacher

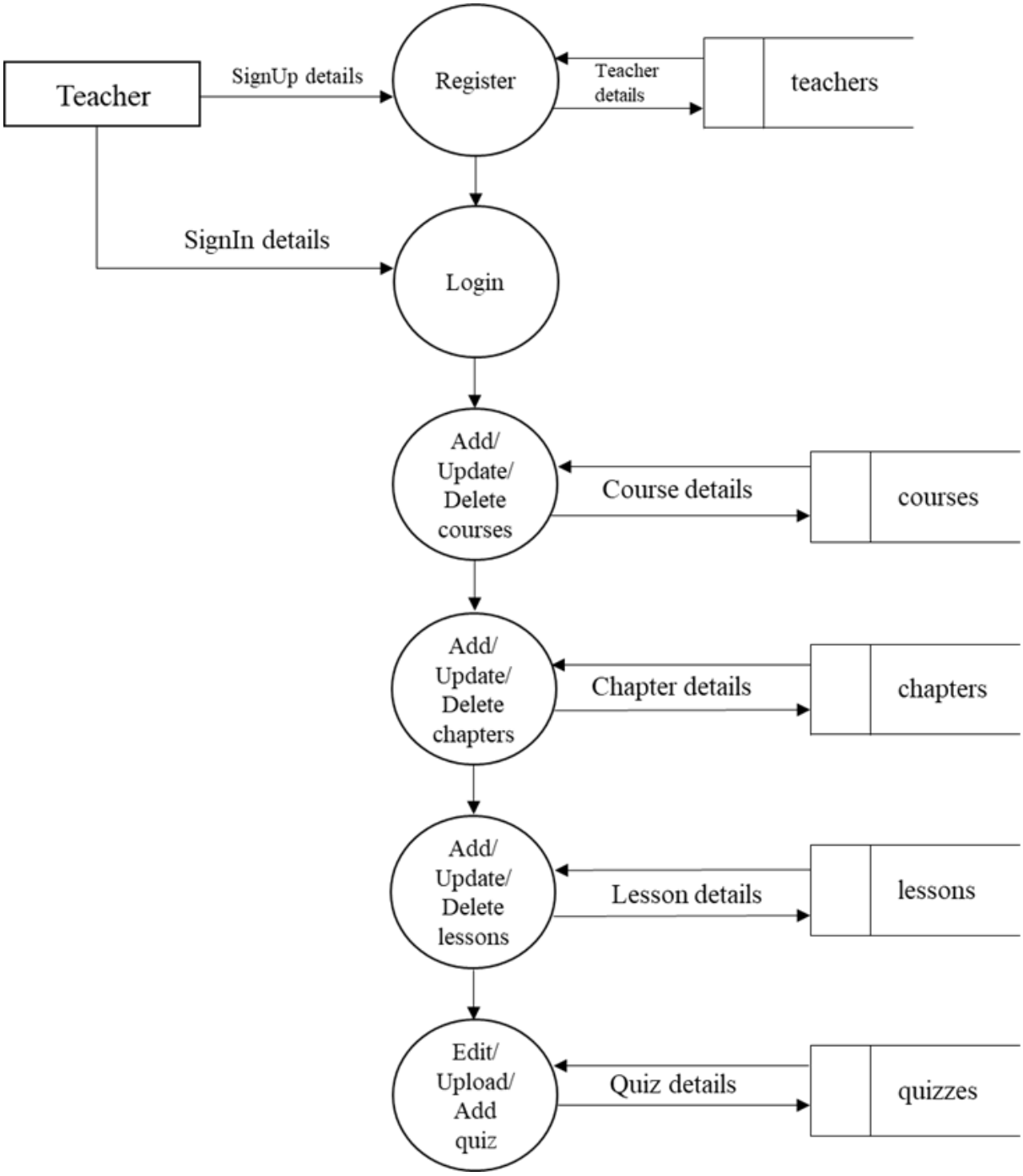


Figure 5.1.3 Level 1 DFD

Level 1 - Student

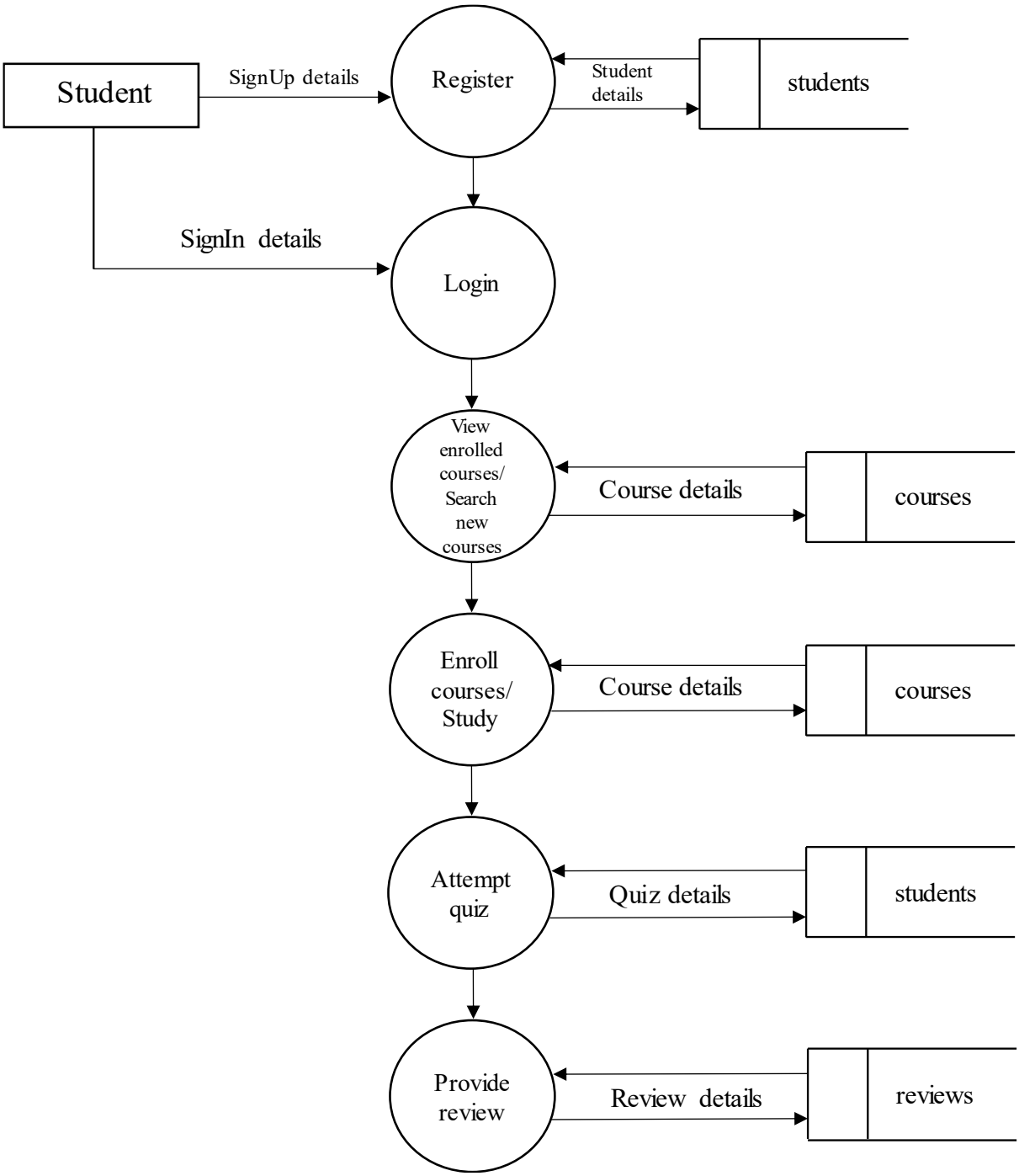


Figure 5.1.4 Level 1 DFD

5.2 COLLECTION DESIGN

In MongoDB Atlas, effective collection design is crucial for optimizing database performance, ensuring scalability, and maintaining efficient data access. A well-structured collection design aligns with application requirements, minimizes redundancy, and enhances query efficiency. Learnfinity's database is structured using multiple collections to manage users, courses, lessons, quizzes, and interactions efficiently. Each collection is designed with appropriate data types, relationships, and indexing strategies to provide a seamless learning experience.

The following sections outline the collection structure for Learnfinity, detailing the schema, field types, and their descriptions.

5.2.1 USER COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the user
email	String	User's email (must be unique)
password	String	Hashed password for security
Imageurl	String	Optional profile image URL
student	ObjectId	Reference to the Student schema (if applicable)
teacher	ObjectId	Reference to the Teacher schema (if applicable)
admin	ObjectId	Reference to the Admin schema (if applicable)
role	Number	User role: 0 = User (default), 1 = Admin, 2 = Teacher, 3 = Student
flag	Number	Status flag: 0 = Inactive, 1 = Active
createdAt	Date	Timestamp for user creation
updatedAt	Date	Timestamp for last update

5.2.2 ADMIN COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the admin record
fullName	String	Admin’s full name
phoneNo	String	Contact number of the admin
email	String	Admin's email address for login and contact
password	String	Hashed password for authentication
role	String	User role, default is "admin"
createdAt	Date	Timestamp of when the admin account was created
updatedAt	Date	Timestamp of when the record was last updated

5.2.3 TEACHER COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for each teacher
fullName	String	Full name of the teacher
institute	String	Institution associated with the teacher
phoneNumber	String	Contact number of the teacher
email	String	Email address of the teacher
gender	String	Gender of the teacher
password	String	Hashed password for authentication
courses	Array	List of course IDs (ObjectId) created by the teacher
flag	Int32	Status flag (e.g., verification status)
createdAt	Date	Timestamp when the teacher account was created
updatedAt	Date	Timestamp when the teacher account was last updated

5.2.4 STUDENT COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for each student
firstName	String	First name of the student
lastName	String	Last name of the student
dob	Date	Date of birth of the student
email	String	Email address of the student
phone	String	Contact number of the student
password	String	Hashed password for authentication
address	String	Residential address of the student
courses	Array	List of enrolled course IDs (ObjectId)
quizzes	Array	List of quiz attempt records
quizzes._id	ObjectId	Unique identifier for each quiz attempt
quizzes.course	ObjectId	The ID of the course for which the quiz was attempted
quizzes.quiz	ObjectId	The ID of the attempted quiz
quizzes.totalMarks	Int32	Total marks available in the quiz
quizzes.marksScored	Int32	Marks scored by the student
completedLessons	Array	List of lesson IDs that the student has completed
createdAt	Date	Timestamp when the student account was created
updatedAt	Date	Timestamp when the student account was last updated

5.2.5 COURSE COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the course
title	String	Main title of the course
name	String	Subtitle or specific focus area of the course
description	String	Brief summary of the course content
imageurl	String	Path to the course thumbnail or cover image
teacher	ObjectId	Reference to the **Teacher** who created the course
students	Array	List of **Student ObjectIds** who have enrolled in the course
chapters	Array	List of **Chapter ObjectIds** related to the course
quizzes	Array	List of **Quiz ObjectIds** linked to the course
reviews	Array	List of **Student Reviews**, including:
reviews._id	ObjectId	Unique identifier for each review
reviews.studentName	String	Name of the student who gave the review
reviews.comment	String	Feedback or comment given by the student
reviews.rating	Int32	Rating given by the student (e.g., out of 5)
createdAt	Date	Timestamp when the course was created
updatedAt	Date	Timestamp when the course was last updated

5.2.6 CHAPTER COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the chapter
name	String	Title of the chapter
course	ObjectId	Reference to the **Course** this chapter belongs to
lessons	Array	List of **Lesson ObjectIds** associated with this chapter
createdAt	Date	Timestamp when the chapter was created
updatedAt	Date	Timestamp when the chapter was last updated

5.2.7 LESSON COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the lesson
number	Int32	Lesson number within the chapter (ordering)
title	String	Title of the lesson
description	String	Brief overview of what the lesson covers
videoUrl	String	Path to the lesson's video content
chapter	ObjectId	Reference to the **Chapter** this lesson belongs to
createdAt	Date	Timestamp when the lesson was created
updatedAt	Date	Timestamp when the lesson was last updated

5.2.8 QUIZ COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for the quiz
title	String	Title of the quiz (e.g., "HTML Quiz")
questions	Array	List of question objects
questions._id	ObjectId	Unique identifier for each question
questions.question	String	The quiz question text
questions.options	Array	Array of possible answer choices
questions.answer	String	Correct answer choice
questions.marks	Int32	Marks allocated for the question
course	ObjectId	Reference to the **Course** the quiz belongs to
createdAt	Date	Timestamp when the quiz was created
updatedAt	Date	Timestamp when the quiz was last updated

5.2.9 REVIEW COLLECTION

FIELDS	DATATYPE	DESCRIPTION
_id	ObjectId	Unique identifier for each review
courseId	ObjectId	Reference to the course being reviewed
studentName	String	Name of the student who gave the review
rating	Int32	Rating given by the student (e.g., out of 5)
comment	String	Feedback or comment given by the student
todaysdate	String	Date when the review was submitted (YYYY-MM-DD)
createdAt	Date	Timestamp of when the review was created
updatedAt	Date	Timestamp of when the review was last updated

5.3 INPUT DESIGN

The input design of Learnfinity ensures a seamless login and course interaction experience through structured form inputs like email, password, and role selection. Real-time validation enhances data accuracy with immediate feedback. A user-friendly UI/UX provides clear prompts for authentication and course enrollment, integrating role-based access control for students, teachers, and admins. The Forgot Password feature enables secure account recovery, optimizing usability and enhancing the learning experience.

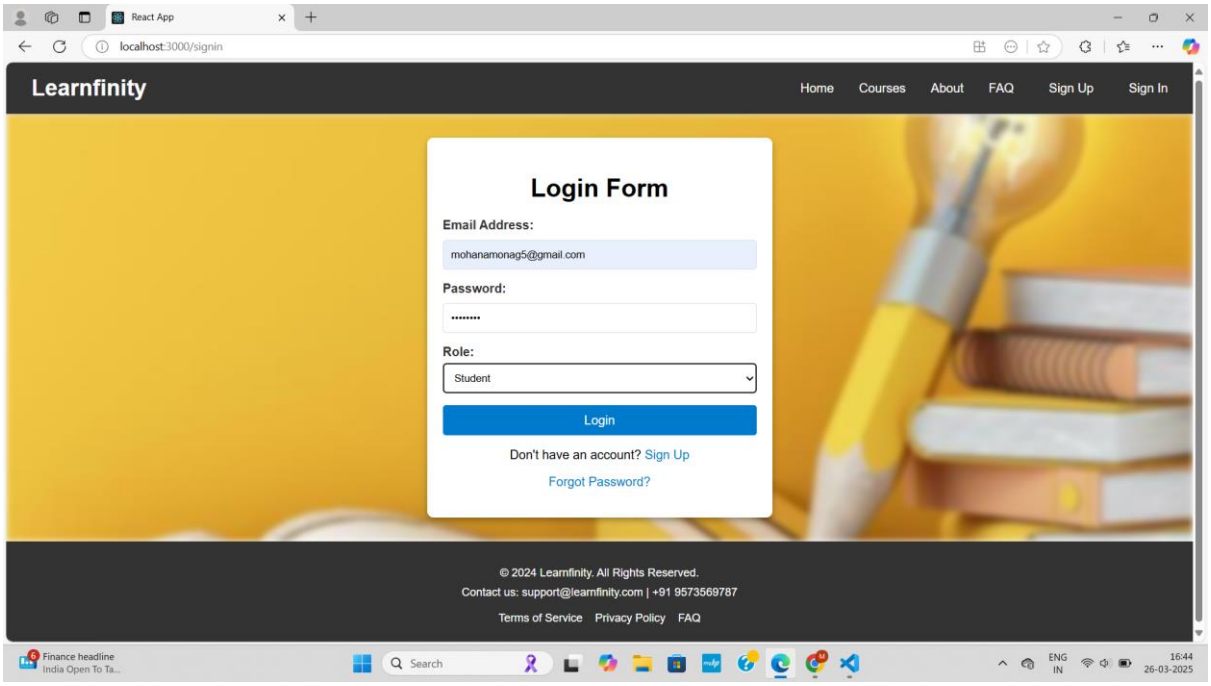


Figure:5.3 User Authentication Interface

Description: Showcases the input design with structured form fields for authentication

5.4 OUTPUT DESIGN

The output design of Learnfinity ensures effective information presentation and seamless navigation for students, teachers, and admins. A structured UI/UX displays enrolled courses, progress tracking, and interactive materials while providing real-time updates and personalized recommendations. Responsive feedback mechanisms enhance usability, making the learning experience more engaging and efficient.

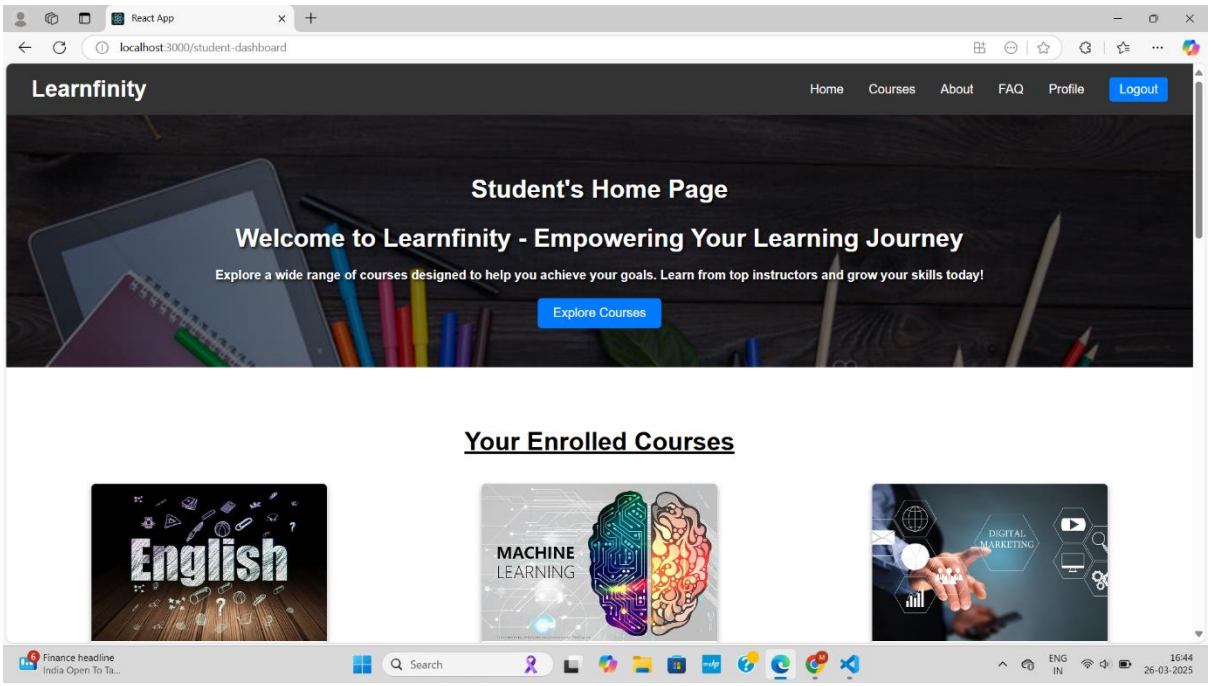


Figure:5.4 Student Dashboard View

Description: Displays the output design of the student dashboard with enrolled courses

CHAPTER - 6

SYSTEM TESTING & IMPLEMENTATION

6.1 SYSTEM TESTING

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is Defect free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

Some prefer saying Software testing definition as a White Box and Black Box Testing. In simple terms, Software Testing means the Verification of Application Under Test (AUT). This Software Testing course introduces testing software to the audience and justifies the importance of software testing.

6.1.1 UNIT TESTING

All modules were tested and individually as soon as they were completed were checked for their correct functionality. Unit testing is carried out by verify and recover errors within the boundary of the smallest unit or a module. In this testing step, each module was found to be working satisfactory per the expected output of the module. In the package development, each module is tested separately after it has been completed and checked with valid data.

6.1.2 INTEGRATION TESTING

The entire project was split into small programs; each of these single programs gives a frame as an output. These programs were tested individually; at last all these programs where combined together by creating another program where all these constructions were used. It gives a lot of problem by not functioning in an integrated manner. The user interface testing is important since the user has to declare that the arrangements made in the frames are convenient and it is satisfied. When the frames are the test, the end user gave suggestion. Since they were much exposed to do the work manually.

6.1.3 FUNCTIONAL TESTING

It examines the functionality of the Learnfinity platform to ensure it meets the requirement specifications. The frontend components and HTML structure are tested across modern web browsers for compatibility. Various input fields, including dropdowns, text boxes, links, email, password, and role selection, are validated for accuracy and proper functionality.

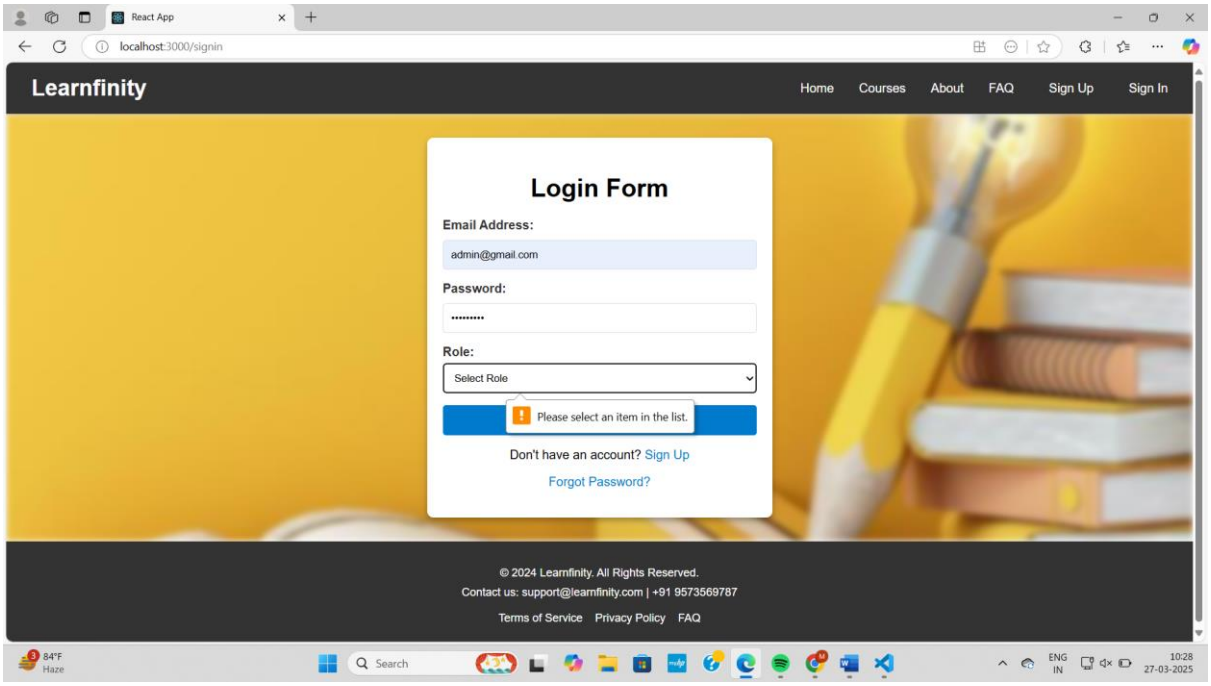


Figure:6.1.3 Login Page

Description: The form validation ensures accurate input by displaying warning messages for invalid or missing fields.

In the above figure, a warning message is displayed for unfilled fields, preventing form submission until all required fields are completed. The form validation ensures users provide necessary inputs before proceeding.

6.2 SYSTEM IMPLEMENTATION

Implementation is the stage where Learnfinity transitions from design to a fully functional e-learning system. This phase includes planning, system testing, user onboarding, and performance evaluation to ensure seamless functionality for students, teachers, and administrators.

The process begins with meticulous planning, defining development timelines, technology stack, and deployment strategies. Learnfinity offers an intuitive user interface that simplifies course navigation, learning, and content management. JWT authentication ensures secure access control, while MongoDB efficiently handles user profiles, courses, enrollments, and progress tracking.

Key Implementation Aspects

- **Students:** Enroll in courses, access lessons, participate in quizzes, track progress, and earn certificates upon course completion.
- **Teachers:** Create and manage courses, chapters, lessons, and quizzes dynamically through a feature-rich dashboard.
- **Admin:** Oversee users, approve teacher requests, manage courses, and send announcements to students.

The backend, built with Node.js and Express.js, ensures a scalable and efficient API layer. The React.js frontend delivers a responsive and engaging user experience. Fetch and Axios are used for handling API requests, ensuring efficient data retrieval and seamless communication between the frontend and backend.

Security, error handling, and real-time updates are prioritized to maintain data integrity and system reliability. Rigorous testing, scalability considerations, and continuous optimization ensure Learnfinity runs smoothly.

To support easy adoption, comprehensive documentation, user training materials, and a feedback system are provided. With a solid foundation in modern web technologies, Learnfinity is now fully operational, offering a secure, scalable, and user-friendly e-learning experience for students, teachers, and administrators.

CHAPTER - 7

CONCLUSION

Every objective defined at the initial stages of the **LEARNFINITY** project has been successfully achieved. A trial run has been conducted, demonstrating the system's capability to overcome the limitations of existing e-learning platforms.

Learnfinity has been developed as a comprehensive web-based e-learning platform, ensuring seamless, secure, and interactive learning experiences. It incorporates structured role-based access for students, teachers, and administrators, providing an intuitive and efficient educational environment.

Leveraging the MERN stack, the platform features a modern UI, a scalable backend, and robust data management. Core functionalities, including course enrollment, progress tracking, quiz participation, certificate generation, course creation and management for teachers, as well as user oversight and platform administration for admins, have been seamlessly integrated with real-time updates and secured using JWT authentication to ensure efficiency, security, and a smooth user experience.

With its intuitive design, scalable architecture, and secure infrastructure, Learnfinity emerges as a powerful solution for digital education. The system is now fully operational, offering institutions, educators, and learners a dynamic and engaging online learning experience, with a strong foundation for future enhancements and scalability.

CHAPTER - 8

SCOPE FOR FUTURE ENHANCEMENT

Future enhancements for Learnfinity focus on improving engagement, personalization, security, and accessibility. A leaderboard system will be introduced to rank students based on quiz scores and course performance, fostering healthy competition and motivation. This feature will encourage students to actively participate in assessments, boosting overall learning outcomes.

An AI-driven course recommendation system will analyze student learning patterns, progress, and interests to suggest relevant courses. By offering personalized learning paths, students can explore subjects tailored to their needs, enhancing their educational journey.

To strengthen security, multi-factor authentication (MFA) and role-based data access will be implemented, ensuring user data protection and preventing unauthorized access. These measures will enhance the platform's reliability and safeguard sensitive information.

Additionally, a dedicated mobile application will be developed for seamless learning on the go. The integration of discussion forums and peer collaboration tools will foster interactive learning beyond course materials, allowing students to engage with instructors and peers in real time.

By incorporating these enhancements, Learnfinity will continue to evolve into a more dynamic, secure, and engaging e-learning platform, offering a highly personalized and interactive educational experience while ensuring accessibility and efficiency for students, teachers, and administrators.

CHAPTER - 9

BIBLIOGRAPHY

WEB REFERENCE

1. https://www.w3schools.com/

2. https://www.mongodb.com/

3. https://react.dev/

4. https://nodejs.org/en

5. https://expressjs.com/

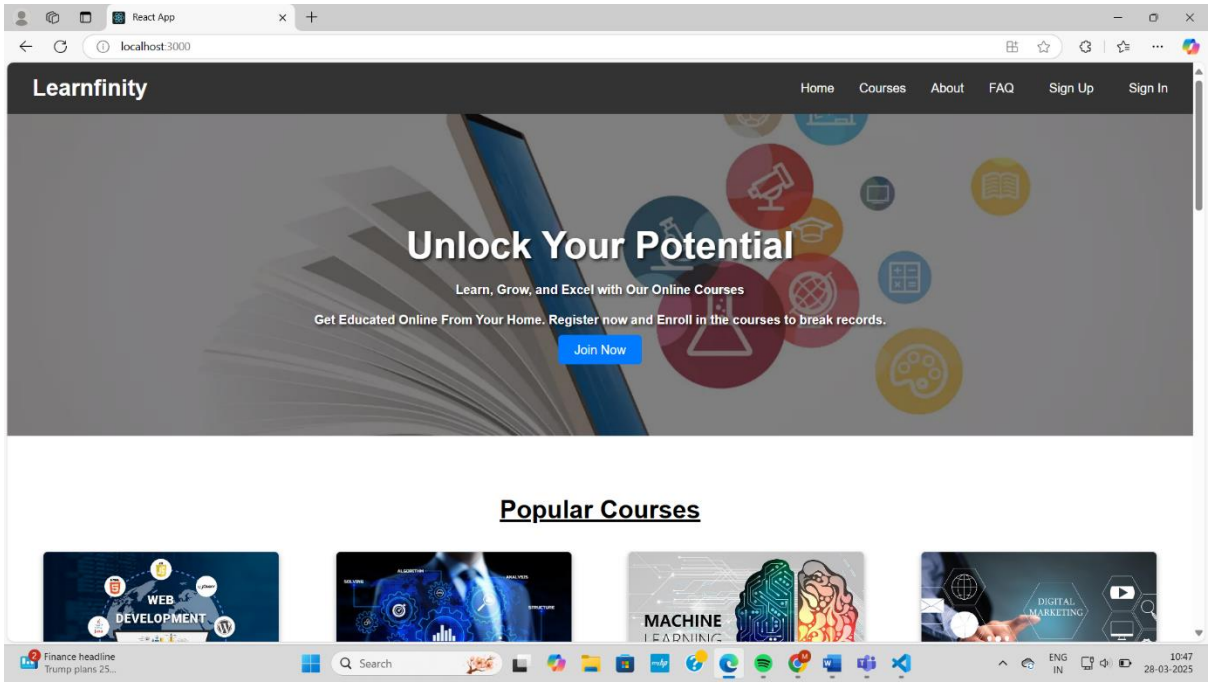
6. https://www.emailjs.com/docs/

CHAPTER - 10

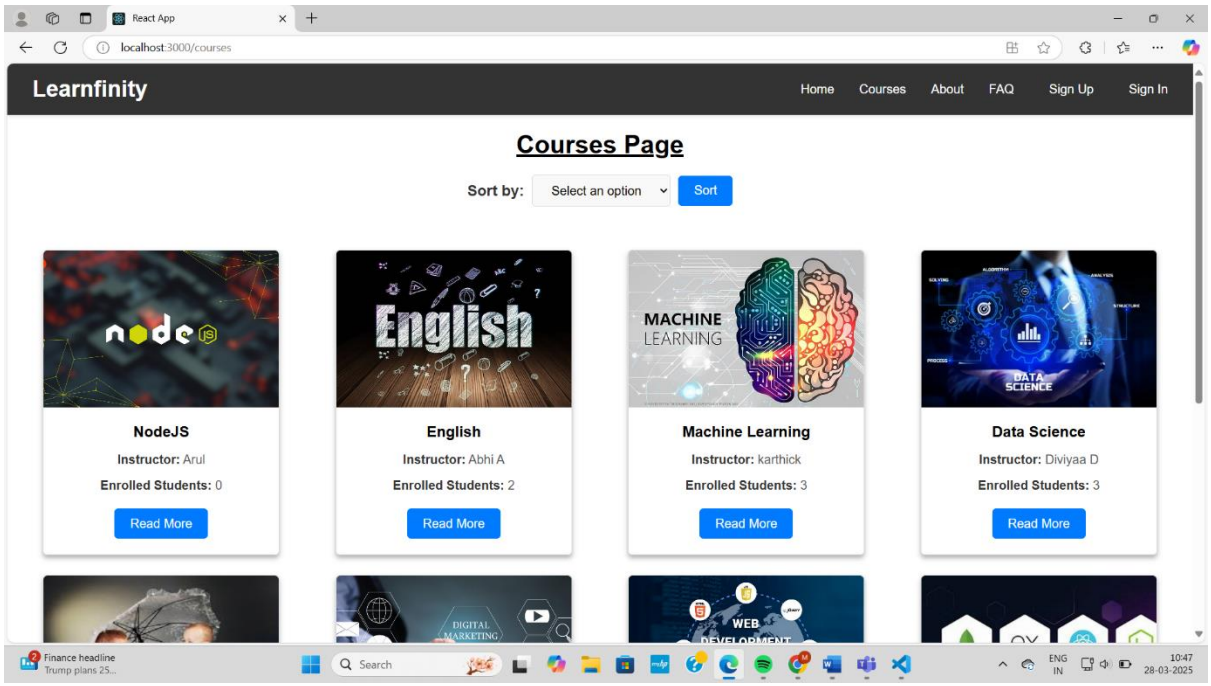
APPENDIX

A. SCREENSHOTS

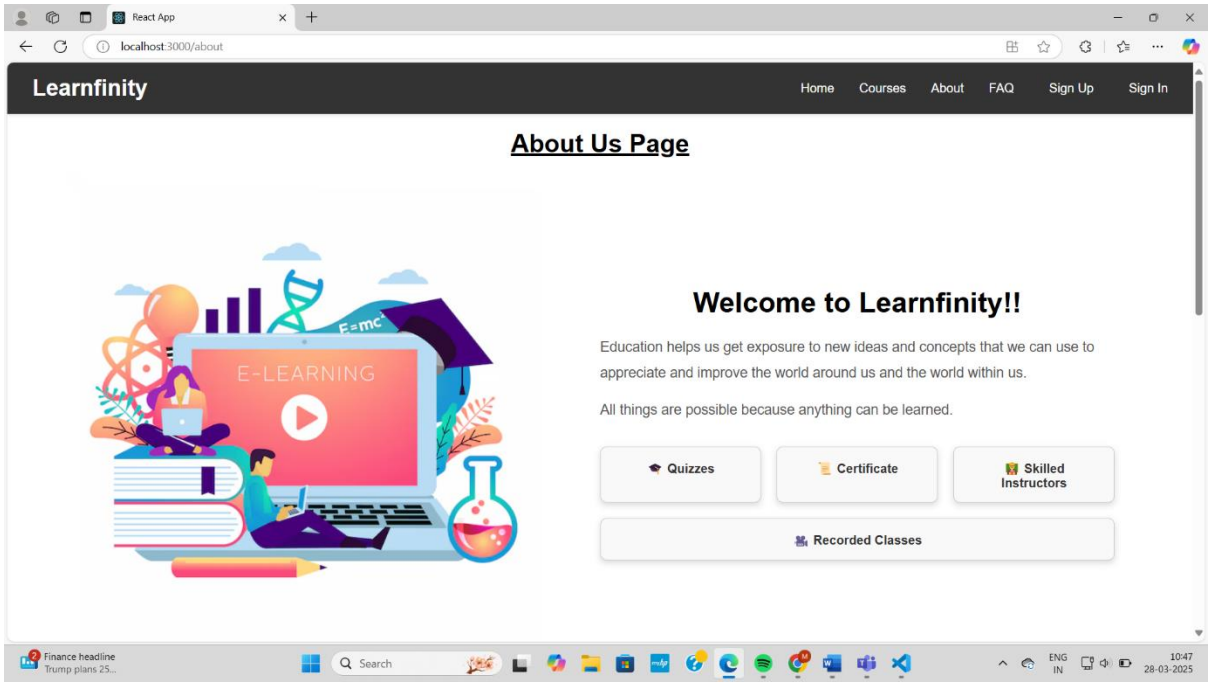
User Home page



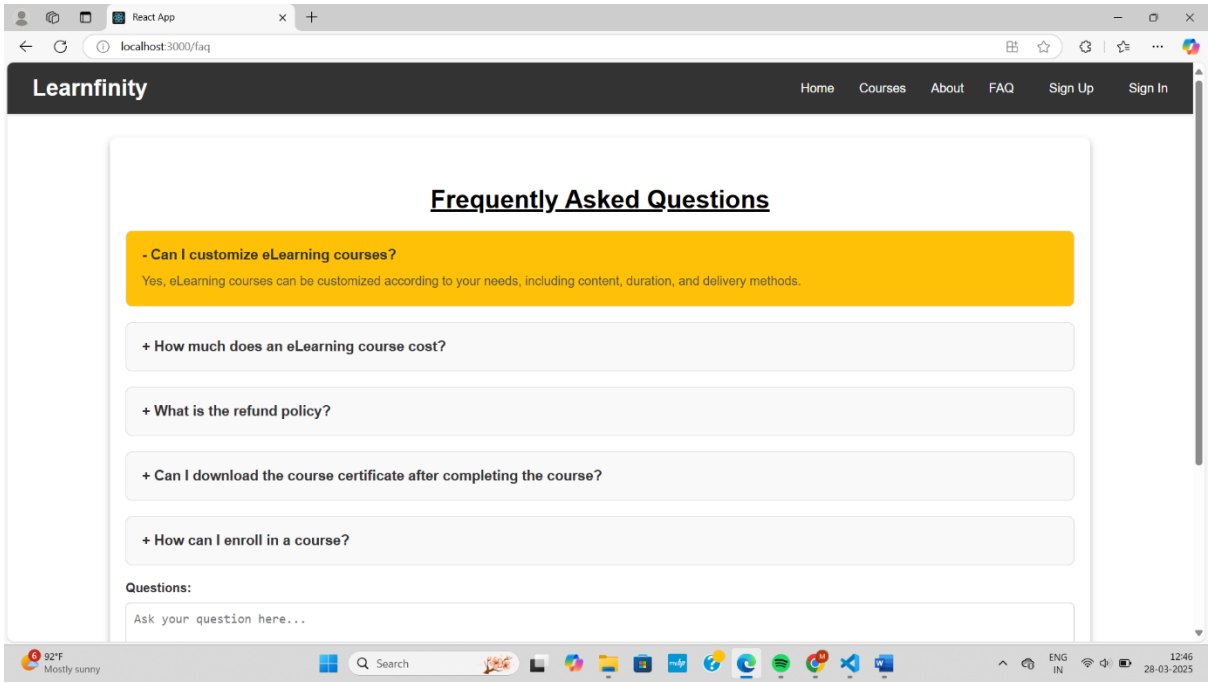
Course page



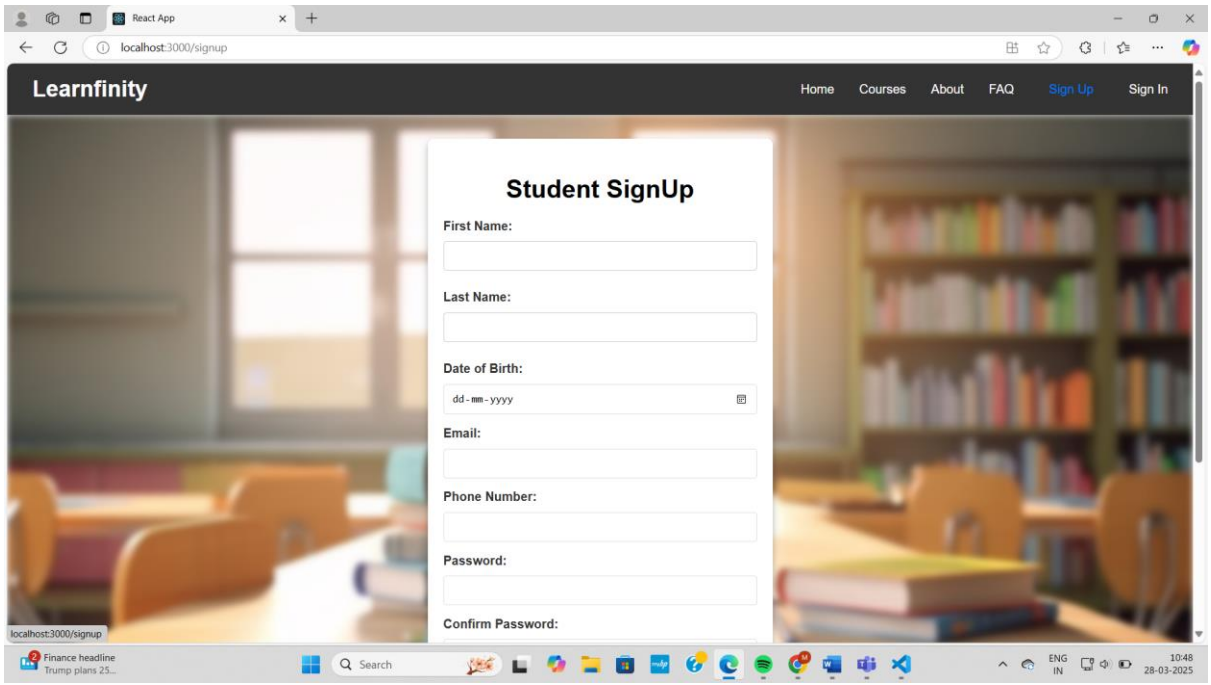
About page



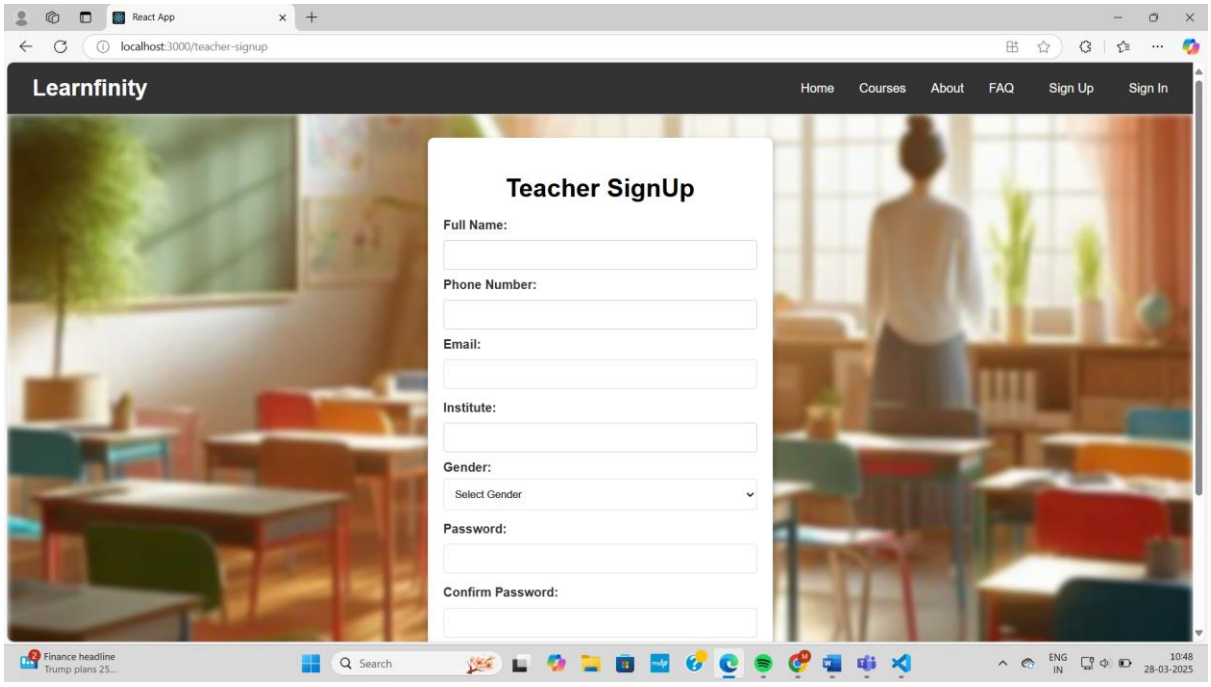
FAQ page



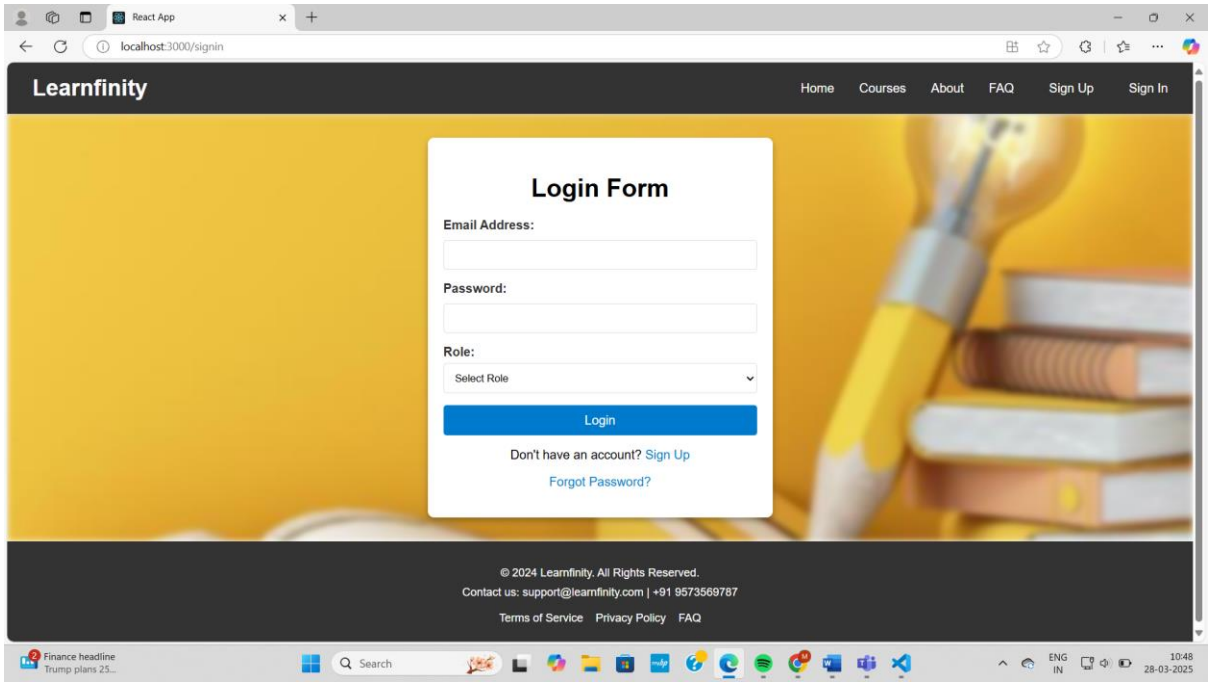
Student signup page



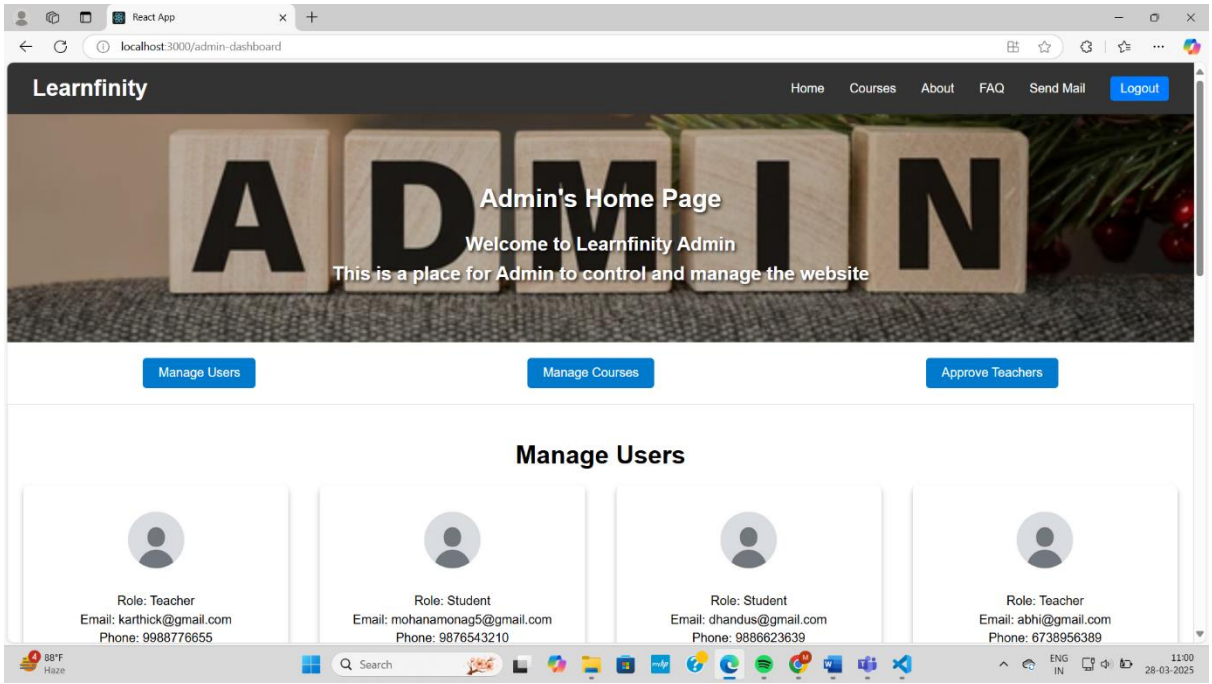
Teacher signup page



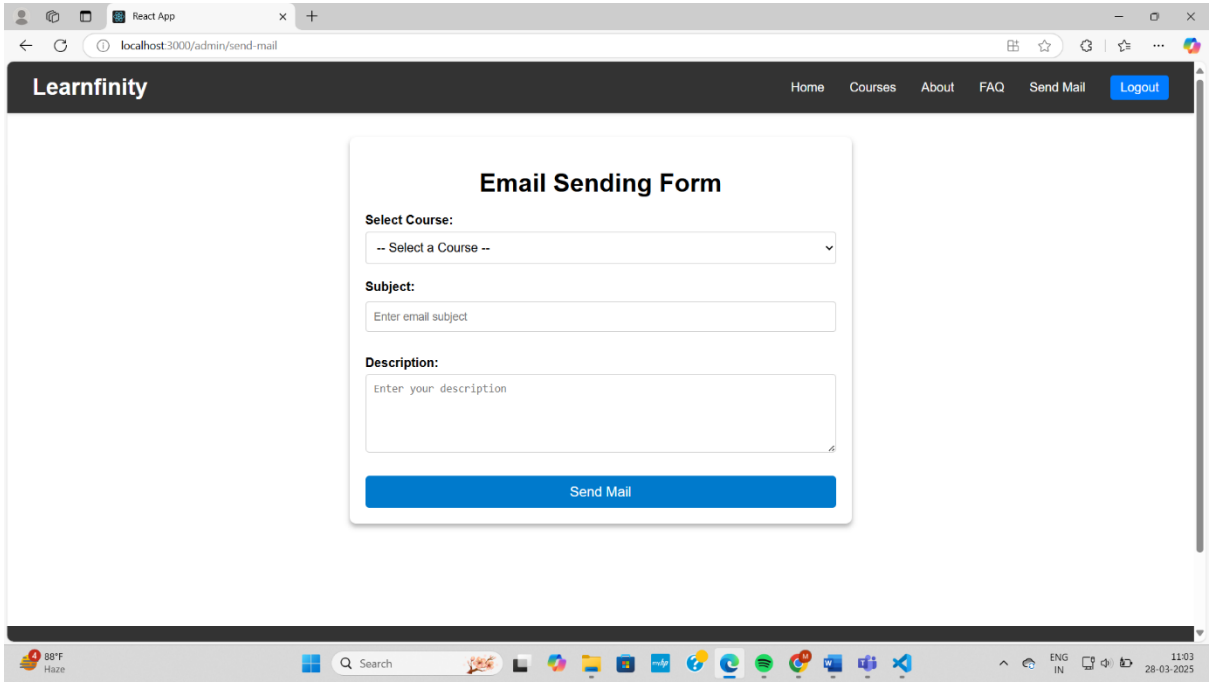
Login page



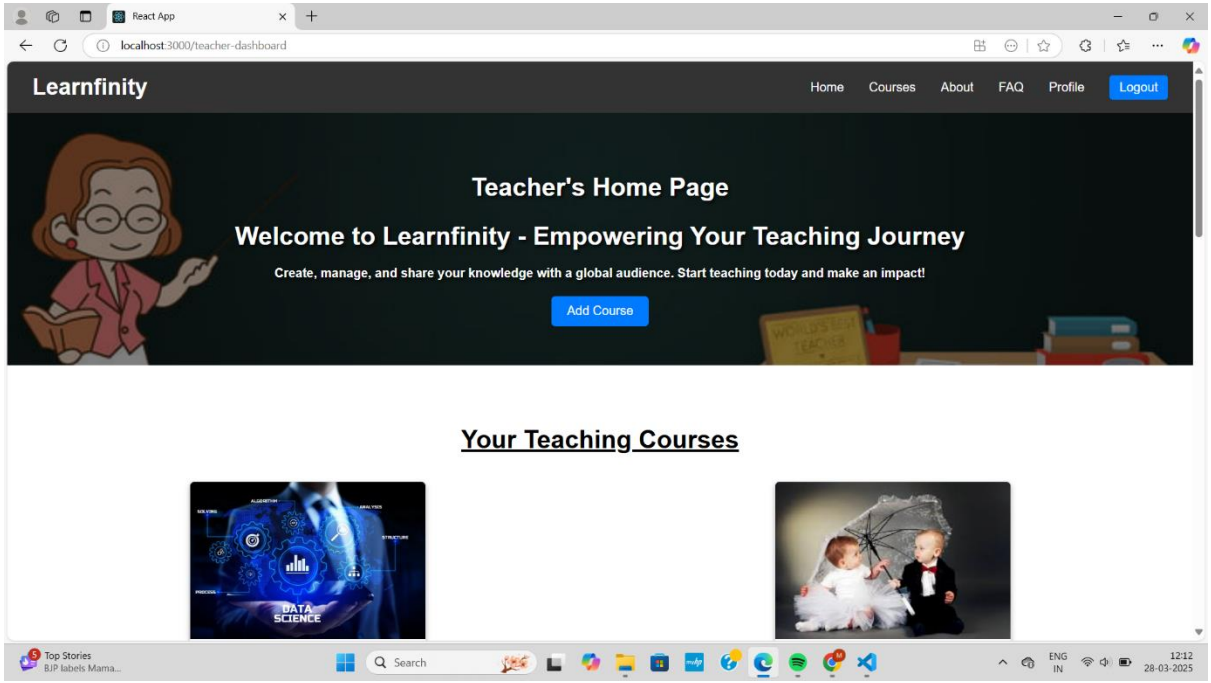
Admin dashboard page



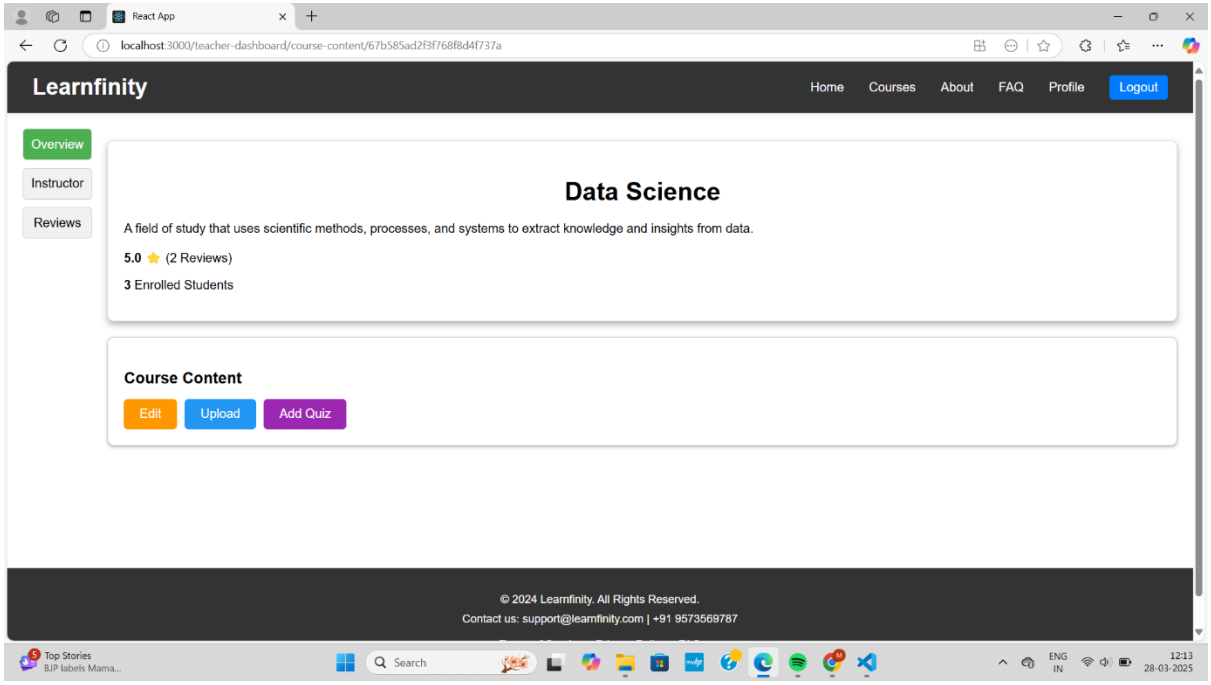
Admin email sending page



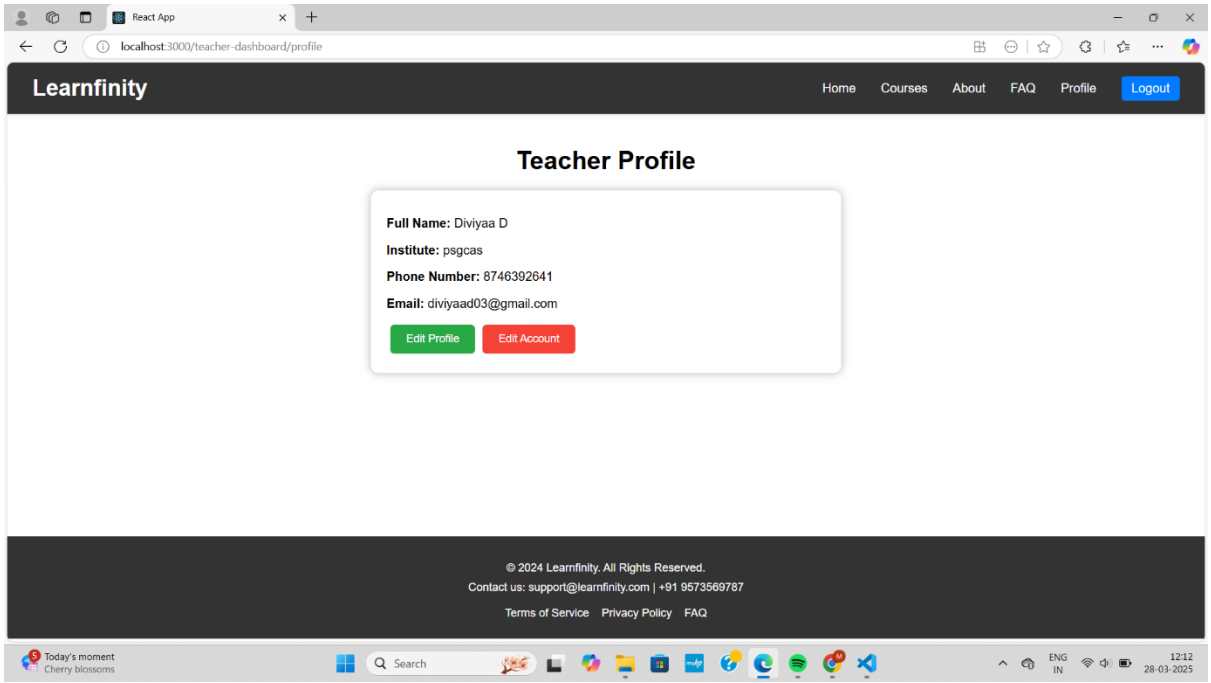
Teacher dashboard page



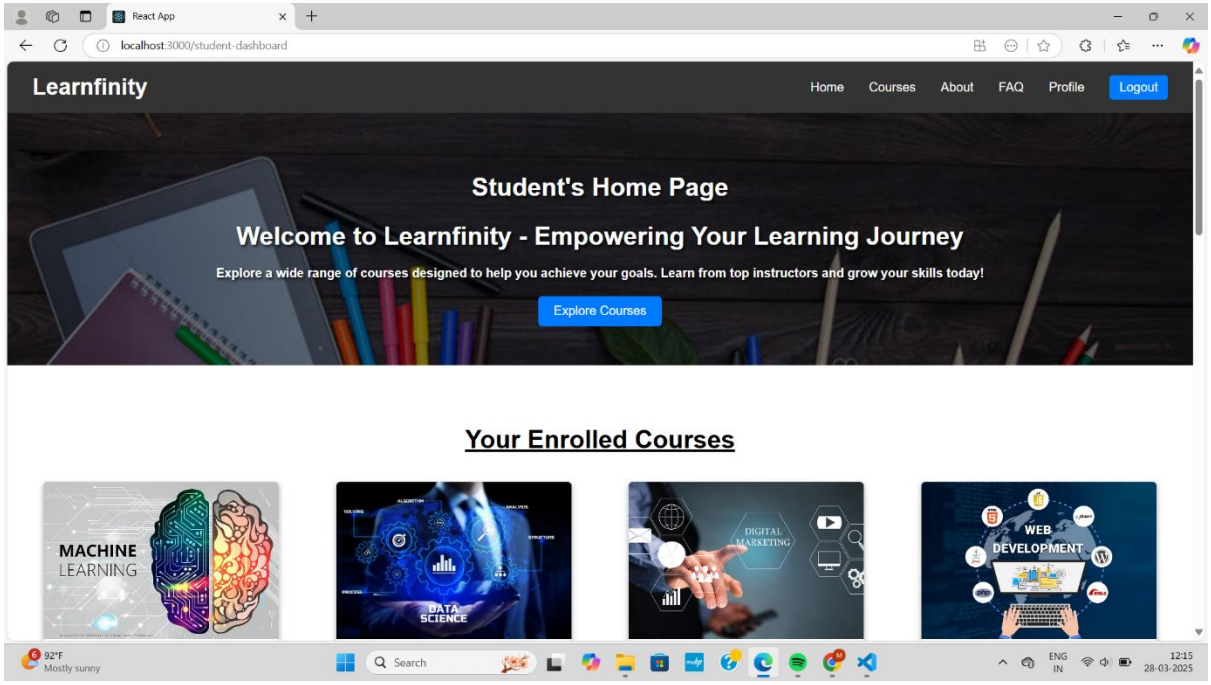
Teacher manages course content page



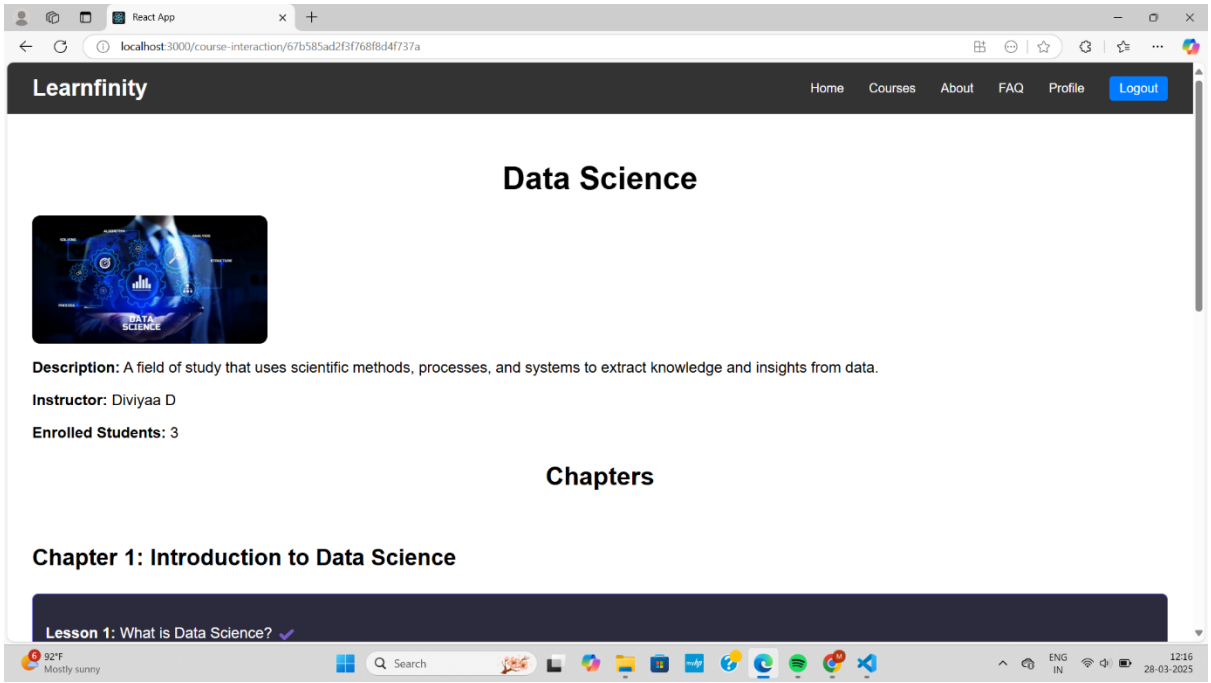
Teacher profile page



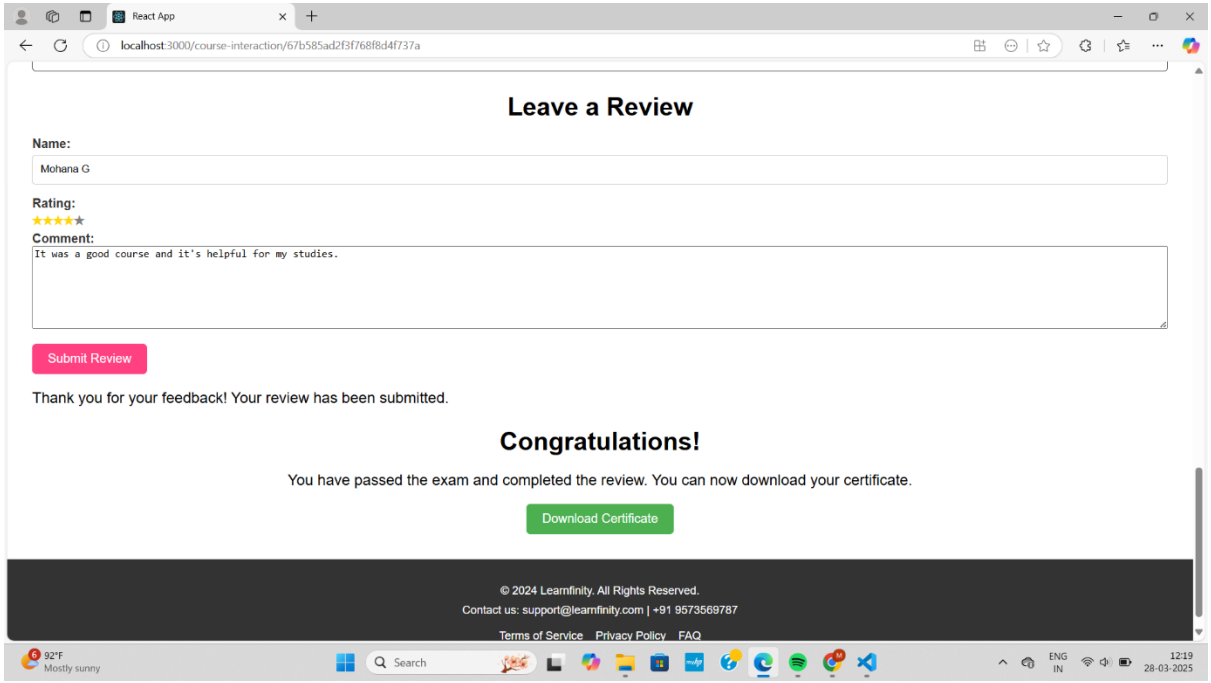
Student dashboard page



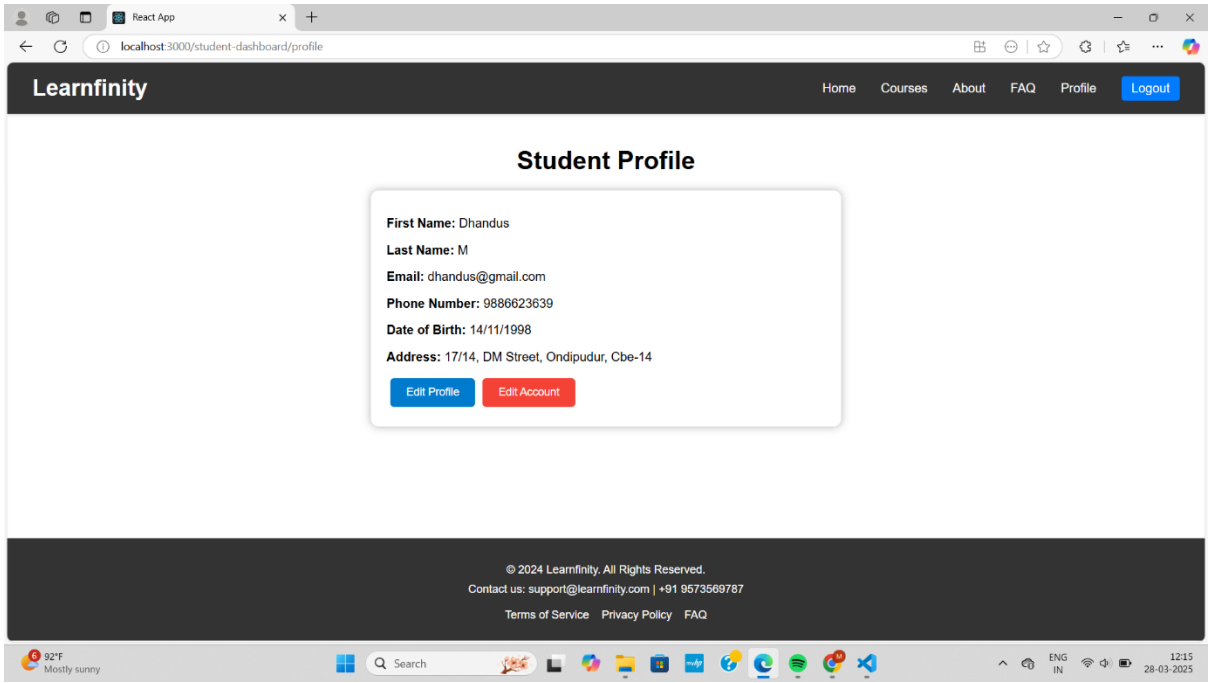
Course materials page



Course completion page



Student profile page



B. SAMPLE CODING

```
import React, { useState } from 'react';
import './SignIn.css';
const SignIn = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [role, setRole] = useState('');
  const [error, setError] = useState('');
  const handleSubmit = async (e) => {
    e.preventDefault();
    if (!email || !password || !role) {
      setError('Please fill all fields');
      return;
    }
    try {
      let loginUrl = '';
      // Select the correct login URL based on the role
      if (role === 'admin') {
        loginUrl = 'http://localhost:5000/api/auth/admin-login';
      } else if (role === 'teacher') {
        loginUrl = 'http://localhost:5000/api/auth/teacher-login';
      } else if (role === 'student') {
        loginUrl = 'http://localhost:5000/api/auth/student-login';
      // Send the email and password to the backend
```

```
const response = await fetch(loginUrl, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
  },
  body: JSON.stringify({
    email,
    password,
    role,
  }),
});
const data = await response.json();
// Check if the user is blocked (status 403)
if (response.status === 403) {
  setError(data.error);  // Display "Your account has been blocked by the admin."
  return;
}
// Handle other error responses
if (!response.ok) {
  setError(data.error || 'Invalid credentials');
  return;
}
  // Check if the response is ok (status 200-299)
  if (!response.ok) {
```

```
      throw new Error('Login failed. Please check your credentials.');
    }
  //const data = await response.json();
  if (data.success) {
    setError('');
    // Store the JWT token in localStorage
    localStorage.setItem('token', data.token);
    localStorage.setItem('role', role);
    // Redirect to the appropriate dashboard based on the role
    if (role === 'admin') {
      window.location.href = '/admin-dashboard';
    } else if (role === 'teacher') {
      window.location.href = '/teacher-dashboard';
    } else if (role === 'student') {
      window.location.href = '/student-dashboard';
    }
  } else {
    setError(data.error || 'Invalid credentials');
  }
} catch (err) {
  // Log and show the error in case of network failure or other issues
  console.error(err);
  setError('Error logging in. Please try again.');
}  };
```

```
return (
  <div className="signin-page">
    <div className="signin-container">
      <h2>Login Form</h2>
      {error && <div className="error-message">{error}</div>}
      <form onSubmit={handleSubmit} className="signin-form">
        <label htmlFor="email">Email Address:</label>
        <input
          type="email"
          id="email"
          name="email"
          value={email}
          onChange={(e) => setEmail(e.target.value)}
          required
        />
        <label htmlFor="password">Password:</label>
        <input
          type="password"
          id="password"
          name="password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
          required
        />
```

```
          <label htmlFor="role">Role:</label>
          <select
            id="role"
            name="role"
            value={role}
            onChange={(e) => setRole(e.target.value)}
            required
          >
            <option value="">Select Role</option>
            <option value="admin">Admin</option>
            <option value="teacher">Teacher</option>
            <option value="student">Student</option>
          </select>
          <button type="submit" className="signin-btn">Login</button>
        </form>
        <div className="additional-links">
          <p>Don't have an account? <a href="/signup">Sign Up</a></p>
          <p><a href="/forgot-password">Forgot Password?</a></p>
        </div>
      </div>
    </div>
  );
};
export default SignIn;
```

SYNOPSIS

The project entitled as "**Learnfinity**" is developed as a web application for an advanced e-learning platform, designed to offer a seamless and secure learning experience.

The objective of this project is to develop Learnfinity, a MERN-based e-learning platform that ensures secure, role-based access while providing an interactive and scalable learning experience with real-time updates and JWT authentication.

The system incorporates role-based access control for students, teachers, and administrators, ensuring structured user interactions. Students can enroll in courses, take quizzes, track their progress, and earn completion certificates. Teachers can create and manage courses, quizzes, and lessons, while administrators oversee platform management, user roles, and course approvals.

Learnfinity is designed for educational institutions, online course providers, and independent instructors to facilitate interactive digital learning. The platform supports multimedia lessons, assessments, and performance tracking, making education accessible and engaging.

The web application is built using React.js for the frontend, Node.js and Express.js for the backend, and MongoDB as the database. This combination ensures a modern UI design, scalable backend infrastructure, and secure data management, resulting in efficient functionality and seamless user interactions.

Mohana G (20MSS026), FIFTH year M.Sc. Software Systems,

Batch (2020-2025), PSG College of Arts & Science, Coimbatore.