# Introduction

* + Project Title: *Online Learning Platform using MERN*
  + Team Members:
    - Sudharshan R
    - Sarath Kumar L
    - Sri Varman A P
    - Sachin Sagar M
    - Manoj Kumar S

# Project Overview

* + Objective is to develop a robust online learning platform that facilitates seamless course delivery, student engagement, progress tracking, and feedback collection. This platform will be built with the MERN stack to ensure high performance, scalability, and an intuitive user interface.
  + Features:
    - User Authentication & Authorization.
    - User profile.
    - Course Management.
    - Lesson & Content Delivery.
    - Payment & Subscriptions.
    - Progress Tracking & Feedback.
    - Admin Dashboard.

# Architecture

* + Frontend:
    - React: Component-based structure for UI.
    - Redux: Global state management for authentication, course data, and progress.
    - Chakra UI: Provides accessible, customizable components and a consistent design system.
    - Chart.js: For visualizing progress (e.g., progress charts).
  + Backend:
    - Database:
      * MongoDB: Stores user, course, and progress data. Flexible schemas with Mongoose handle various content types (videos, quizzes) and track user progress.
    - API Structure:
      * RESTful Endpoints: Organized in the routes/ folder by feature (e.g., /auth, /courses, /users, /progress).
      * Controllers: Handle business logic (e.g., user login, course creation), maintaining separation of concerns.
    - Authentication & Authorization:
      * JWT (JSON Web Tokens): Manages secure user sessions and role-based access.
      * Role-Based Access Control: Middleware in authMiddleware.js enforces permissions for students, instructors, and admins, restricting access based on user roles.
    - Mongoose Models:
      * User Model: Handles user information, roles, and enrolled courses.
      * Course Model: Stores course details, content structure, and associated lessons.
      * Lesson Model: Tracks individual lessons within courses, including content type (e.g., video, quiz).
      * Progress Model: Tracks user progress within courses, enabling tracking and progress display.
    - Middlewares:
      * Authentication Middleware: Verifies JWT tokens to secure routes.
      * Error Handling Middleware: Centralized error handling to ensure consistent API responses and debugging.
    - Services:
      * Email Service: Sends notifications like enrollment confirmations.
      * Payment Service: Manages secure payment processing for course subscription

# Setup Instructions

* + Prerequisites:

1. *Node.js* (version 22.11.0)
2. *MongoDB* (version 8.0.1)
   * Installation:
3. Clone the repository:

https://github.com/Sudharsreddy/Online-Learning-Platform-using-MERN.git

1. Navigate to the project directory.
2. Install dependencies:
   * Frontend: cd client && npm install
   * Backend: cd server && npm install
3. Set up environment variables:

* MongoDB URI
* Any necessary API keys for authentication and payments

# Folder Structure

* + Client:
    - *src*: Contains main React components, services, and assets.
    - *public*: Static files for the React application.
    - *package.json*: Lists dependencies and scripts for the frontend.
  + Server:
    - *routes*: Defines API routes for materials, users, and courses.
    - *models*: Schemas for MongoDB collections.
    - *controllers*: Functions for handling API requests.
    - *config*: Configuration files for environment variables.
    - *package.json*: Lists dependencies and scripts for the backend.

# Running the Application

* <https://e-learning-2eqp.onrender.com/>

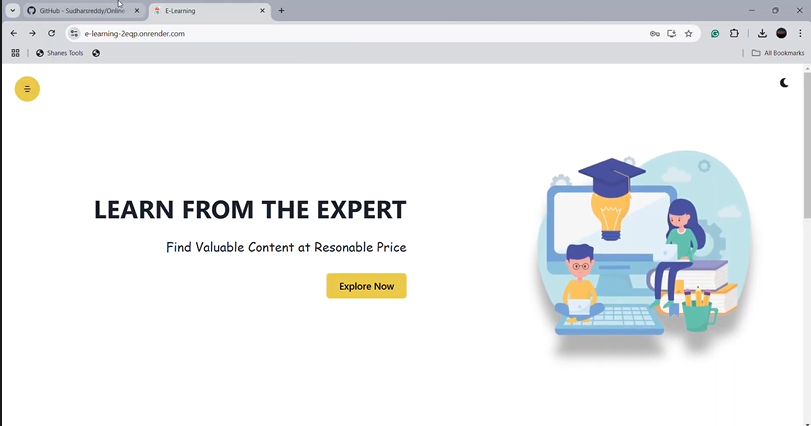
# API Documentation

* + Endpoints:
    - Register a new user
    - Get logged-in user profile
    - Logout user
    - Get user profile by ID
    - Update user profile by ID
    - Get all users
    - Get all available courses
    - Get details of a specific course
    - Create a new course
    - Update course details
    - Delete a course

# Authentication

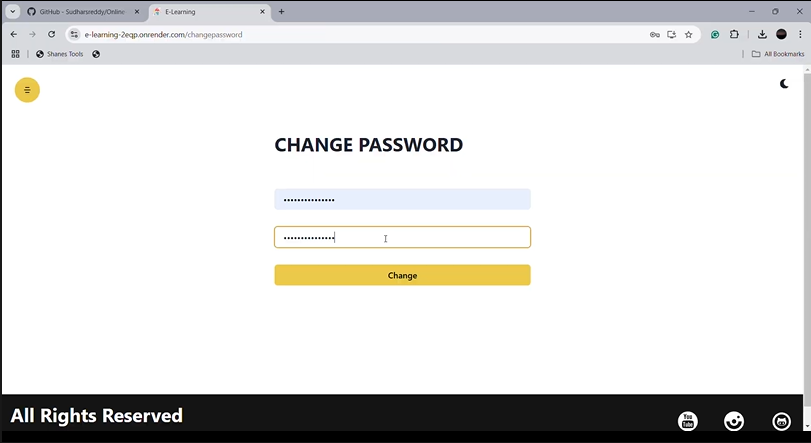
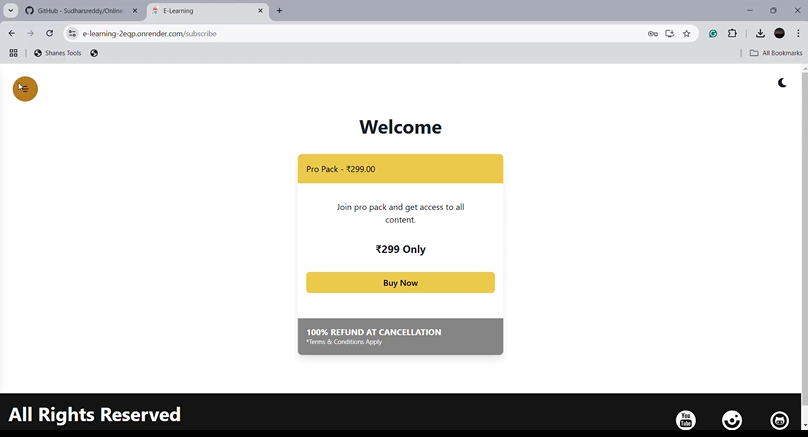
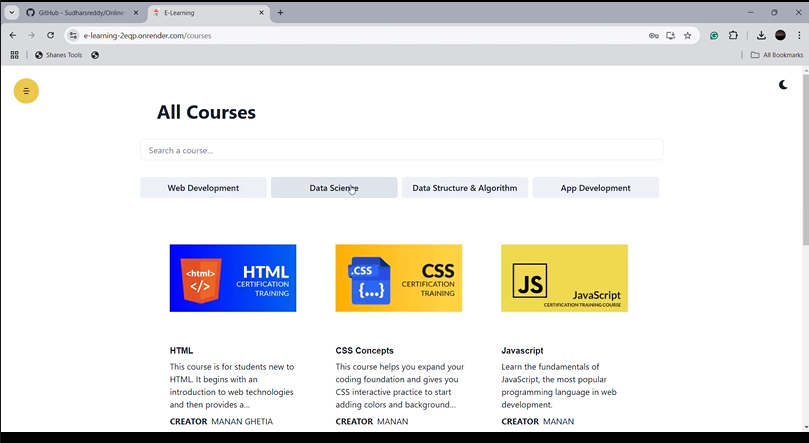
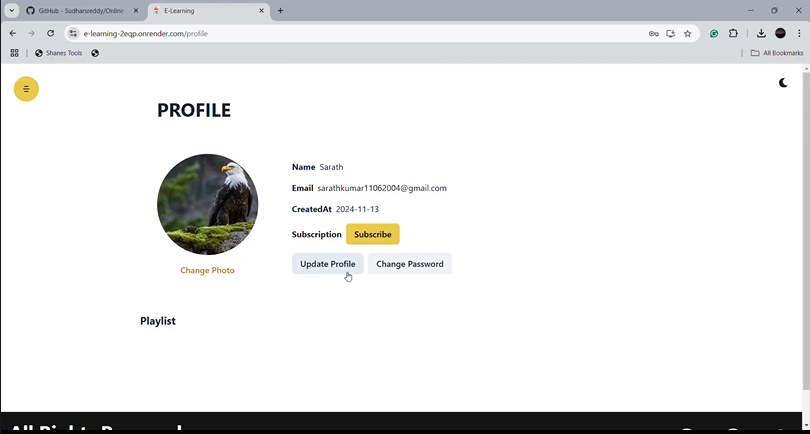
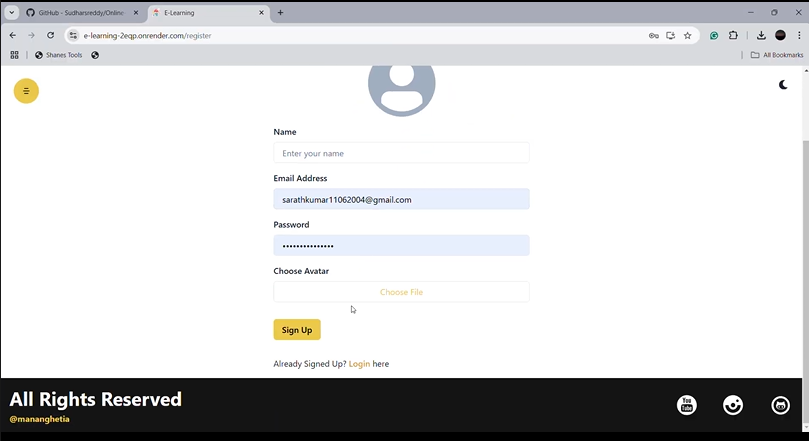
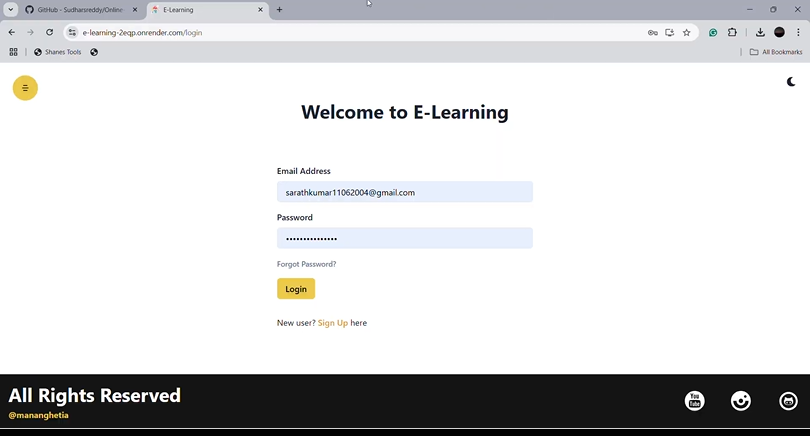
* + Authentication is managed using JWT tokens. Upon login, a token is issued and stored on the client side to maintain session state. Middleware is implemented to verify tokens for protected routes, ensuring secure access to user-specific and sensitive data.

# User Interface

****

1. **Testing**
   * Testing strategies include unit testing with *Jest* for React components and API testing with *Mocha* and *Chai*. These tests ensure components and endpoints work as expected under various scenarios.

# Screenshots or Demo

****

1. **Known Issues**
   * Document any bugs, limitations, or areas needing improvement, such as performance issues with large data sets or specific edge cases in the shopping cart functionality.

# Future Enhancements

* + Potential features to add include:
    - Enhanced Content Delivery.
    - Personalized Learning Paths.
    - Gamification and Engagement Features.
    - Advanced Analytics and Reporting.
    - Community Features.
    - Certification and Career Support.
    - Mobile Application Support.
    - Integration with External Tools.
    - Advanced Security and Compliance.
    - AI and Machine Learning.
    - Monetization Enhancements.
    - Scalability and Performance Improvements.