**Project Title**

**LearnHub: Your Center for Skill Enhancement**

**Team ID:** **LTVIP2025TMID52959**

**Team Members:**

Here List team members and their roles:

**1**.**M Uma Maheswari:** Combines both frontend and backend responsibilities, ensuring smooth communication between the two. This role also handles bug fixing, feature integration, and overall system performance.

**2. L Suvarchala** (Frontend Developer):Responsible for designing the user interface using React.js. This role focuses on ensuring a responsive, user-friendly design, as well as integrating the frontend with backend APIs.

**3.L**.**Kiran Kumar** and **Kundeti Lakshmi Sumanth** (Backend Developer): Develops the backend server using Node.js and Express.js, ensuring the creation of secure, scalable RESTful APIs, as well as handling authentication, data processing, and business logic.

**Introduction:**

Welcome to LearnHub – Your center for Skill Enhancement.  
In today's digital age, LearnHub is designed to redefine the way students and teachers interact through an innovative online learning platform. Whether you are a student eager to explore new knowledge or a teacher aiming to share your expertise, LearnHub offers a smooth and user-friendly experience tailored to your needs.

The platform provides a structured environment for managing courses, tracking learning progress, and facilitating communication between learners and educators. LearnHub is built to support a modern learning journey that is secure, efficient, and accessible from anywhere.

**Project Overview:**

LearnHub is a full-stack web application that serves as a comprehensive learning management system for students, teachers, and administrators. It provides features that support online course creation, student enrollment, lesson management, and certification.

**Key Features:**

* Role-based login for students, teachers, and admins
* Course creation, editing, and deletion by teachers
* Lesson management within each course
* Student enrollment and real-time progress tracking
* Certificate generation for students who complete all lessons
* Dashboard for each user type to manage their learning or teaching activities
* Search and filter functionality to find courses easily
* Secure authentication using tokens and protected API routes
* Responsive frontend for both desktop and mobile users

LearnHub is built using modern technologies such as React.js (Vite), Node.js, Express.js, and MongoDB. It ensures a smooth and interactive learning experience for all users.

**Goals of Project:**

The main goal of LearnHub is to create a complete and user-friendly **Online Learning Platform** that connects students, teachers, and administrators in one place. It aims to:

* Provide a structured environment for managing online courses and lessons
* Help students track their learning progress and earn completion certificates
* Enable teachers to easily create, edit, and manage their courses and lessons
* Support role-based access control for secure and personalized user experiences
* Ensure accessibility across devices with a responsive and modern interface
* Offer a real-world, full-stack project experience using the MERN stack

**Features of LearnHub:**

UserAuthentication and Role Management

* Secure login and registration with JWT token-based authentication
* Role-based access for students, teachers, and admins

**Student Features**

* Browse and search available courses
* Enroll in courses and view enrolled ones in the dashboard
* Track lesson completion and view progress bars
* Download certificate after completing all lessons

**Teacher Features**

* Create, update, and delete courses
* Add, edit, and remove lessons inside each course
* Manage enrolled students and course content

**Admin Features**

* View and manage all users, courses, and system activity
* Maintain overall control and system moderation

**Course and Lesson Management**

* Teachers can manage courses and lessons with rich inputs
* Students can view lessons and mark them as completed

**Certificate Generation**

* Automatically generate downloadable PDF certificates for students who complete a course

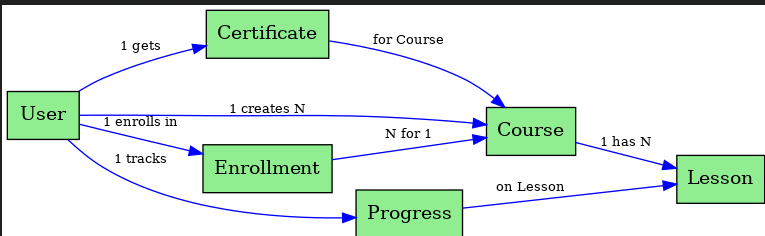
**Search and Filtering**

* Real-time search functionality for courses by title

**Responsive Design**

* Clean and responsive UI that works on desktops, tablets, and mobile devices

**Technical Architecture**

****

**1. User Interface**

The frontend built using React.js with Vite provides an intuitive and responsive user interface for:

* Students to browse courses, enroll, view lessons, and track progress
* Teachers to create and manage courses and lessons
* Admins to manage the platform and users

**2. Web Server**

Built using Express.js, the backend acts as the core server handling:

* API requests from the frontend
* Business logic for authentication, user management, course operations, lesson tracking, and certificate generation

**3. API Gateway**

The Express server functions as the central gateway for all requests. It:

* Authenticates users
* Validates role-based access
* Directs requests to appropriate controllers and services

**4. Authentication Service**

Implemented with JWT, this service:

* Handles secure login and registration
* Protects private routes
* Restricts access based on user roles (student, teacher, admin)

**5. Database (MongoDB)**

MongoDB stores and manages:

* User information (students, teachers, admins)
* Courses and lessons
* Enrollment and completion records
* Certificates and roles

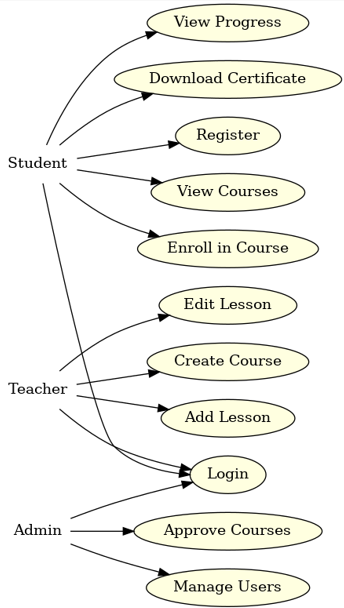
**6. Course & Lesson Features**

* Students can enroll in and complete lessons
* Teachers can add/edit/delete courses and lessons
* Progress is tracked and visualized using completion bars

**7. Certificate Generator**

A dynamic certificate generation feature creates downloadable PDFs when a course is fully completed by the student.

**ER Diagram Overview**

****

**User–Course Relationship**

* Many-to-Many
* One user can enroll in many courses; one course can be taken by many users.
* Stored via an array of user IDs (enrolled) in the Course schema**.**

**User–CompletedLesson Relationship**

* One-to-Many
* A user can complete many lessons. Tracked through a separate CompletedLesson collection**.**

**Course–Lesson Relationship**

* One-to-Many
* Each course has multiple lessons, stored by referencing courseID in the lesson.

**Pre-Requisites**

**✅ Node.js & npm**

* Used for running server-side JavaScript
* Install from: <https://nodejs.org/en/download>

**✅ Express.js**

* Backend web framework for handling routing and APIs
* Install via terminal:

**npm install express**

**✅ MongoDB**

* NoSQL database used to store users, courses, lessons, progress, etc.
* Install locally or use [MongoDB Atlas](https://www.mongodb.com/cloud/atlas)

**✅ React.js with Vite**

* Frontend library used for building fast and interactive user interfaces
* Setup using Vite:

npm create vite@latest learnhub -- --template react

cd learnhub

npm install

npm run dev

**✅ Git & GitHub (Optional but Recommended)**

* Use Git for version control and GitHub to host your project repository
* Download Git: <https://git-scm.com/downloads>

**✅ Visual Studio Code**

* Preferred editor for writing clean, structured, and well-formatted code
* Download: <https://code.visualstudio.com/download>

**How to Get Started**

**✅ Backend Setup**

1. Create a folder for the backend (e.g., /server)
2. Initialize project with npm init -y
3. Install required packages:

npm install express mongoose cors dotenv bcryptjs jsonwebtoken multer nodemon

**✅ Frontend Setup**

1. Use Vite to create the frontend:

npm create vite@latest frontend -- --template react

cd frontend

npm install

npm run dev

**✅ Database Connection**

* Use Mongoose to connect MongoDB with your backend:

npm install mongoose

* MongoDB connection string (.env):

env

MONGO\_URI=mongodb+srv://<username>:<password>@cluster.mongodb.net/learnhub

**✅ Run the Project**

* Start backend server**:**

**nodemon index.js**

* Start frontend:

npm run dev

**To Run the Existing LearnHub Full-Stack Project Locally**

**Step 1: Download the Project Code**

Download the complete LearnHub project from the following location:

GitHub **Repository:  
https://github.com/SiriVennelakaturi/LearnHub**

**Step 2: Install Dependencies**

Follow these steps to install all the required packages:

1. Open your terminal or command prompt
2. Navigate to the LearnHub project folder:

cd LearnHub

1. Install frontend dependencies:

cd frontend

npm install

1. Install backend dependencies:

cd ../backend

npm install

**Step 3: Configure Environment Variables**

1. In the backend folder, create a .env file if it doesn't exist.
2. Add the following required environment variables:

PORT=5000

MONGO\_URI=your\_mongodb\_connection\_string

JWT\_SECRET=your\_jwt\_secret\_key

Make sure your MongoDB server is running locally or use MongoDB Atlas with a valid connection string.

**Step 4: Start the Development Servers**

Start the backend server:

cd backend

npm run dev

In another terminal window or tab, start the frontend server:

cd frontend

npm run dev

**Step 5: Access the Application in the Browser**

Open your browser and go to:

http://localhost:5173

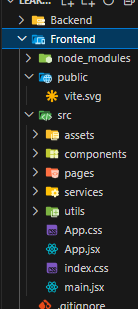
You should now see the login page of the LearnHub application. If the page loads successfully, your setup is complete and running properly.

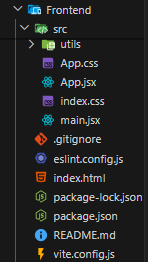
**Project structure:**

* Inside the LearnHub directory, we have the following folders



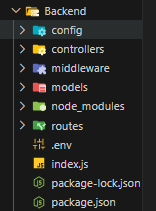
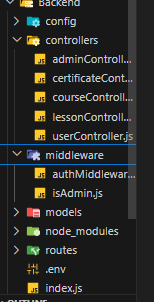
* **frontend directory:**

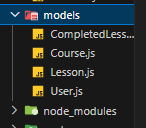




* **Server directory:**

The below directory structure represents the directories and files in the server folder (back end) where, node js, express js and mongodb.

**= **

****

**Project Flow:**

Use the code in:

https://github.com/SiriVennelakaturi/LearnHub

**Milestone 1: Project Setup and Configuration**

• Folder Setup:

Start by creating two main folders for organizing your codebase:

* frontend – for the React-based user interface.
* backend – for the Node.js/Express server and MongoDB connection.

• Installation of Required Tools:

Frontend Tools (React-based UI)

| Tool/Library | Installation Command |
| --- | --- |
| React with Vite | npm create vite@latest ./ |
| Bootstrap (or Tailwind) | npm install bootstrap |
| Axios (API Calls) | npm install axios |
| Firebase (optional) | npm install firebase |
| Framer Motion (optional animation) | npm install framer-motion |

Navigate to the frontend folder and run the above commands.

Backend Tools (Express Server)

| Tool/Library | Installation Command |
| --- | --- |
| Express.js | npm install express |
| Mongoose | npm install mongoose |
| Bcrypt (for password hashing) | npm install bcrypt |
| Body-Parser | npm install body-parser |
| CORS | npm install cors |
| dotenv (env file) | npm install dotenv |

Navigate to the backend folder and run the above commands.

**Milestone 2: Backend Development**

• Project Structure:

* Initialize your backend folder using npm init -y.
* Set up folders like routes, controllers, models, and middleware.

• Create Express.js Server:

* Create a server file (e.g., index.js) to start the Express server.
* Use middleware such as:
  + body-parser to handle JSON requests
  + cors to allow cross-origin access

• Define API Routes:

* Create routes for:
  + User Authentication (Register/Login/Profile)
  + Courses (Create, Edit, Delete, View)
  + Lessons (Add, Complete)
  + Certificate (Generate if completed)

• Implement Mongoose Models:

* Create schemas and models for:
  + User
  + Course
  + Lesson
  + CompletedLesson

• User Authentication:

* Use JWT for secure login.
* Build registration and login APIs.
* Add protected routes using a middleware that checks the token.

• Lesson Completion Tracking:

* Let students mark lessons as completed.
* Store this in a CompletedLesson model.

• Error Handling:

* Add centralized error handling middleware.
* Return appropriate HTTP status codes (e.g., 400, 403, 500).

**Milestone 3: Database Development**

* Use MongoDB Atlas or local MongoDB for the database.
* Create collections for:
  + Users
  + Courses
  + Lessons
  + CompletedLessons
* Code to connect the database in backend/config/db.js.

Example:

import mongoose from 'mongoose';

const connectDB = async () => {

await mongoose.connect(process.env.MONGO\_URI);

console.log('MongoDB connected');

};

export default connectDB;

**Milestone 4: Frontend Development and Integration**

1. Setup React Application:

* Create a React app using Vite.
* Use React Router for navigation.
* Setup Axios to make API calls.

2. Design UI Components:

* Create and design the following components:
  + Login Page
  + Register Page
  + Dashboard (Student/Teacher)
  + Course Details Page
* Use Bootstrap or Tailwind for styling.

3. Frontend Logic and API Integration:

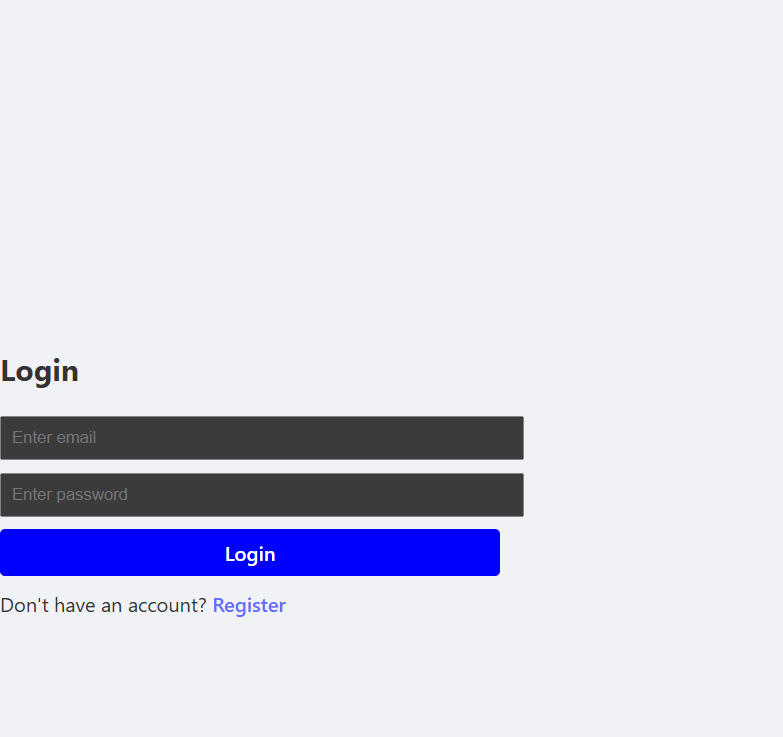
* Connect login, register, and dashboard to the backend using Axios.
* Implement logic to:
  + View enrolled courses
  + Mark lessons as complete
  + Track course progress
  + Download certificate (if course is completed)

**Milestone 5: Project Implementation and Testing**

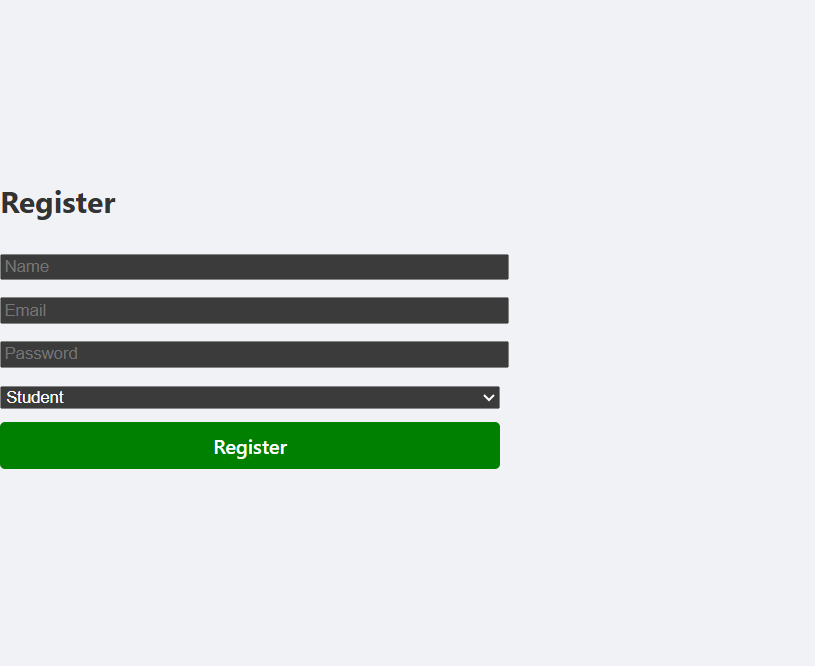
* Run both frontend and backend servers.
* Test the complete flow:
  + Register/Login
  + Enroll in courses
  + Mark lessons as complete
  + See progress and certificate
* Fix bugs if any appear.

After completing development and testing, your LearnHub Online Learning Platform is ready to deploy or share with others.

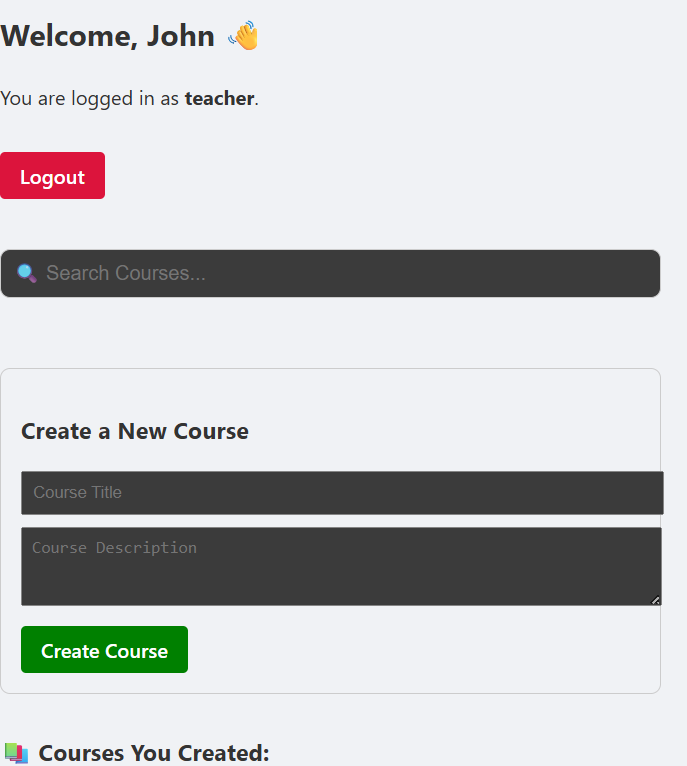
Login Page:

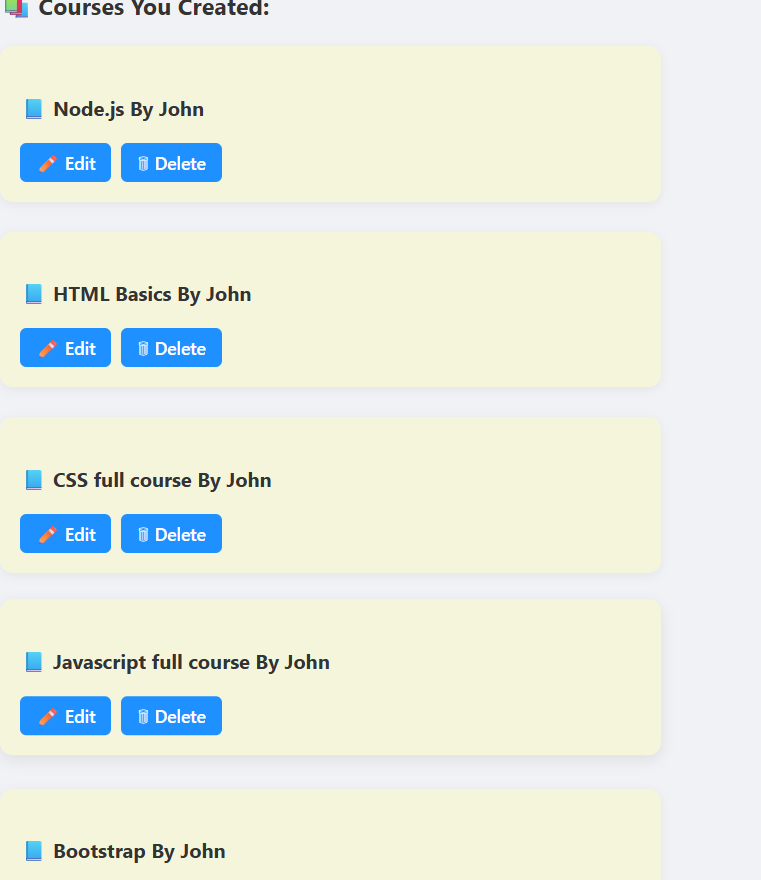


Registration Page:

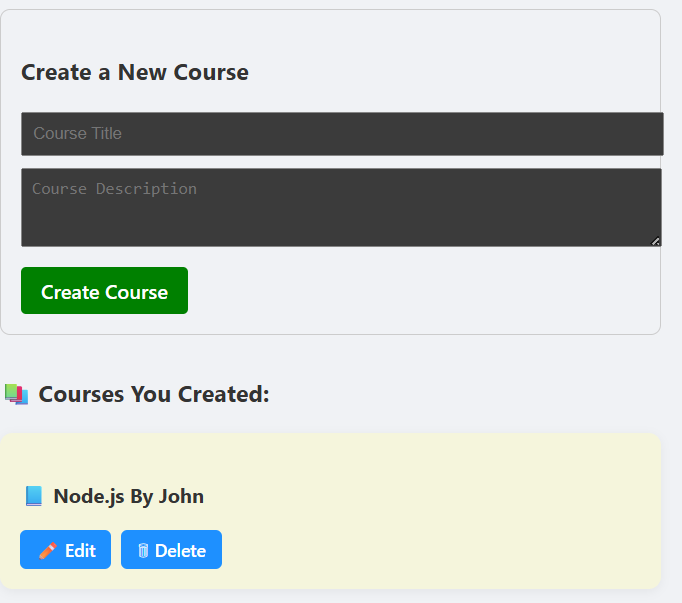


Teacher’s Page:

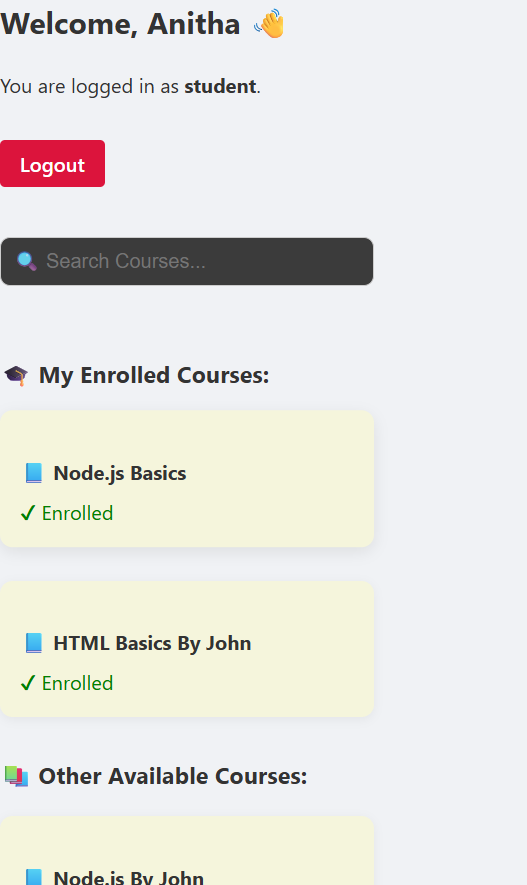




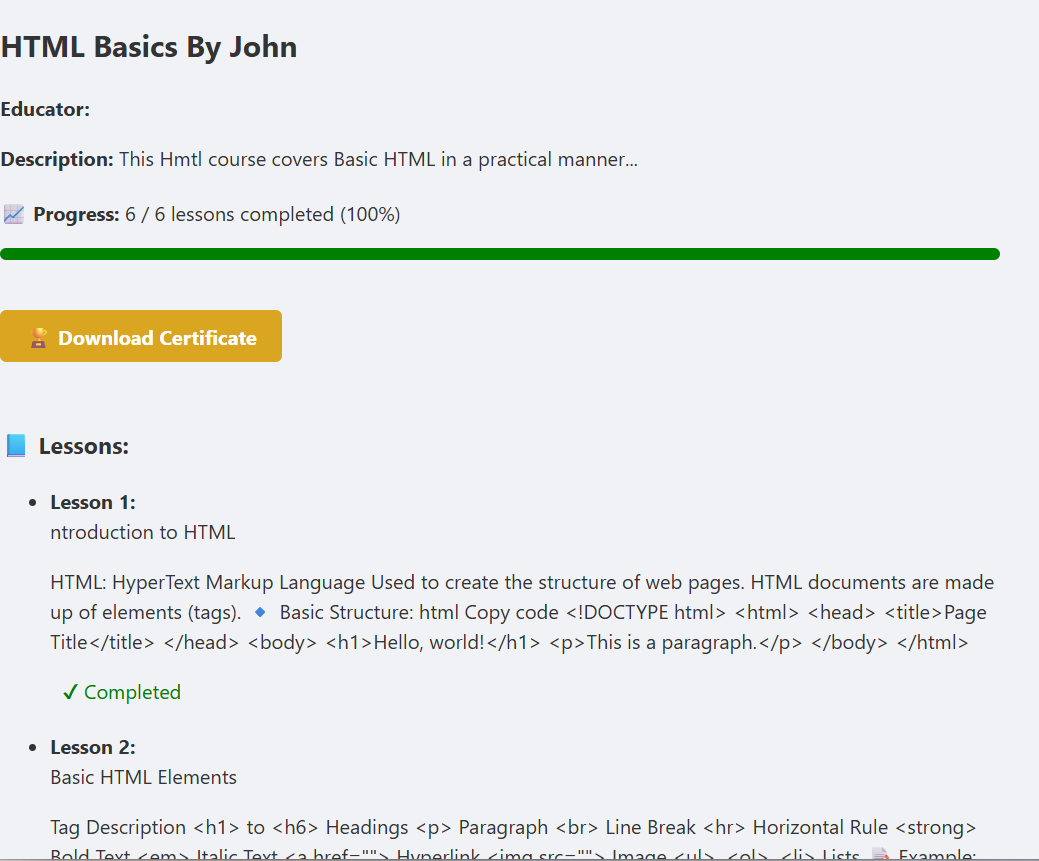
Adding or Deleting or Editing of courses(Teacher’s view):



Student’s Page:



Progress Bar(enrolled courses):



Certification(after completing the course):



Finally, for any further assistance, use the links below:

Demo Link:https://drive.google.com/file/d/1yToftQi69p2f5tBWtkM0vPtnUgt-yLKI/view?usp=drive\_link