Course Management Web Application

Introduction:

The Course Management Web Application is a Django-based system designed to facilitate the management of courses, including functionalities such as course addition, editing, viewing syllabi, and managing projects associated with each course. The application is built to be user-friendly and efficient, ensuring that course administrators can manage course-related information seamlessly.

Features:

User Authentication:

Login Functionality: Users can log in with their credentials verified against the database.

Dashboard: Upon successful login, users are redirected to the dashboard where they can access various features of the application.

Course Management:

Add Course: Users can add new courses by providing the course name and details. A unique course ID and course SK ID are generated automatically.

Edit Course: Users can edit existing courses, where the previous course record is marked as inactive, and a new record is created with the updated information.

Delete Course: Courses can be marked as deleted, making them inactive and preventing them from appearing in active course lists.

View Courses:

Course List: Users can view a list of all active and non-deleted courses along with their details.

Syllabus: Each course has an associated syllabus that can be viewed, detailing the topics covered in the course.

Projects: Users can also view projects associated with each course, providing a comprehensive view of all course-related activities.

Design and Layout:

User Interface:

Responsive Design: The application features a responsive design that ensures usability across various devices, including desktops, tablets, and mobile phones.

Clean and Modern Layout: Pages are designed with a clean and modern aesthetic, using a consistent color scheme and typography to enhance user experience.

Navigation:

Easy Navigation: The application provides easy navigation through a well-structured layout, making it simple for users to find and use different features.

Centralized Dashboard: The dashboard serves as a central hub from where users can access different functionalities related to course management.

Implementation Details:

Backend:

Django Framework: The application is built using Django, a high-level Python web framework that encourages rapid development and clean, pragmatic design.

MySQL Database: The application uses MySQL for storing and managing data, ensuring reliable and scalable database management.

Frontend:

HTML/CSS: The front end is implemented using HTML and CSS, ensuring a responsive and visually appealing interface.

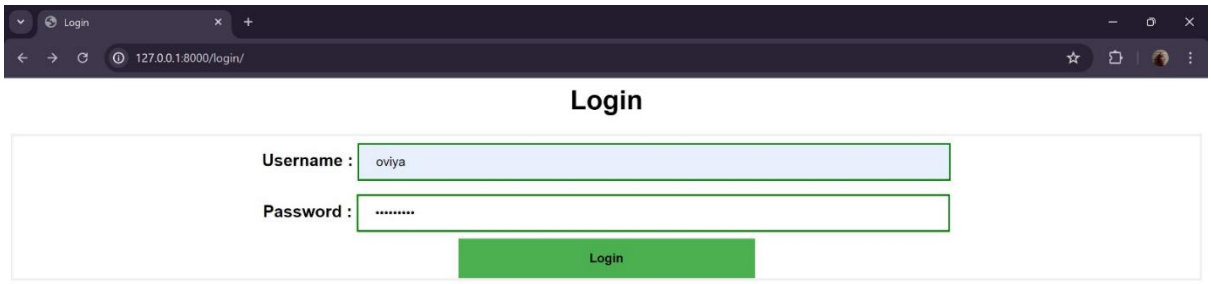
Django Templates: Django's templating engine is used to render dynamic content on the web pages, making the application interactive and user-friendly.

Security:

CSRF Protection: The application includes CSRF tokens in forms to protect against cross-site request forgery attacks.

Password Encryption: User passwords are securely stored using hashing techniques, ensuring that sensitive information is protected.

LOGIN PAGE:



Page Overview:

The login page is designed using Django, a high-level Python web framework.

The page features a dark theme with a prominent "Login" header.

Two input fields are provided for username and password.

A green "Login" button allows users to submit their credentials.

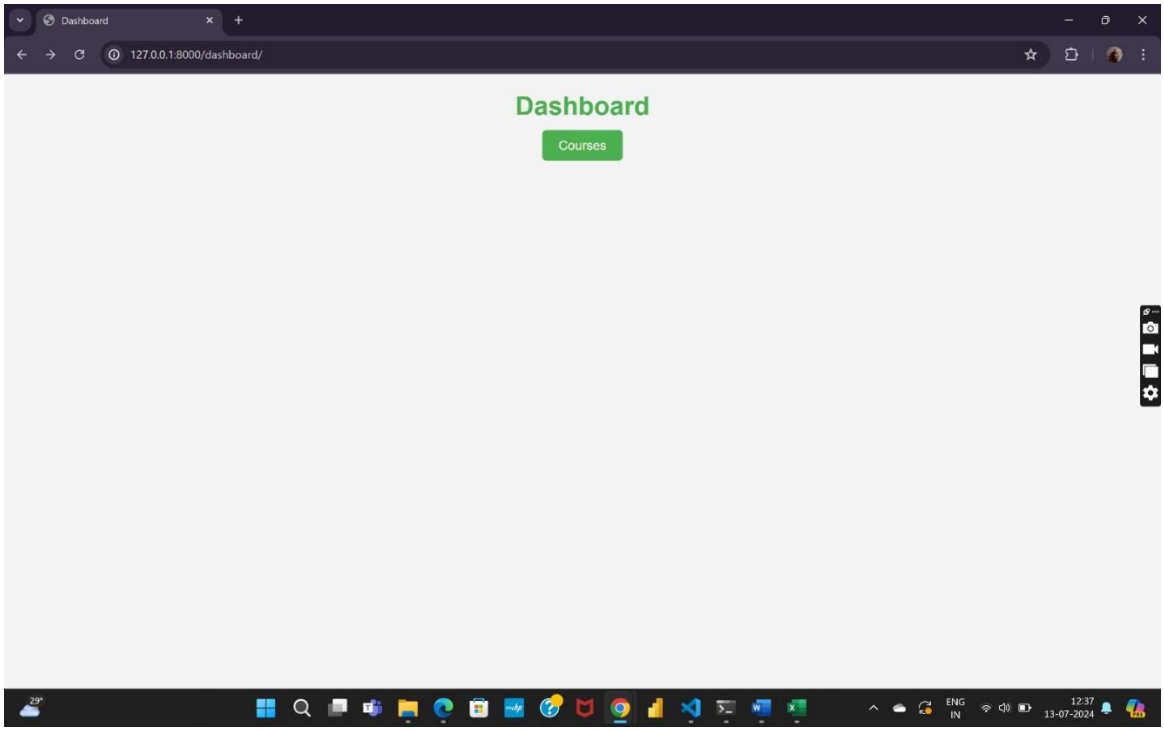
Components:

Username Field: The input field labelled "Username" has the placeholder text "oviya."

Password Field: The password input field displays obscured characters to ensure security.

URL: The page is hosted locally at '127.0.0.1:8000/login'.

DASHBOARD PAGE:



Page Overview:

The dashboard page is designed to display courses.

The page features a dark theme with a prominent “Dashboard” header.

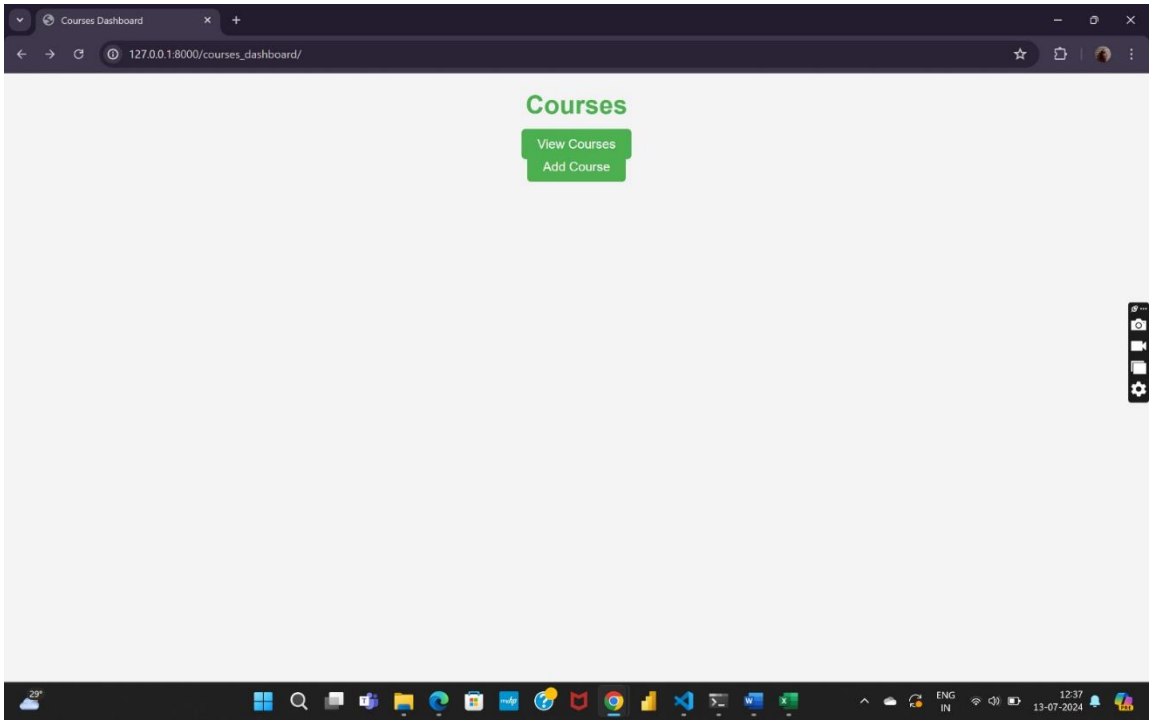
There is a section labeled “Courses”, which currently shows no listed courses.

Components:

Sidebar: The left side of the page contains a sidebar with icons, likely for navigation within the website or application.

URL: The page is hosted locally at ‘127.0.0.1:0000/dashboard’.

COURSES PAGE:



Page Overview:

The page is part of a course management system.

It features a white background with a header titled “Courses” in green font.

There are two options below the header: “View Courses” and “Add Course”.

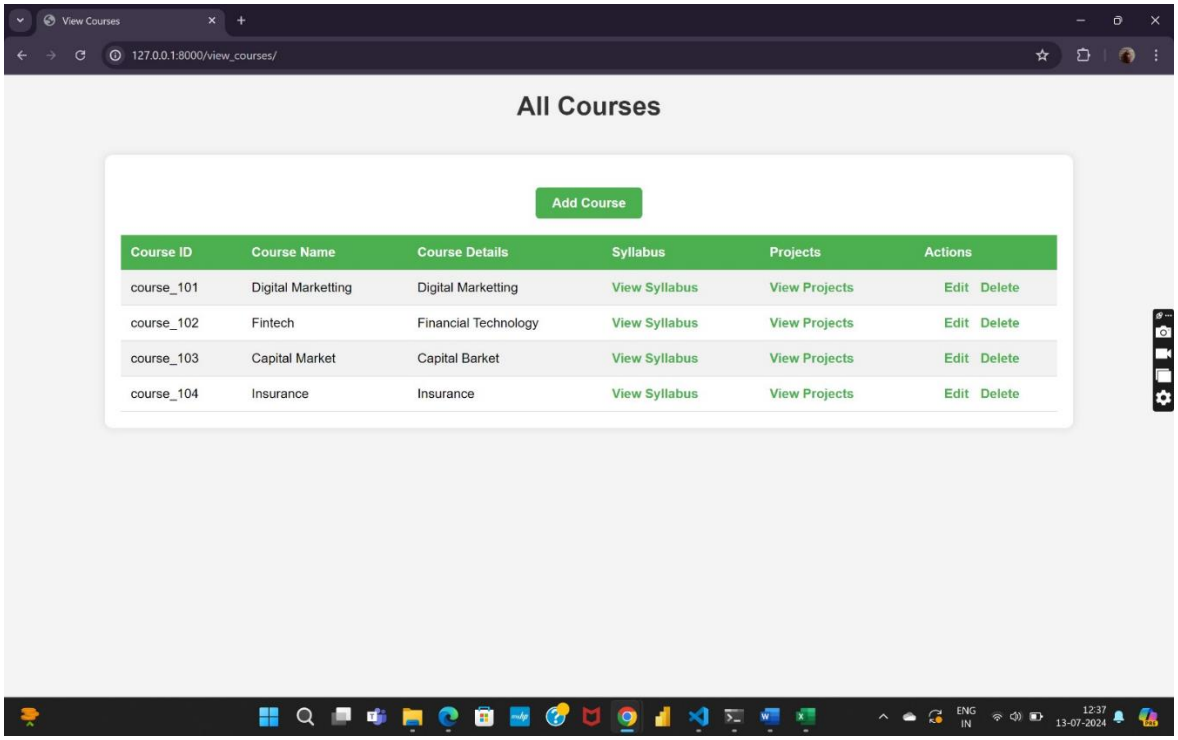
Components:

View Courses: This option likely allows users to see a list of available or enrolled courses.

Add Course: This option probably enables users to add new courses to the system.

Taskbar: The bottom taskbar shows the time (1:32 PM) and various notification icons, indicating a Windows operating system.

VIEW COURSES PAGE:



Page Overview:

The page is part of a course management system titled “All Courses”.

It features a table with columns labeled Course ID, Course Name, Details, Syllabus, Projects, Edit, and Delete.

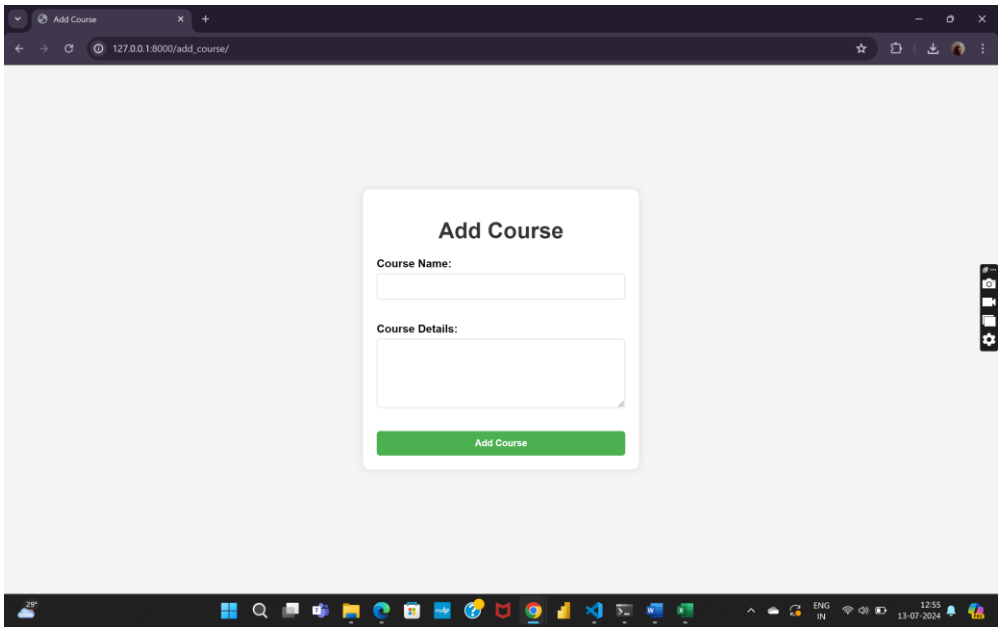
There are three courses listed: Digital Marketing (course_101), Fintech (course_102), and Insurance (course_103).

Components:

Course Details: Each course has buttons for viewing details and syllabus, viewing projects, and options to edit or delete the course.

Add Course: A green button labeled “Add Course” is available above the table for adding new courses.

ADD COURSES PAGE:



Header:

Title: The title of the page is "Add Course".

Main Content:

Form: The main content consists of a form with the following fields:

Course Name: A text input for entering the name of the course.

Course Details: A text input for entering the details of the course.

Submit Button: A button to submit the form and add the course to the database.

Aesthetics:

Color Scheme: The page uses a simple and clean color scheme. The background is light grey, and the form is displayed in a white container.

-Typography: The text is styled in a readable and professional manner, making it easy to fill out the form.

Layout:

Container: The form is contained within a central white box, which helps in focusing the user's attention on the form fields.

Form Fields: Each form field is separated by a subtle margin, enhancing readability and organization.

CSS Styles:

Container: The `.container` class styles the main content area with padding, rounded corners, and a box shadow for a polished look.

Form Fields: Each input and text area is styled with padding, borders, and rounded corners to make the form visually appealing and user-friendly.

Button: The submit button is styled with a green background, white text, and rounded corners. It changes color on hover to provide feedback to the user.

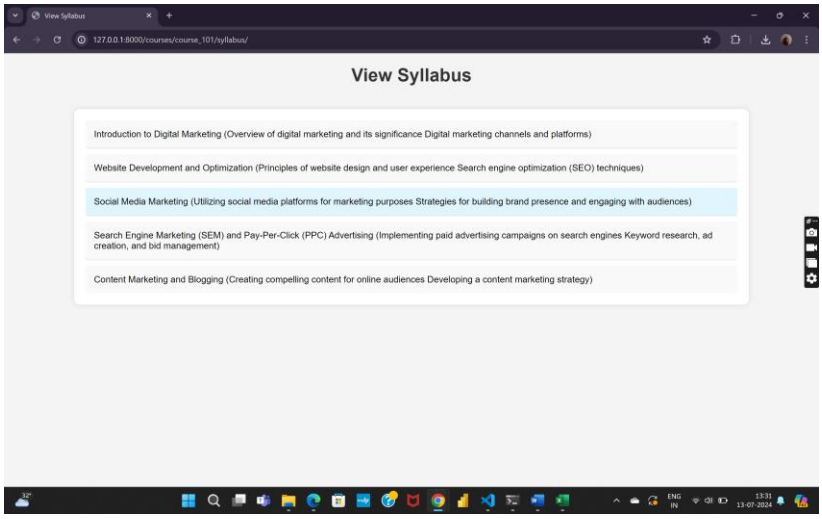
Backend Logic:

The backend logic handles the form submission by extracting the course name and course details from the POST request and inserting them into the database. The `add_course` view generates a unique course ID and course SK ID, sets the `update_by` field to the current user's username, and records the current date and time.

User Experience:

The page is designed to provide a straightforward and intuitive form for adding a new course. The clean layout and easy-to-read typography enhance the user experience, making it simple for users to input the necessary information.

VIEW SYLLABUS PAGE:



The "View Syllabus" page is a part of the Course Management Web Application designed to provide users with detailed information about the syllabus of a selected course. This feature ensures that users, such as students and instructors, have easy access to the course topics and can understand the curriculum structure.

Features:

Syllabus Display:

Course Topics: The page lists all the topics covered in the course, each accompanied by a brief description. This helps users quickly grasp the course content.

User-Friendly Layout: The information is presented in a clean, organized manner, making it easy to read and navigate.

User Interface:

View Syllabus Page:

The "View Syllabus" page showcases the syllabus of the selected course. Below is a screenshot of the page, highlighting its structure and elements:

Components

Header: The title "View Syllabus" clearly indicates the purpose of the page.

Syllabus List: Each syllabus item is displayed as a separate entry, with the topic name and a brief description.

Technical Details:

Backend Integration:

Database Query: The syllabus data is fetched from a MySQL database where the course information is stored.

Dynamic Rendering: Django templates are used to dynamically generate the syllabus list based on the selected course.

Frontend Design:

HTML/CSS: The page is designed using HTML for structure and CSS for styling, ensuring a responsive and aesthetically pleasing interface.

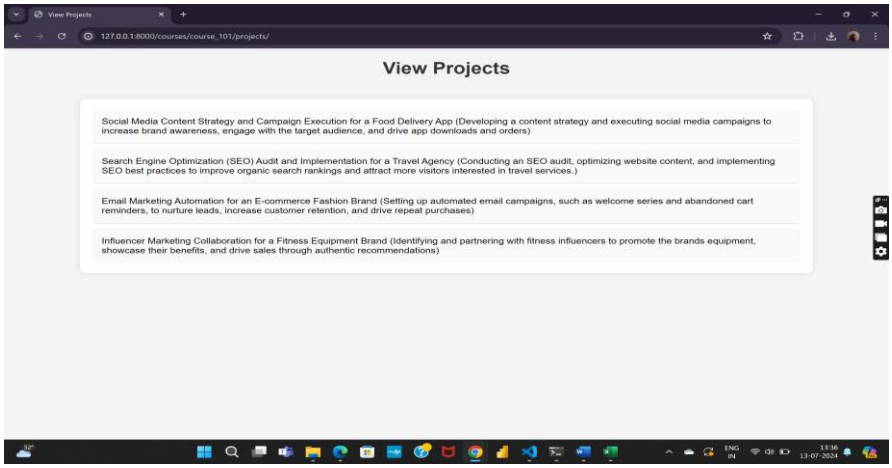
Django Templates: The Django templating engine is used to insert the dynamic content into the HTML structure, providing real-time data display.

Security:

Data Sanitization: Input data is sanitized to prevent SQL injection and other common security threats.

Access Control: Only authenticated users can access the page, ensuring that sensitive course information is protected.

VIEW PROJECTS PAGE:



The "View Projects" page is an integral part of the Course Management Web Application created using Django. This page is designed to provide users with detailed information about various projects associated with a specific course. It allows students and instructors to view and understand the projects they will be working on during the course.

Features

Project Display

Project Titles and Descriptions: The page lists all the projects related to the course, each with a concise title and description to give users a clear understanding of the project objectives.

Organized Layout: The projects are presented in an organized manner, making it easy for users to read and comprehend the information.

User Interface

View Projects Page

The "View Projects" page displays the list of projects for a selected course. Below is a screenshot of the page, illustrating its design and layout:

Components

Header: The title "View Projects" clearly indicates the content and purpose of the page.

Project List: Each project is displayed as a separate entry, with the project name highlighted and a brief description provided below.

Technical Details

Backend Integration

Database Query: The project data is fetched from a MySQL database where the course information is stored.

Dynamic Rendering: Django templates are used to dynamically generate the project list based on the selected course.

Frontend Design

HTML/CSS: The page is designed using HTML for structure and CSS for styling, ensuring a responsive and visually appealing interface.

Django Templates: The Django templating engine is used to insert the dynamic content into the HTML structure, providing real-time data display.

Security

Data Sanitization: Input data is sanitized to prevent SQL injection and other common security threats.

Access Control: Only authenticated users can access the page, ensuring that sensitive project information is protected.

CONCLUSION:

The Course Management Web Application is a robust and user-friendly solution designed to streamline the management of educational content. Developed using Django, the application offers essential features such as adding courses, viewing syllabi, and managing projects, making it a valuable tool for instructors and students alike.

Key Strengths:

1. Dynamic Course Management: Instructors can easily create and update courses, ensuring that the application can adapt to new educational needs.

2. Clear Syllabus Presentation: The application provides a structured and detailed view of course syllabi, helping students understand the course content and structure.

3. Comprehensive Project Management: By listing detailed project information, the application supports students in managing their coursework effectively.

Technical Excellence:

Backend Reliability: Built on Django with a MySQL database, the backend ensures data integrity and security.

Frontend Usability: The clean and intuitive user interface, designed with HTML, CSS, and JavaScript, enhances the user experience.

Security Measures: Robust validation and user authentication mechanisms protect against common security threats.

In conclusion, this application is a comprehensive solution for managing course-related activities, enhancing both teaching and learning experiences. With its strong foundation and potential for future enhancements, it is well-positioned to meet the evolving needs of educational institutions.