LabLens : Microscope Booking System

Submitted in partial fulfillment of the requirements of the degree of

Bachelor of Engineering (Information Technology)

By

Sannidhi Kailaje Roll No – 22

Department of Information Technology

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY,

Chembur, Mumbai 400074

(An Autonomous Institute, Affiliated to University of Mumbai)

April 2024

Contents :

LabLens

Name of student	Sannidhi Kailaje
Class_Roll no	D15A_22
D.O.P	3/04/25
D.O.S	17/04/25
Sign and Grade	

Title : LabLens - Microscope Booking System

Project Description :

The Microscope Booking System is a full-stack web application designed to simplify the process of reserving microscopes in academic and research labs. It allows students and researchers to browse available microscopes, book slots in real-time, and manage their bookings through a personal dashboard. The system provides secure login/registration with OTP-based verification and ensures that no conflicting bookings are allowed. Built using Flask, MongoDB, and frontend technologies like HTML, CSS, JavaScript, and Bootstrap, the platform emphasizes usability, performance, and responsiveness.

Requirement gathering :

The requirements were derived by analyzing common challenges in laboratory slot management through user interviews with lab coordinators and students. Key pain points included lack of centralized booking, double-booking issues, and inefficient manual systems. Functional requirements focused on login, booking management, real-time slot validation, and booking history. The primary goal was to identify pain points in existing systems, such as poor UI/UX, lack of real-time tracking, and absence of categorization or visual indicators. Both individual users and small team workflows were considered to ensure the solution remains flexible and scalable. Functional requirements included CRUD operations for tasks, visual progress tracking, and a responsive design, while Non-functional needs emphasized security (OTP verification), mobile responsiveness, and scalability. Feedback from early users helped shape the core features and user experience of Trackify.

System Requirements :

Hardware:

- Processor: Intel i3/i5 or AMD Ryzen (dual-core+)
- RAM: 8 GB minimum
- Storage: 1 GB available space
- Internet: Required for OTP & MongoDB Atlas

Software:

- OS: Windows/macOS/Linux
- Python 3.8+
- MongoDB Atlas or Local MongoDB
- Visual Studio Code
- Git

Technologies Used :

Layer	Technology
Frontend	HTML, CSS, Bootstrap, JS
Backend	Python (Flask), Flask-Login
Database	MongoDB
Auth	OTP via SMTP (Gmail)
Hosting	Render (Backend)
Analytics	Google Analytics (GA4)

Setup Instructions :

To set up the Microscope Booking System, several components need to be configured properly. First, ensure that Python 3.8 or higher is installed from the official Python website. During installation on Windows, it's important to check the option to add Python to the system PATH. After installing, you can verify it using python --version and pip --version. It's recommended to create a virtual environment using python -m venv venv to manage dependencies separately. This environment can be activated with venv\Scripts\activate on Windows or source venv/bin/activate on macOS/Linux.

Once Python is set up, install the necessary packages using a requirements.txt file, which includes Flask, Flask-Login, Flask-SocketIO, pymongo, dnspython, and python-dotenv. These packages support user authentication, real-time updates, and MongoDB integration.

The system uses MongoDB as its backend database, which can be set up either locally or via MongoDB Atlas. For local setups, MongoDB Community Edition should be installed and started on the default port. For cloud setups, MongoDB Atlas allows you to create a free cluster, set up a user, whitelist your IP, and generate a connection string. This string should be securely stored in a .env file along with the Flask SECRET_KEY and email credentials if OTP functionality is used.

After all configurations, the application can be launched using python app.py, and accessed via http://localhost:5000. If deploying to a production server like Render, Gunicorn is used to run the Flask app. With this setup, the Microscope Booking System can run seamlessly, supporting real-time bookings, user authentication, and data persistence via MongoDB.

Backend Setup

1. Navigate to your project folder:
   cd microscope-booking-system

2. Create a virtual environment:
   python -m venv venv

3. Activate the virtual environment:
   venv\Scripts\activate

4. Install required Python dependencies:
   pip install -r requirements.txt

5. Start the Flask development server:
   python app.py

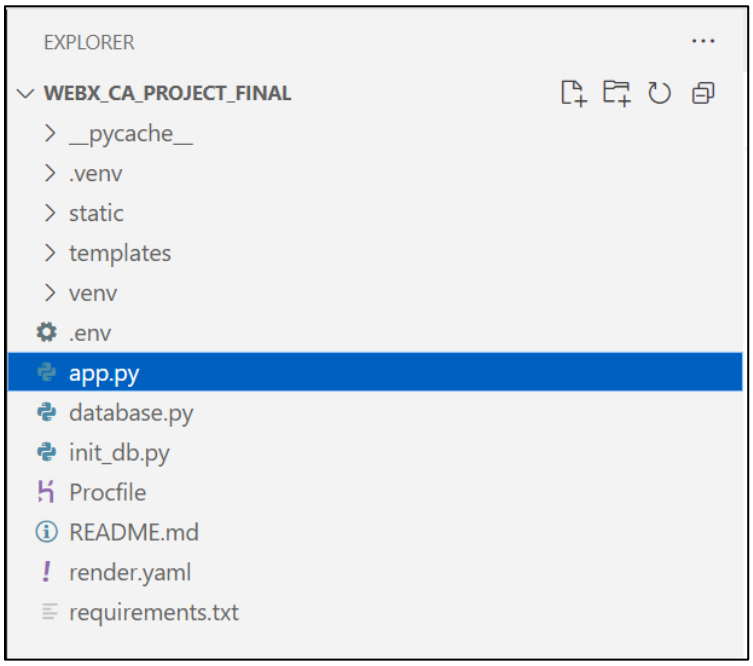
Frontend Setup

The frontend of the Microscope Booking System is built using standard web technologies — HTML5, CSS3, JavaScript, and Bootstrap — and is rendered through Flask templates.

Access the Website

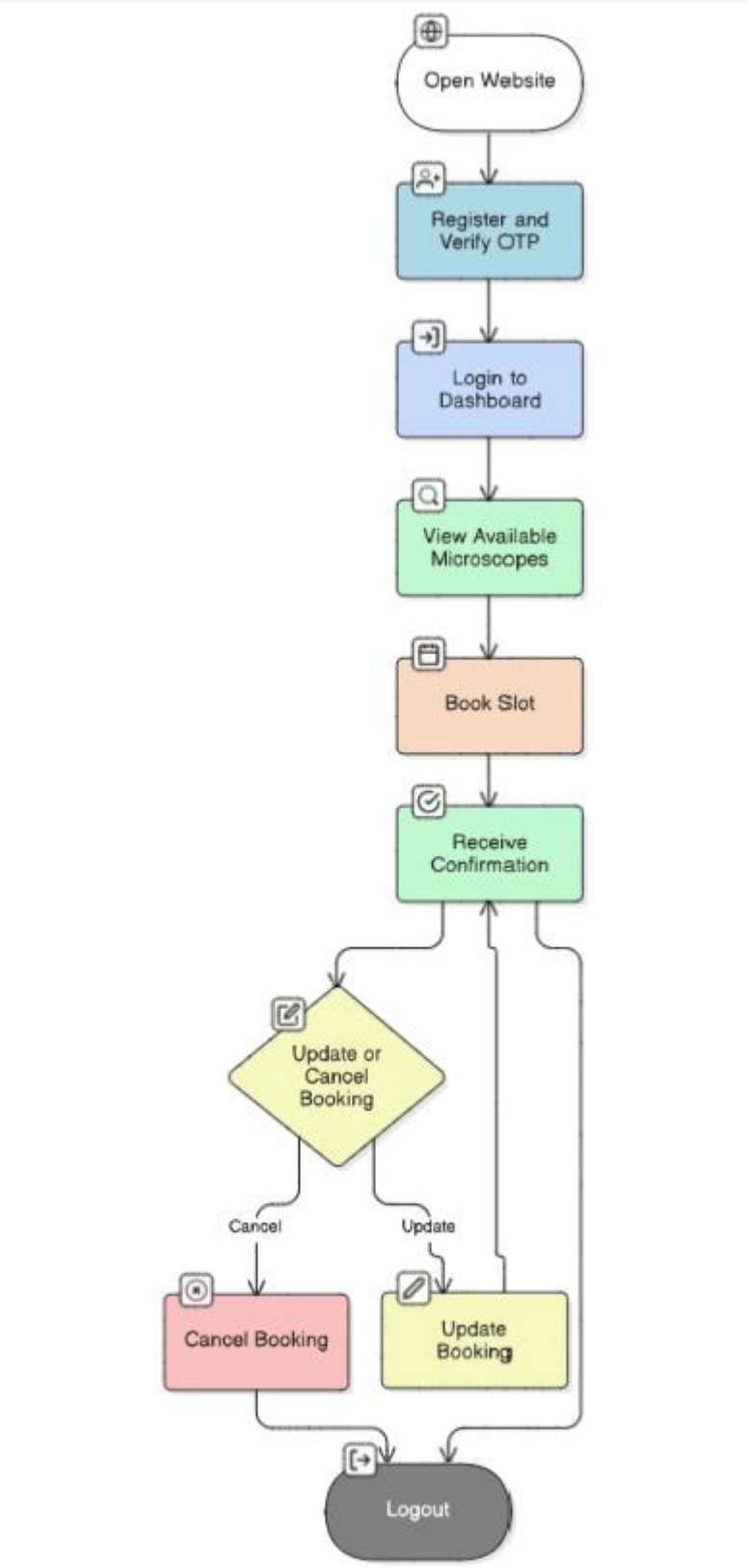
- Once the server starts, visit: **http://localhost:5000**
- hosted link: **https://webx-miniproject.onrender.com/**

Project Structure:

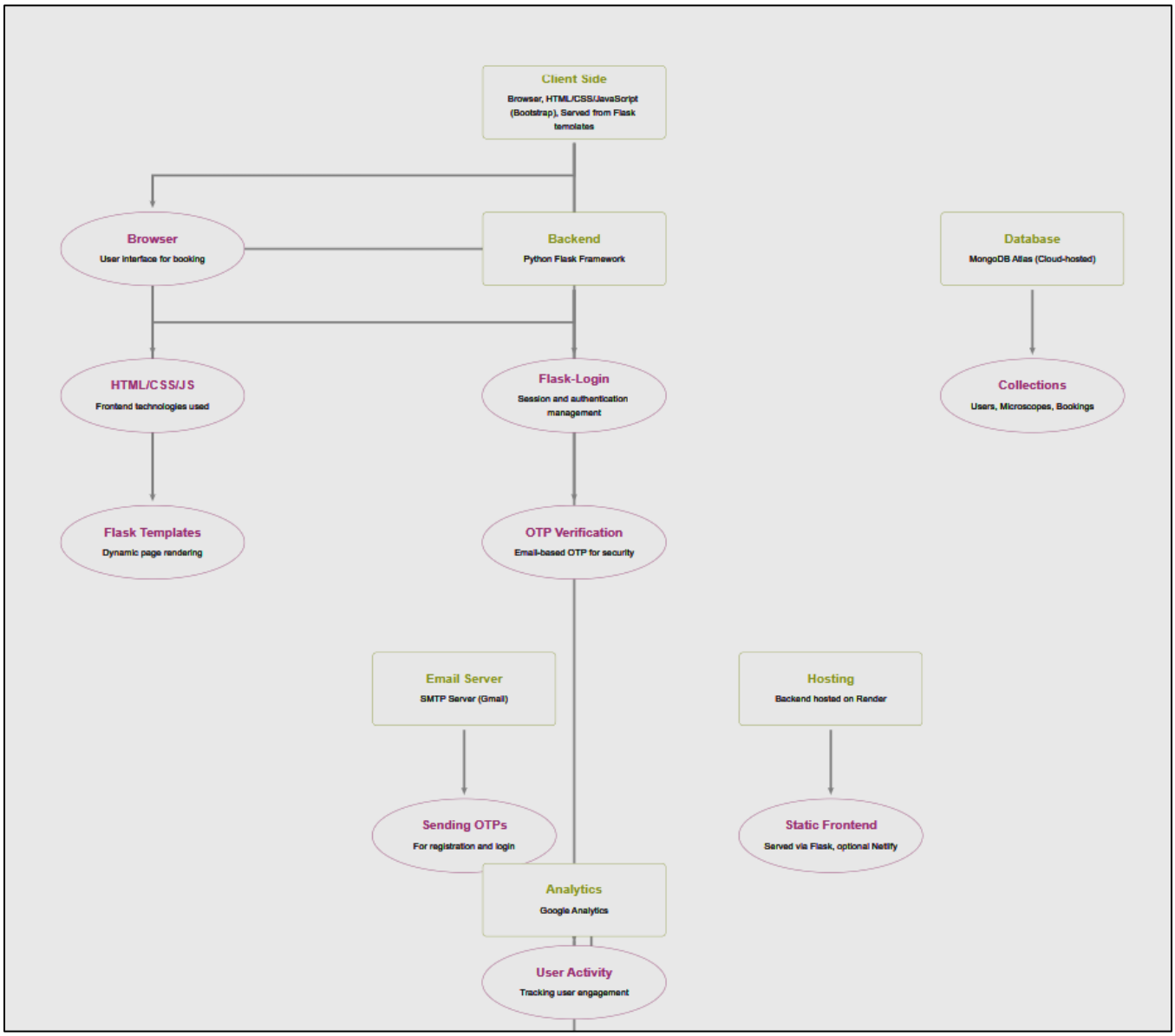


Architectural Diagrams :

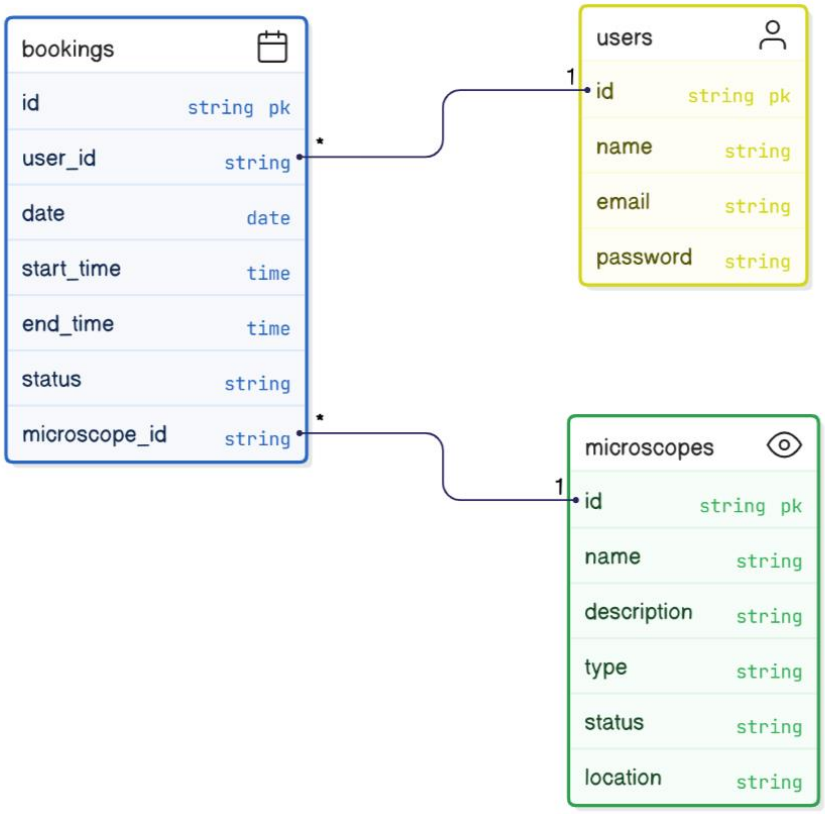
a) Web Flow Diagram -



b) System Architecture Diagram :



c) UML diagram :



Features Implemented :

- **User Registration & Login**
  Users can create accounts and securely log in to manage their bookings. Authentication ensures that only verified users access the system.

- **Email-based OTP Verification**
  An OTP is sent to the user□s email during registration to confirm identity, adding a layer of security and preventing spam accounts.

- **Dashboard with Real-Time Booking Overview**
  The dashboard provides a summary of upcoming bookings, microscope availability, and past activity, allowing users to manage everything in one place.

- **Slot Booking, Updating, and Cancellation**
  Users can easily book available time slots, modify existing bookings, or cancel them when needed, all in a few clicks.

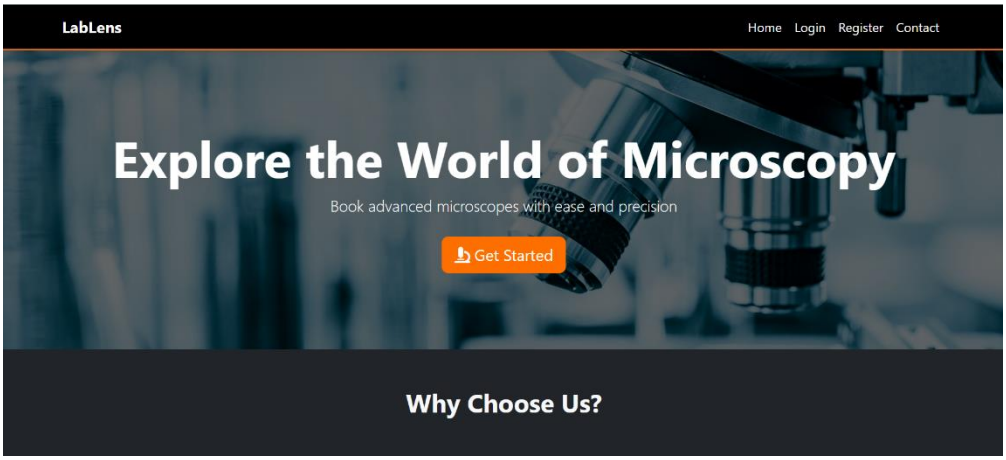
- **Microscope Listing with Details**
  Each microscope is displayed with its name, type, description, and location, helping users select the right one for their needs.

- **Secure Session Management**
  User sessions are handled securely, with auto logout on inactivity and access protection for sensitive actions.

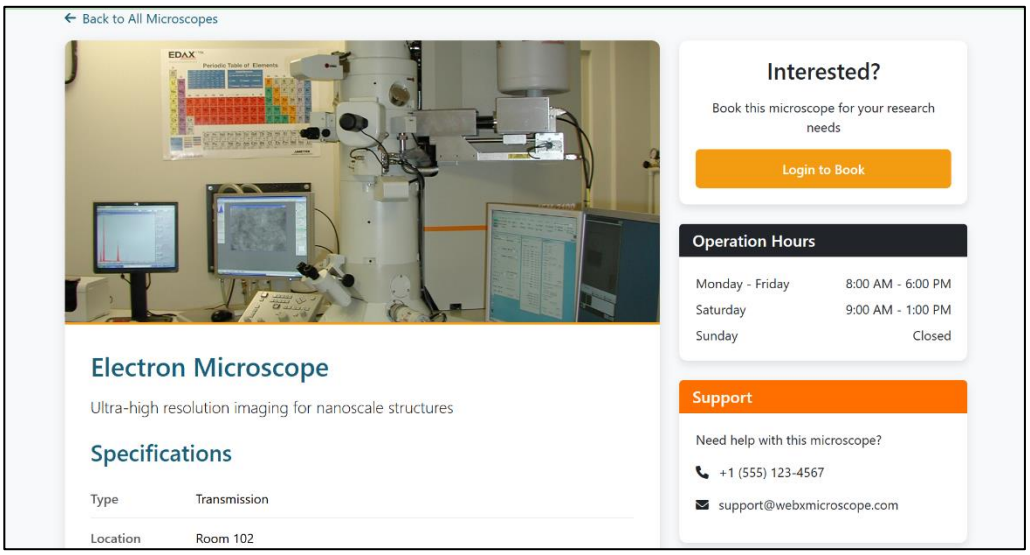
- **Google Analytics Integration**
  Site usage is tracked using Google Analytics to monitor traffic, user behavior, and booking trends for future improvements.

Screenshots of implementation :

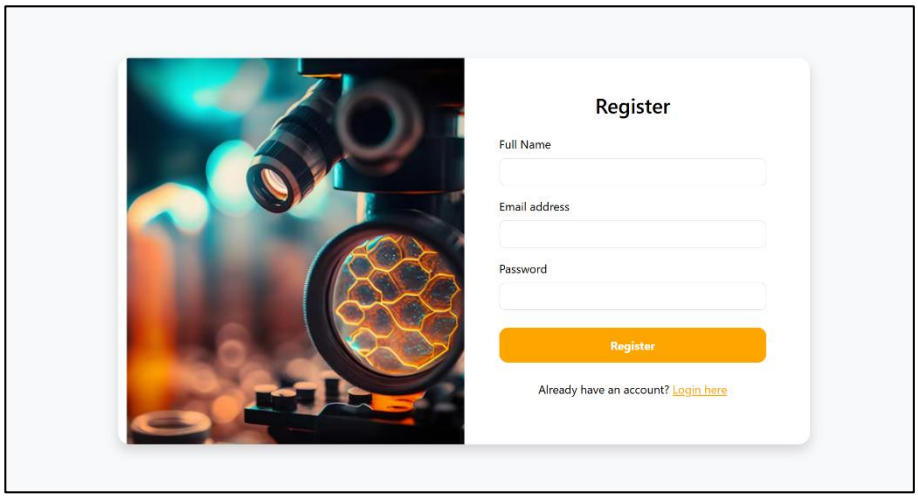
Home Page -



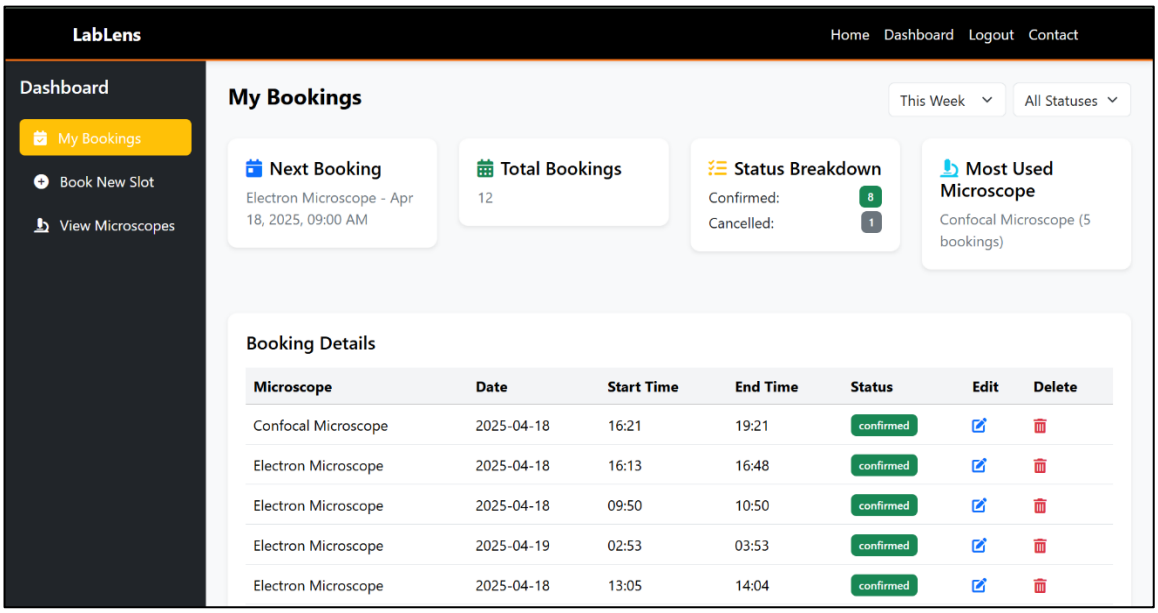
Microscope Details Page-



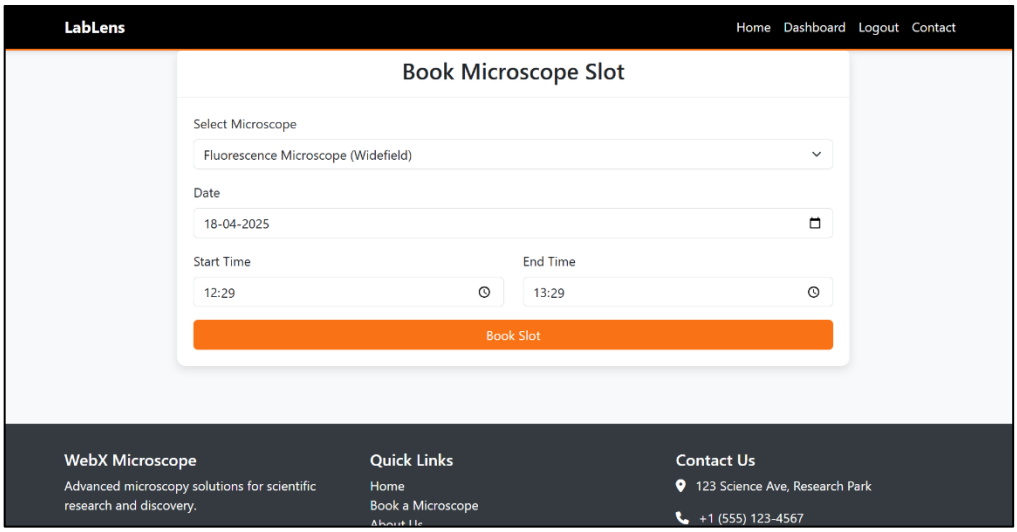
Registration page -



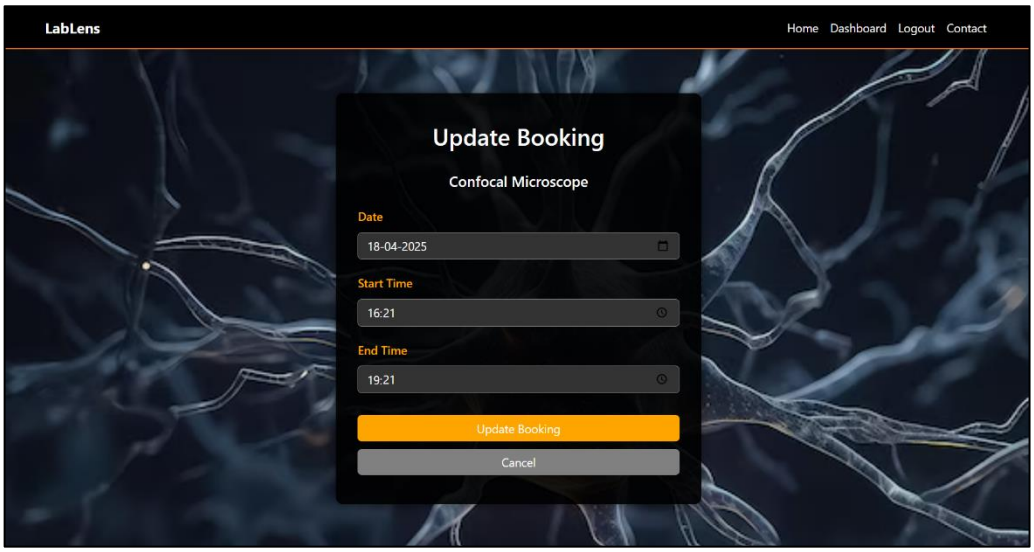
User Dashboard -



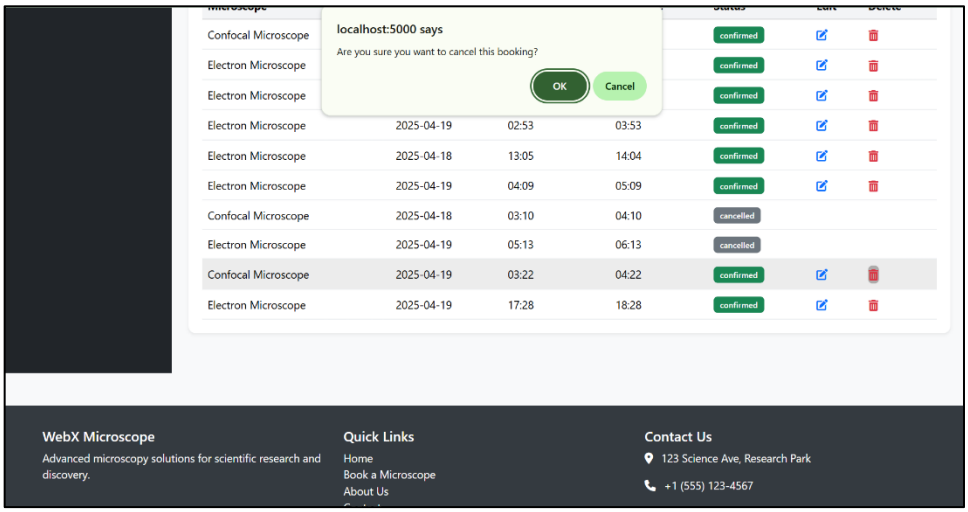
Slot Booking page -



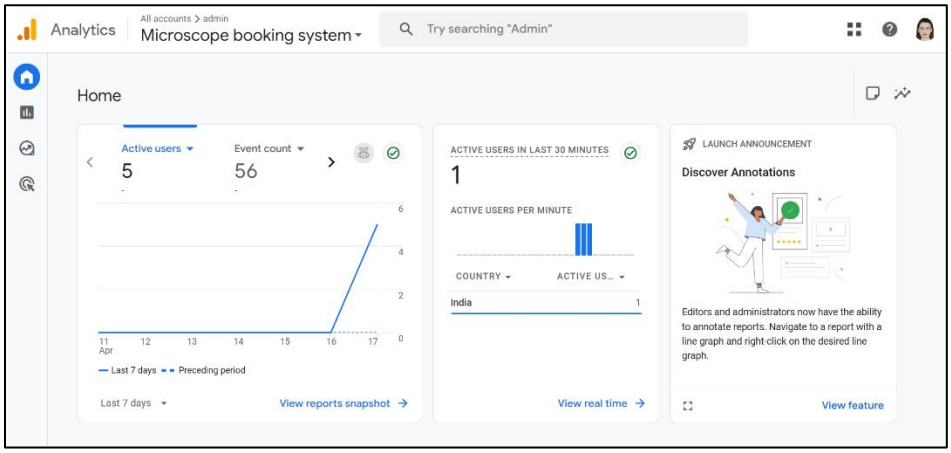
Update booking Page -

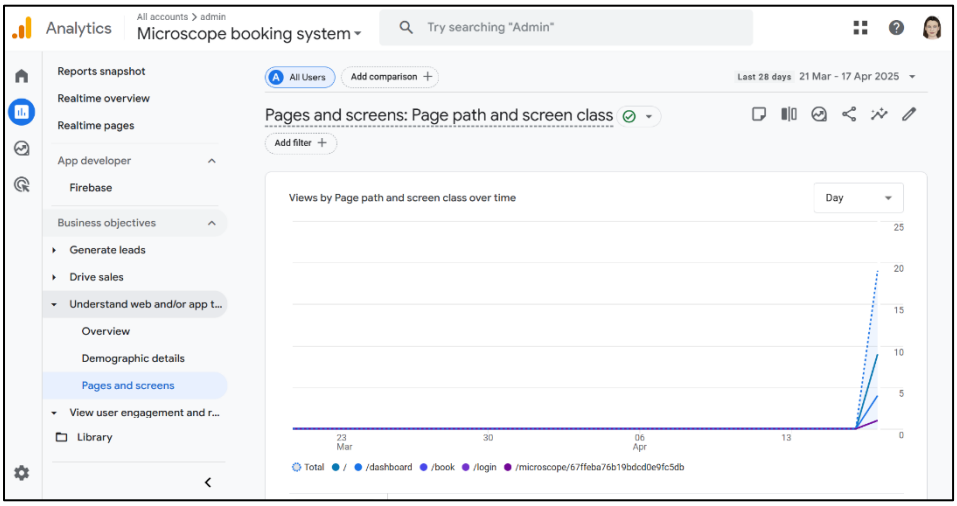


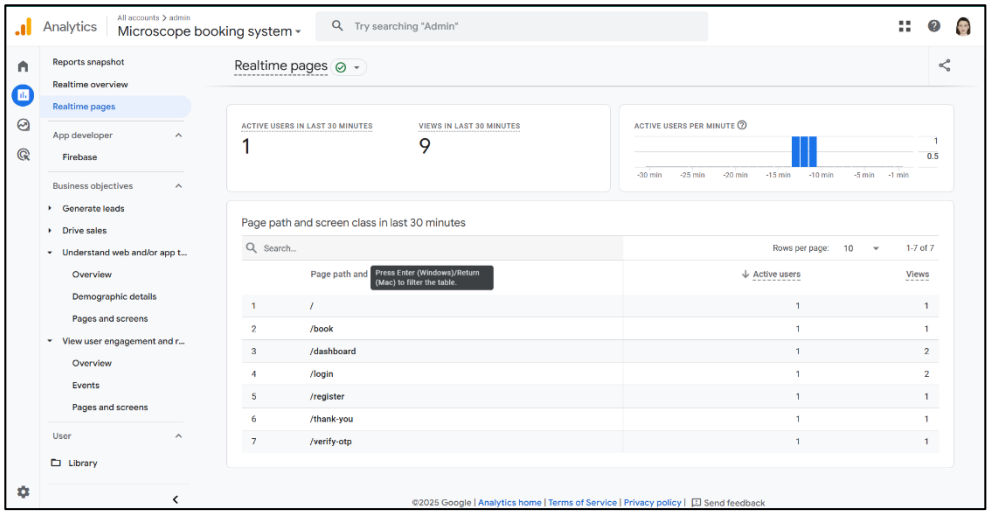
Delete Booking Page:

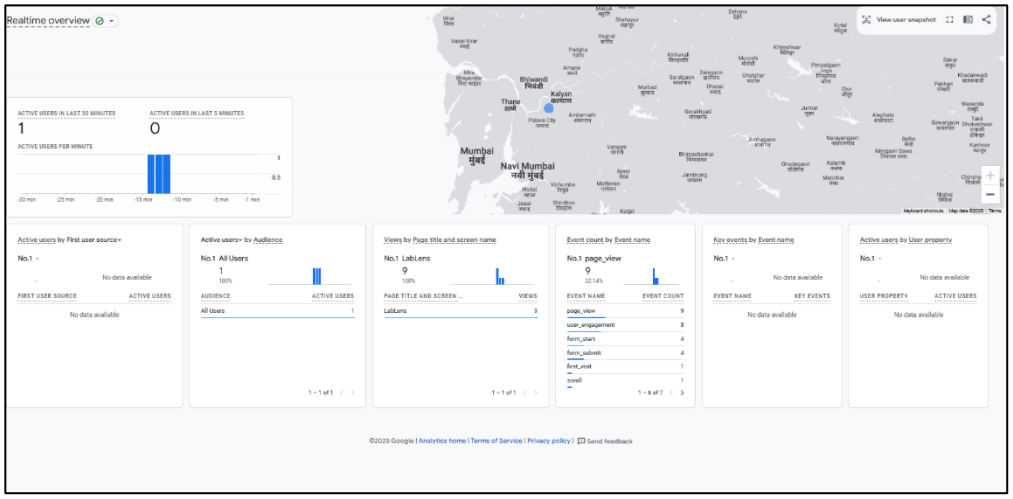


Google Analytics :



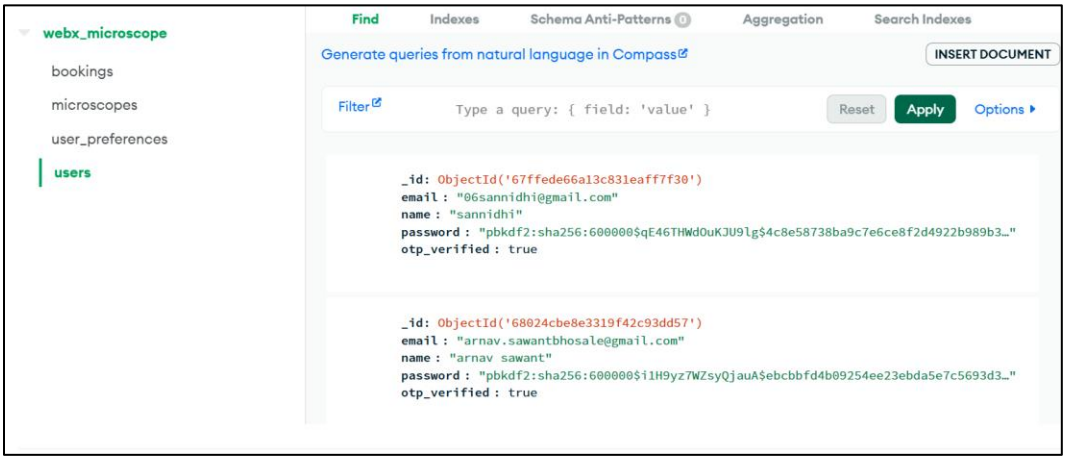




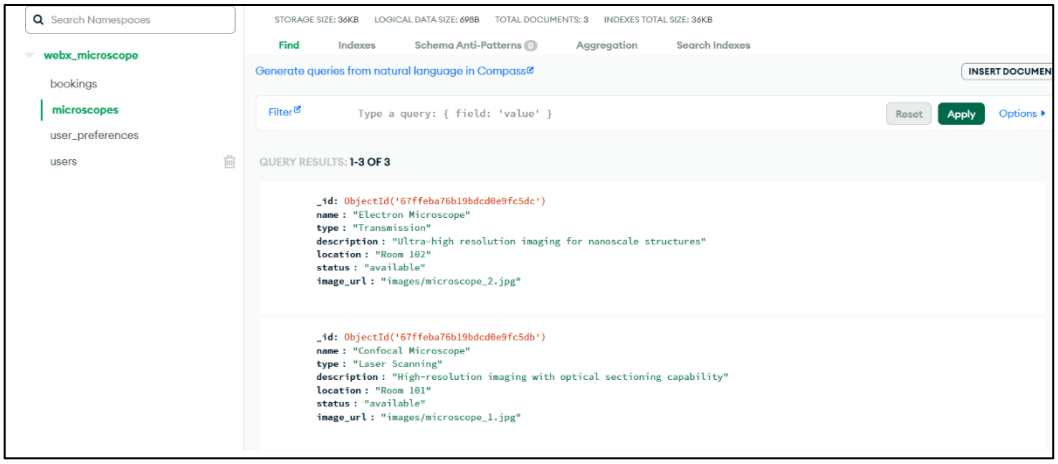


Database (MongoDB) -

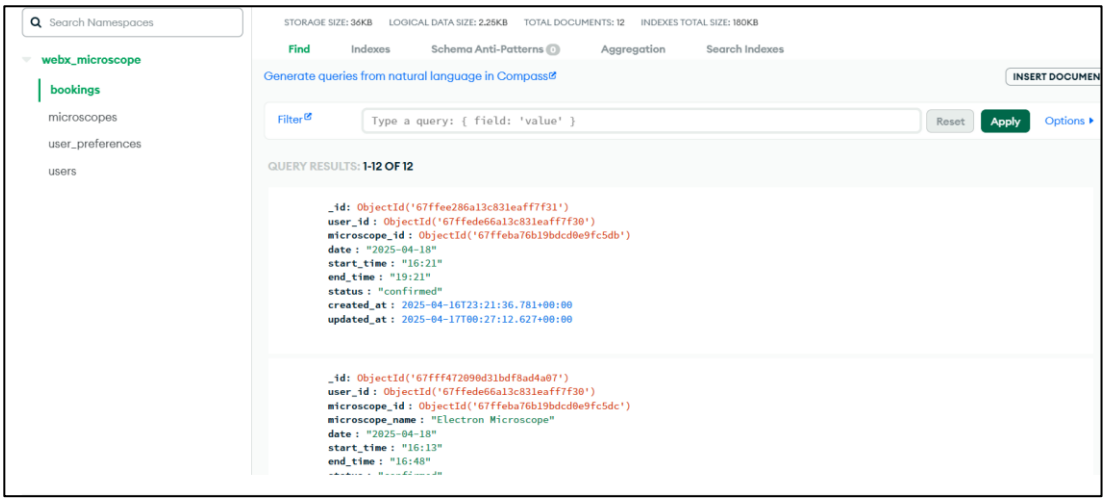
users



microscopes:



Bookings:



Github Link : https://github.com/Skailaje/webx_miniproject.git

Future Scope

The Microscope Booking System has significant potential for expansion into a comprehensive digital lab management platform. Future enhancements may include the integration of AI-based slot suggestions, which could recommend optimal time slots based on historical booking trends and user preferences. The system can also incorporate a calendar view with drag-and-drop slot management for better visual scheduling. To enhance communication, email and SMS notifications for booking confirmations, reminders, and cancellations could be implemented.

A dedicated admin dashboard could be developed further to provide real-time analytics on microscope usage, peak hours, user activity, and maintenance needs. Integration with Google Calendar, institutional login systems (SSO), and QR code scanning for on-site check-ins are also feasible additions. Long-term vision includes mobile application support and multi-language accessibility to broaden user reach and improve the overall booking experience in academic and research environments.

Conclusion

The **Microscope Booking System** is a modern, full-stack web application designed to streamline microscope slot booking in academic and research settings. Built using **Flask** and **MongoDB**, the system ensures secure login, real-time booking management, and user-friendly features such as OTP verification and a dynamic dashboard.

The project setup involves configuring essential components like Python, Flask, MongoDB (Atlas or local), and integrating email services for OTP-based authentication. The UI is built using HTML, CSS, JavaScript, and Bootstrap, providing a clean and responsive experience across devices.

Through features like real-time booking validation, session management, and booking history, the platform minimizes double bookings and improves lab efficiency. Its successful implementation demonstrates practical knowledge in **web development, user experience design, and backend integration**, making it a scalable and impactful tool for institutions aiming to digitize their lab resource management.