Legal Case Management System

with DevOps Implementation (CI/CD Pipeline)

Course Name: Data Analytics

Institution Name: Medicaps University – Datagami Skill Based Course

Student Name(s) & Enrolment Number(s):

Sr no	Student Name	Enrolment Number
1.	Jogendar Das Bairagi	EN22ME304045
2.	Roshan Bankar	EN22EL301072
3.	Vivek Gangrade	EN22CS3011098
4.	Maahi Maheshwari	EN22ME304057
5.	Shubhangi Prajapat	EN22ME304094

Group Name : 04D10

Project Number : DO-17

Industry Mentor Name :

University Mentor Name :

Academic Year : 2025-26

Content

1 .Problem Statement & Objectives

1.1 Problem Statement

In today's fast-evolving legal landscape, case management primarily depends on manual processes for maintaining, updating, and tracking case records. These manual systems are prone to critical challenges that compromise efficiency and data integrity.

Key problems identified in traditional legal case management include:

- Data loss due to improper storage and lack of backup mechanisms
- Significant delays in retrieving and updating case information
- Human errors introduced during manual data entry and updates
- Lack of structured format for storing case-related information
- Absence of automated testing and deployment processes
- Difficulty in scaling the system as case volume increases

To address these problems, an automated Legal Case Management System has been developed. The system leverages DevOps practices — specifically Continuous Integration and Continuous Deployment (CI/CD) — to deliver a faster, more reliable, and efficient solution for managing legal case data.

1.2 Project Objectives

The following objectives were defined to guide the development of this project:

#	Category	Objective
1	**Data Management**	To manage legal case records efficiently using a structured, digital format
2	**Process Automation**	To automate repetitive manual processes to reduce human error and time
3	**Data Security**	To provide secure data storage with appropriate access controls
4	**CI/CD Pipeline**	To implement a complete CI/CD pipeline for automated build, test, and deploy
5	**Reliability**	To improve overall system reliability, uptime, and performance
6	**Deployment**	To enable faster and consistent deployment of application updates

1.3 Scope of the Project

The scope of this project is defined around the digital transformation of legal case record management. The system covers the following functional boundaries:

- Storing and retrieving legal case information in a structured digital format
- Updating case records with version-controlled changes
- Automated deployment of updates through a CI/CD pipeline
- Containerization of the application using Docker
- Orchestration of containers using Kubernetes

The following areas are outside the current scope but are identified as potential future enhancements: advanced analytics dashboards, real-time notification systems, mobile application interfaces, and deep cloud-native integrations.

2 .Proposed Solution

2.1 Key Features

The Legal Case Management System provides the following core capabilities:

Feature	Description
Case Record Management	Create, read, update, and delete (CRUD) legal case entries
Data Storage & Retrieval	Persistent storage with fast query and retrieval capabilities
CI/CD Automation	Automated build, test, and deployment pipeline on every code change
User-Friendly Interface	Intuitive frontend for entering and viewing case data
Secure Data Handling	Input validation, secure connections, and controlled access
Continuous Integration	Automated unit and integration testing on each commit

2.2 Overall Architecture & Workflow

The system is built on a layered architecture that separates concerns across four major components: the Frontend, Backend, Database, and CI/CD Pipeline.

2.2.1 Architecture Components

- Frontend — Provides the user interface where legal professionals enter, view, and manage case data. Built with a responsive web framework to ensure usability across devices.
- Backend — Handles all business logic, API endpoints, and request/response processing. Acts as the intermediary between the frontend and the database.
- Database — Stores all case-related information in a structured format. Ensures data integrity and supports efficient queries.
- CI/CD Pipeline — Automatically triggers on code changes to build, test, and deploy the latest application version without manual intervention.

2.2.2 Application Workflow

The workflow for a typical user interaction is as follows:

- User submits a request (e.g., add a case) via the Frontend interface
- The request is transmitted to the Backend over a secure API connection
- The Backend processes the data, applies business logic, and communicates with the Database
- The Database stores or retrieves the relevant case data and returns results
- The Backend sends a response back to the Frontend for display

2.2.3 CI/CD Workflow

- Developer pushes code changes to the GitHub repository
- CI pipeline triggers automatically — code is compiled and unit tests are executed
- If tests pass, Docker builds a new container image
- The image is pushed to a container registry
- Kubernetes pulls the new image and performs a rolling deployment
- The application is live with zero downtime

2.3 Tools & Technologies Used

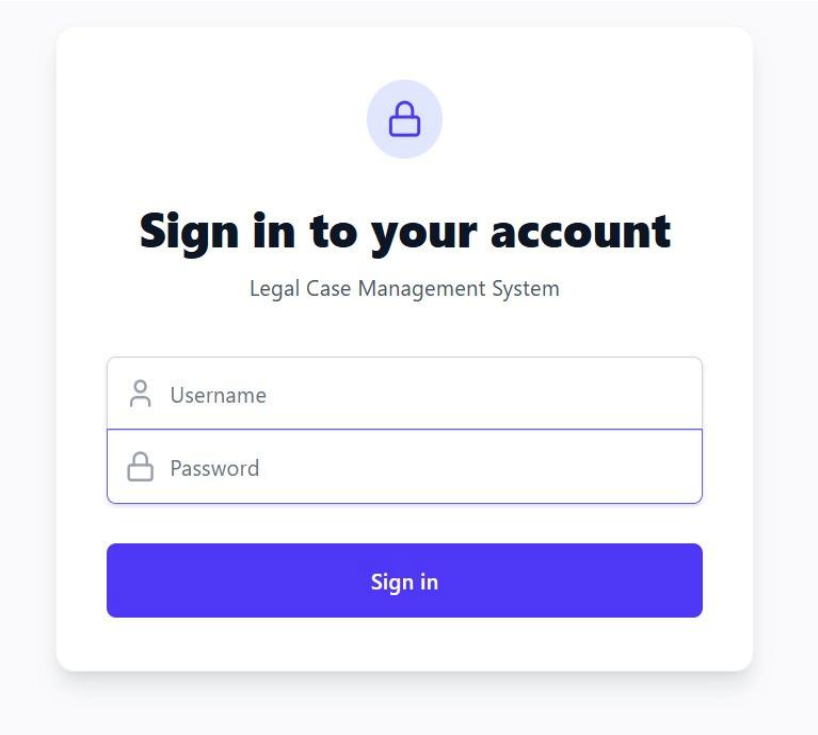
Category	Tool / Technology	Purpose
Version Control	GitHub	Source code management, collaboration, and CI/CD trigger
Containerization	Docker	Package the application and dependencies into portable containers
Orchestration	Kubernetes (K8s)	Manage, scale, and deploy containerized applications
CI/CD Automation	GitHub Actions / Jenkins	Automate the build, test, and deployment pipeline
Programming Language	Python / JavaScript	Backend application logic and API development
Web Interface	HTML / CSS / JavaScript	Frontend user interface for case data interaction
Database	PostgreSQL	Structured storage of legal case records
Testing	Unit & Integration Tests	Automated verification of application functionality

3 .Results & Output

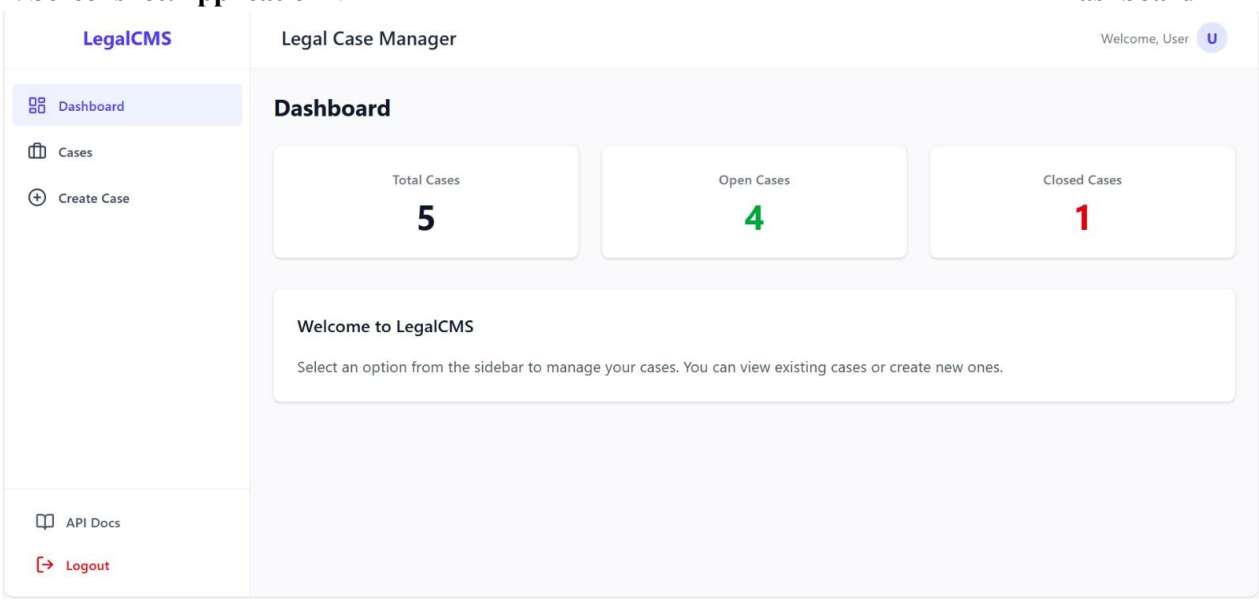
3.1 Screenshots / Outputs

The following sections describe the expected outputs and interface screens of the Legal Case Management System. Project-specific screenshots should be inserted in the designated areas below.

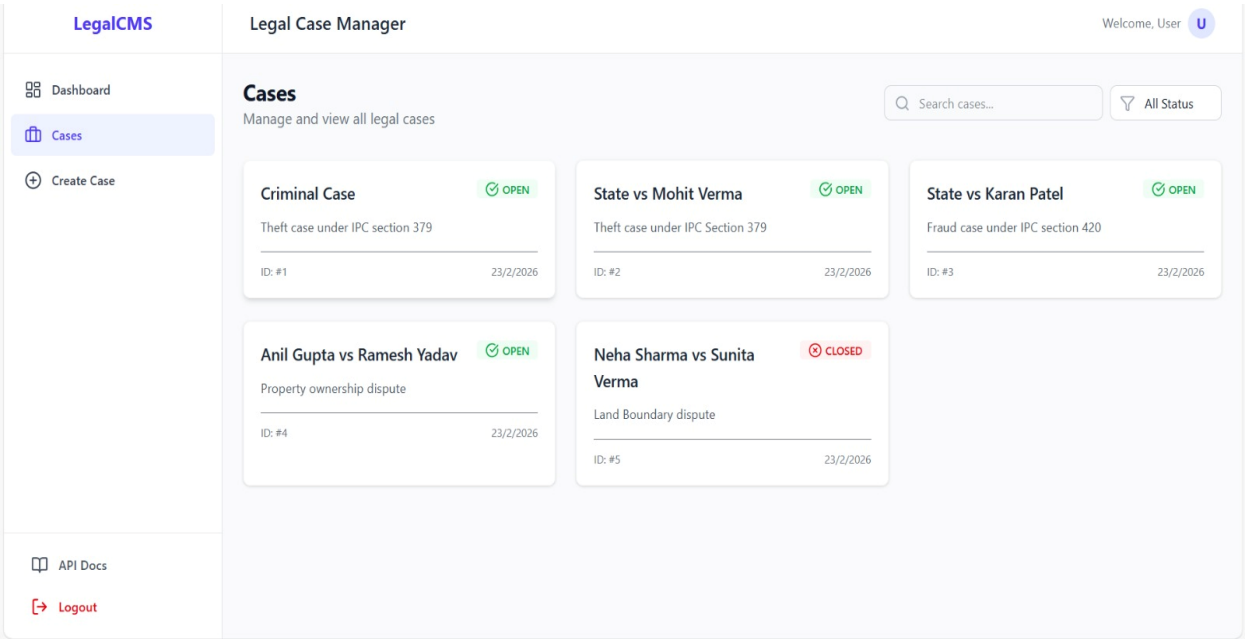
1 .Screenshot :login



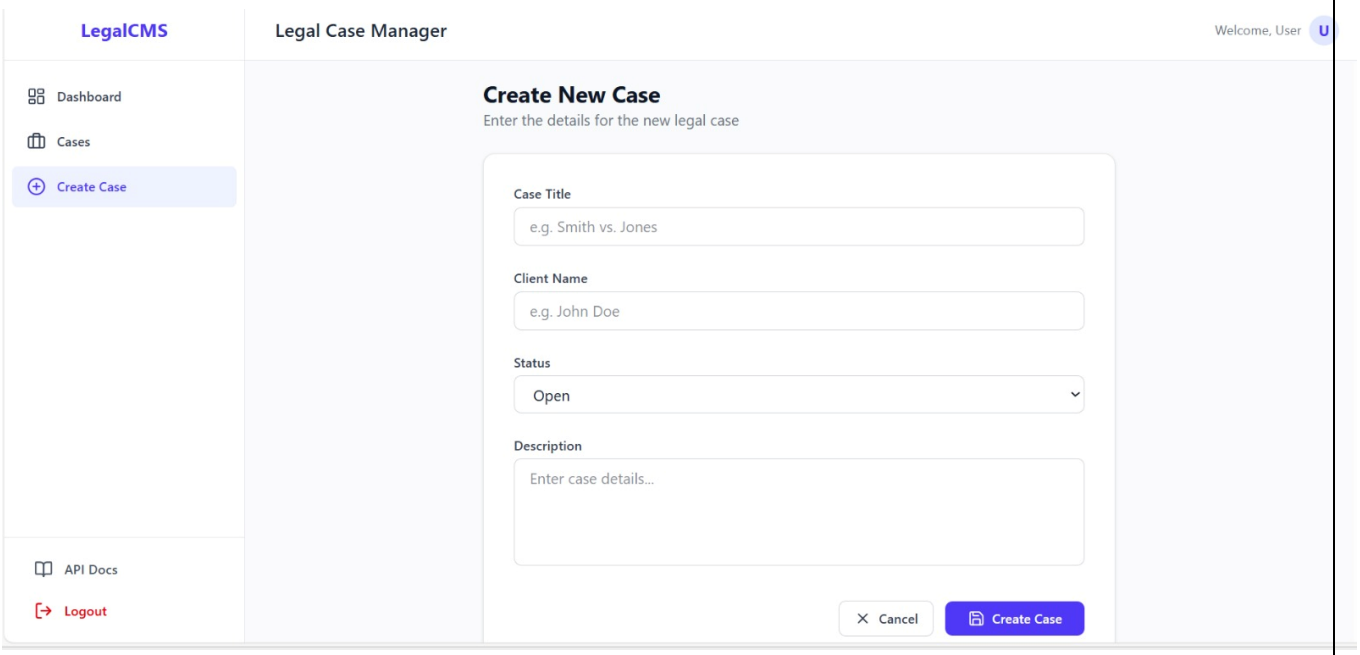
2. Screenshot:ApplicationN Dashboard



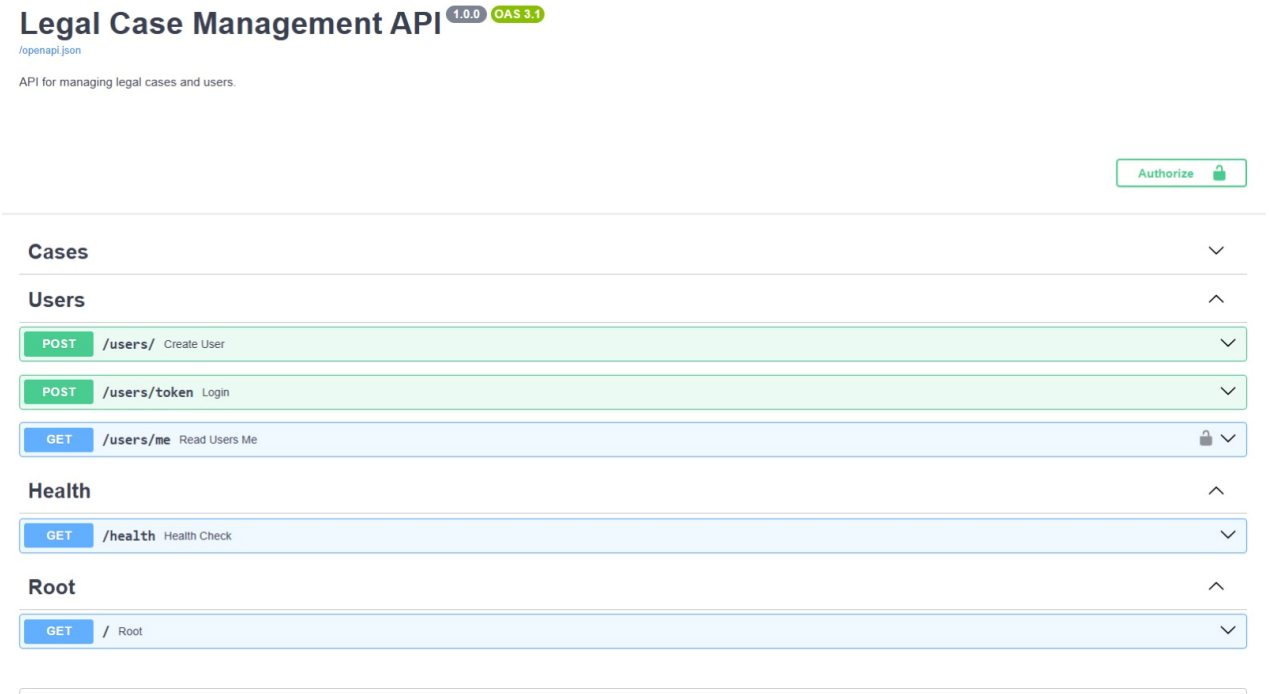
3.Screenshot : Case Entry Form



4.Screenshot: create new cases



5. Screenshot: API for legal case



3.2 Reports / Dashboards / Models

The system was evaluated based on functional correctness, deployment automation, and overall system performance. The following summarizes the key findings:

Evaluation Area	Outcome
Case Data Processing	All case records were stored, retrieved, and updated without data loss
Automated Testing	Unit and integration tests executed successfully on each code commit
CI/CD Deployment	Deployment was triggered automatically upon passing test verification
Docker Containerization	Application packaged and running as container with all dependencies
Kubernetes Orchestration	Pods deployed and managed successfully with rolling update strategy
System Performance	Response times were within acceptable limits under test load

3.3 Key Outcomes

The project achieved all primary objectives. The following outcomes were observed:

- Case management process was fully automated, eliminating manual record-keeping
- Manual errors were significantly reduced through data validation and structured input
- Faster deployment was achieved — code changes go live in minutes via CI/CD
- System reliability improved through containerized, stateless deployment
- DevOps practices were successfully integrated into the development lifecycle
- Rollback capability was established in case of deployment failures

4 .Conclusion

This project successfully demonstrated the development and deployment of a Legal Case Management System using modern DevOps principles. The system addresses the core inefficiencies found in traditional manual case management by providing a digital, automated, and structured platform for managing legal case data.

Through the implementation of a CI/CD pipeline using GitHub and Docker, the project achieved automatic testing and deployment of code changes. Kubernetes was used to manage the deployment lifecycle, ensuring the application remains scalable and available. The separation of concerns across Frontend, Backend, and Database layers provided a clean architecture that is easy to maintain and extend.

This project provided practical, hands-on experience in the following areas:

- Software development and architecture design
- Version control and collaborative development using GitHub
- Application containerization using Docker
- Container orchestration and deployment using Kubernetes
- Building and configuring CI/CD pipelines for automated deployment
- Testing automation and quality assurance practices

Overall, the project is a successful implementation of a DevOps-integrated software system that can serve as a foundation for more advanced legal case management solutions in the future.

5 . Future Scope & Enhancements

While the current system fulfills its intended objectives, several enhancements are identified for future development iterations. These improvements would transform the system into a comprehensive, enterprise-grade legal management platform.

Enhancement	Description
Cloud Deployment	Deploy the system on AWS, Azure, or GCP for global accessibility, automatic scaling, and improved disaster recovery capabilities
Advanced Security	Implement role-based access control (RBAC), OAuth 2.0 authentication, end-to-end encryption, and audit logging
Notification System	Add email and SMS alerts for case updates, hearing reminders, and deadline notifications to relevant stakeholders
Data Analytics	Integrate a reporting and analytics dashboard to track case trends, resolution times, and performance metrics
Mobile Application	Develop a cross-platform mobile app (iOS/Android) using React Native or Flutter for on-the-go case management
System Scalability	Optimize database indexing, introduce caching layers (Redis), and implement load balancing for high-volume usage
AI Integration	Incorporate AI/ML models for predictive case outcome analysis, document classification, and intelligent search
Document Management	Add support for uploading, storing, and managing legal documents such as FIRs, affidavits, and court orders

These enhancements would position the Legal Case Management System as a robust, scalable, and intelligent solution capable of supporting the operational needs of law firms, public prosecutors, and judicial institutions at scale.

—