BITS Pilani
Pilani | Dubai | Goa | Hyderabad

SE ZG503 FULL STACK APPLICATION DEVELOPMENT
ASSIGNMENT- WEB APPLICATION DEVELOPMENT

Submission Date: 04 May 2025

Assignment Submission Mode: ELearn LMS- File Upload

Weightage: 20 % As per course handout

OBJECTIVE:

Develop a full-stack web application for the given problem statements that follows modern application development principles required to design and implement a robust system.

The assignment expects you to develop a full-stack web application for the problem statement given below.

This is an individual take-home assignment to be carried out by each learner independently.

PROBLEM STATEMENT

TITLE: SCHOOL VACCINATION PORTAL

Problem Statement:

You are tasked with building a full-stack web application to manage and track vaccination drives in a school.

A web-based vaccine tracking system to be specifically designed for school vaccination administration, tracking, and management. The system will be used by school coordinators to manage student records, schedule vaccination drives, update vaccination statuses, and generate reports. This system should include interaction between the frontend (React-based UI) and a backend service (Node.js/Express, Python/Flask, or any framework of your choice) with appropriate data modeling and persistence (e.g., MongoDB/PostgreSQL/SQLite).

You will be responsible for implementing:

- Backend APIs and models to manage data
- Frontend interfaces to interact with the system
- Business logic validations at the backend
- Basic file handling and basic authentication

FEATURES:

This system is intended to be a **comprehensive internal tool** for schools to manage student vaccination efforts. It focuses on real-world workflows, encourages data integrity, and involves designing core backend systems and frontend interfaces.

Below are the key features

1. AUTHENTICATION & ACCESS CONTROL (SIMULATED)

The application will be used by a school coordinator (admin role).

Login flow can be hardcorded or token-based login can be used; no need to implement a real authentication system.

After login, the user should be directed to a personalized dashboard with actionable data.

2. DASHBOARD OVERVIEW

Display important metrics and insights, such as the total number of students, the number and percentage of students vaccinated, Upcoming vaccination drives (only those within the next 30 days)

Provide quick navigation links to manage students, vaccination drives, and reports.

Also, it should handle empty states (e.g., no drives scheduled) gracefully.

Backend Requirements: Create an Aggregate data APIs (count, percentage, upcoming drives) with efficient querying and conditional logic

3. STUDENT MANAGEMENT

Add/edit student details individually via a form. Also, bulk import of students is done through a CSV upload. View/search students by name, class, ID, or vaccination status.

Mark students as vaccinated within a drive context. One student should not be vaccinated twice for the same vaccine.

Backend Requirements: Create CRUD APIs for student data. APIs to fetch and update vaccination records for students

4. VACCINATION DRIVE MANAGEMENT

Create and manage drives, including Vaccine name, Date of the drive, Number of available doses, and applicable classes (e.g., Grades 5–7). Prevent overlaps and enforce that drives are scheduled at least 15 days in advance. Edit drives before their scheduled date; disable editing for past drives.

Backend Requirements: Drive scheduling logic with date validation, Update/disable logic for expired drives, Prevent scheduling conflicts

User Story 1.1: As a school coordinator, I should be presented with a landing page on successful login.

Description: The school coordinator page should contain a dashboard view and a navigation menu to Add/Manage student details, Manage/update Vaccination status, generate reports. The Dashboard to contain the status of analytics (number of students vaccinated, upcoming vaccination drives etc.

Acceptance Criteria:

If there are no vaccination drives, it should display a message “no upcoming drives”.

The status of number of students registered vs number of them vaccinated to be displayed.

User Story 1.2: As a school coordinator user, I should be able to select the menu items.

Description: On clicking the menu item “Add/Manage student details”, the user should be navigated to the student details page.

Acceptance Criteria:

A form to accept individual student details to be provided

Bulk upload of student details option to be provided.

User Story 1.3: As a school coordinator user, I should be able to generate the report of the vaccinations administered with student details.

Description: On clicking the “generate report” and selecting appropriate filters the user should be able to view the list of students with vaccination details like vaccinated status, date of vaccination, name of vaccine. Filtering options with name of vaccination to be present

Acceptance Criteria:

The student details with vaccination details like vaccinated status, date of vaccination, name of vaccine to be displayed in a table.

Pagination to be applied.

Option to download the details in a CSV/ excel/ pdf to be provided.

User Story 1.4: As a school coordinator, I should be able to book a vaccination drive

Description: The school coordinator should be able to enter the details of the vaccination drive and create it after approval from the relevant authorities

Acceptance Criteria:

A form to enter the details of the vaccination drive [date, number of vaccines available]

On submit the details to be updated in the upcoming vaccination drives.

User story 1.5: As a school coordinator, I should be able to view the details of the upcoming vaccination drive and modify it

Description: The school coordinator should be able to view/edit the details of the vaccination drive

Acceptance Criteria:

The school coordinator should view the details of vaccination drive.
Option to edit the dates of the drive, number of slots/vaccines to be available.
The edit option to be disabled for completed vaccination drives

ASSIGNMENT EXPECTATIONS

PART 1 BACKEND

You are expected to implement the API s required for the problem statement. Your implementation should follow modern web application development practices.

Use Node.js (or another backend framework of your choice) to handle server-side logic. Develop REST APIs or GraphQL APIs to handle CRUD operations with a database. Ensure proper routing, URL mapping, and response handling. Use any database (SQL/NoSQL) for data storage and retrieval.

PART 2. FRONTEND

Design and implement the frontend UI for the problem statement. Use a client-side JavaScript framework (React) to build the user interface. Design an interactive, user-friendly UI with responsiveness and state management. Use modular components and route-based structure in the frontend

PART 3: INTEGRATION

A complete end-to-end demonstration of the user stories implemented. Integrate the frontend and backend components and demonstrate the capabilities. Demonstrate communication between the front end and back end.

DELIVERABLES:

1. **Documentation**: System overview, Full application architecture, frontend-backend interaction, API endpoints, API documentation (can be via Swagger/Postman or markdown), DB schema or model diagrams, Brief documentation on any assumptions, UI/UX wireframes if any. Provide clear instructions to run both frontend and backend locally. Also include output snapshots of UI, snapshots of API response in tools like Postman.
2. **Demonstration Video of the Working Prototype**: Video recording showing the demonstrations of the user stories implemented both frontend and backend.
   Note: if you are exceeding size limits when uploading videos to the elearn portal, add it to a Google Drive and submit the link)
3. **GitHub Repository:** A well-organized repository with a clear README file that includes instructions for setting up the application.

SUBMISSION INSTRUCTIONS

1. Upload the complete code to the GitHub repositories. GitHub repository access is to be kept public and open for all evaluators.
2. Create a ZIP file including all the deliverables [(Design document, Demonstration Video, Github Link (mentioned in a notepad)]and upload a single ZIP file to the LMS.
3. **Academic Honesty:** You can discuss with peers and refer to the internet to understand the concept better. However, you may not share code with other groups or do not do a verbatim copy from the Internet/Generative AI responses/GitHub repositories. Include appropriate references. The code will be tested for plagiarism. If found guilty, no marks will be awarded.

WEIGHTAGE AND EVALUATION:

Total: 20 Marks

Activity	Marks
Backend Design and Implementation	7
Frontend Design and implementation	7
Integration	6

The relevant documents to be submitted only in the elearn portal. After the completion of deadline, Assignment will be evaluated by a group of instructors on the submissions made.

Evaluation Criteria:

- Implementation of User Stories: Functionality, accuracy, and user experience of the implemented stories.
- Code Quality: Readability, adherence to best practices, and documentation.
- Code originality

NOTE:

- This is a take-home assignment to be carried out by each learner independently
- In case of any further queries, use discussion forums, or reach out to me at akshaya.ganesan@pilani.bits-pilani.ac.in.