NAME	DINESH KARTHICK D
ROLL.NO	7376221EE111
SEAT.NO	387
PROJECT ID	27
PROBLEM STATMENT	FITNESS CERTIFICATE PORTAL

Full Stack - Python Stack (AI)

Front End	• HTML • CSS • JS
Back End	• Python • Django(Python Web)
Database	• PostgreSQL • MySQL
API	• OpenAPI • SOAP APIs • REST Ful API

PROBLEM STATEMENT:

Develop a portal indicating the summary of Fitness Certificate categories. Number of NO's in the corresponding FC must be displayed. Duration of FC submission is bimonthly. number of 3 consecutive NO's in the subcategory of each FC needs to be displayed and the corresponding list needs to be popped up.

PROJECT FLOW:

1.PURPOSE:

This portal aims to provide a comprehensive overview of Fitness Certificate categories, highlighting areas of concern with the number of NO's. By offering timely insights and alerts for three consecutive NO's, it facilitates efficient management of compliance and safety standards in fitness assessments, promoting a proactive approach to risk mitigation and regulatory adherence

2. System Overview:

This document outlines the design of the Fitness Certificate Portal, a web-based application for managing Fitness Certificate (FC) evaluations conducted by admin.

System Objectives:

The primary objective is to develop a user-friendly Fitness Certificate Portal that addresses the limitations of manual processes. This application will offer the following functionalities:

- **Data Management:** Securely store and manage FC evaluation data collected from various departments.
- **Data Visualization:** Present a clear and concise summary table highlighting key FC evaluation metrics.
- **Alerts and Notifications:** Identify categories or subcategories with concerning trends (e.g., consecutive 'NO' responses).
- **Reporting and Analytics:** Generate reports and charts for deeper insights into FC compliance trends.

System Architecture:

The system will adopt a three-tier architecture for modularity and maintainability.

- **Frontend (HTML, CSS, JavaScript)**: This layer handles user interactions and visual presentation of data. It will be responsible for building the user interface using HTML, styling it with CSS, and implementing interactivity using JavaScript.
- **Backend (Python ,Django)**: This layer serves as the application's core, handling business logic and data persistence. Django, a high-level Python web framework, will streamline backend development tasks.
- **Database (MySQL,PostgreSQL):** The system will leverage MySQL, a popular relational database management system, to store FC evaluation data in a structured manner.

System Requirements:

Functional Requirements :

- **Data Upload:** Secure mechanism for authorized users to upload bimonthly FC evaluation data through Portal.
- **Data Processing:** The Django backend will process the uploaded data, calculate totals, identify consecutive 'NO' responses, and generate reports (if applicable).
- **Data Visualization:**
    - Display a summary table with FC categories, total 'NO's, and links for concerning trends (potentially implemented using JavaScript).
    - Pop-up window showcasing detailed information on subcategories with consecutive 'NO's (developed using HTML, CSS, and JavaScript).
- **Reporting and Analytics:** Generate customizable reports and charts on FC compliance trends over time (potentially using Django libraries or third-party tools).

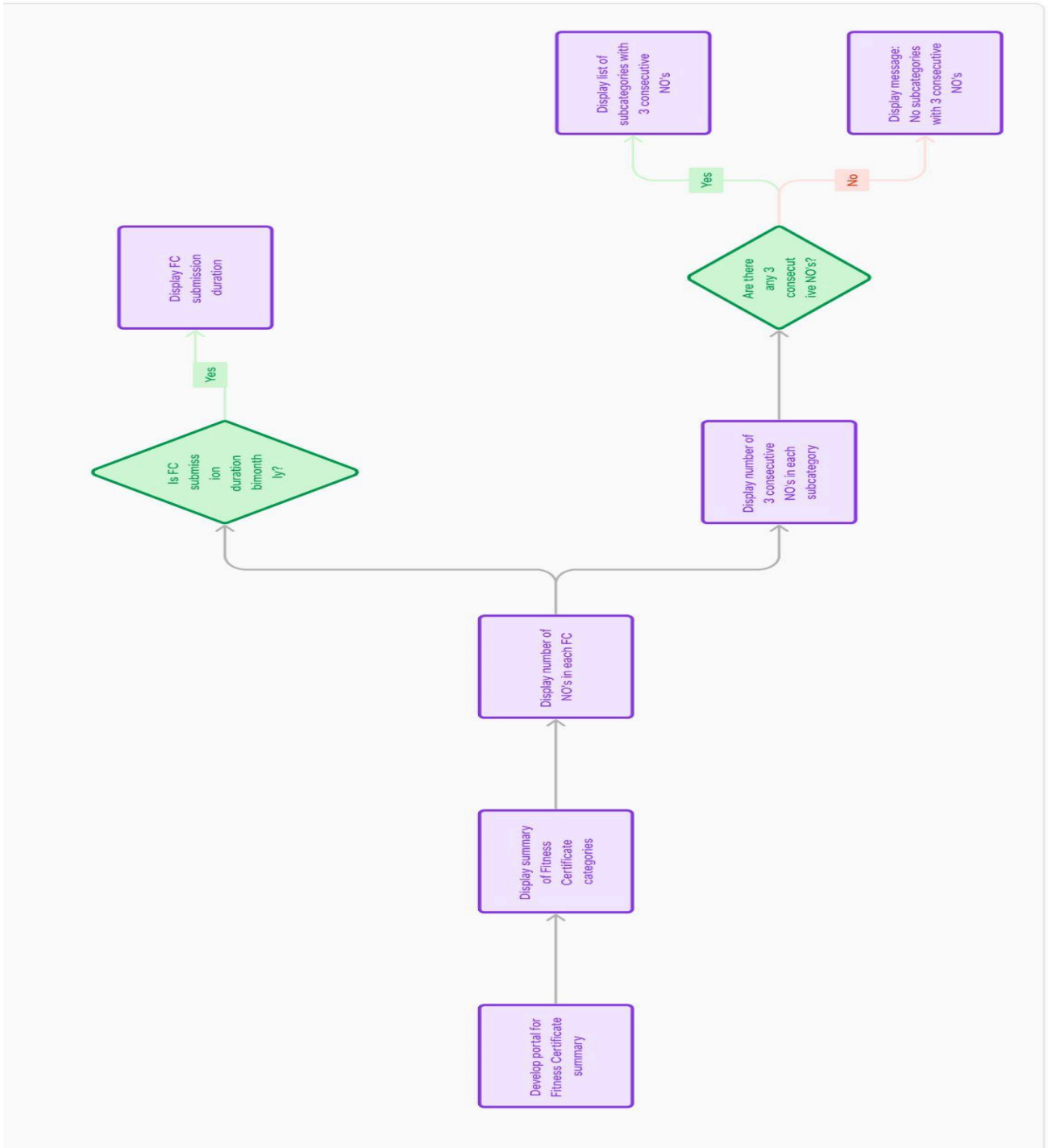
3. Data Flow:

- **Data Input:** Authorized users will interact with a web form built using HTML and JavaScript to upload bimonthly FC evaluation data.
- **Data Processing:** The Django backend will receive the uploaded data, process it using Python code, and store relevant information in the MySQL database.
- **Data Storage:** FC evaluation data will be stored in MySQL tables with appropriate schema definitions.
- **Data Retrieval:** Users can access the summary table and generate reports based on their needs. When users click on specific categories, Django will retrieve pop-up details from the database and send them to the frontend for presentation.
- **Data Output:** The processed data will be presented visually through the HTML table, pop-up details, and reports.

4.Technologies:

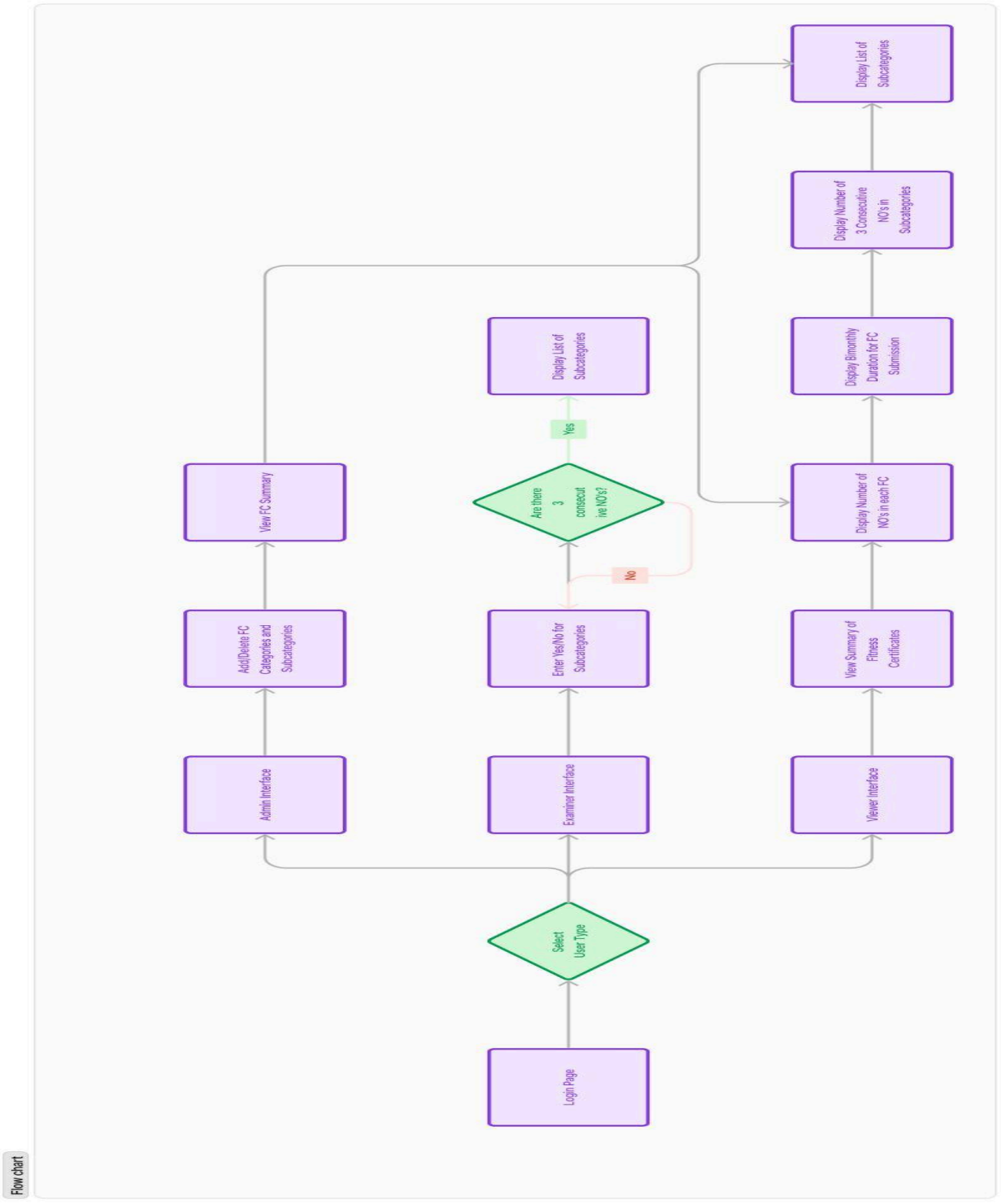
- **Frontend**: HTML, CSS, JavaScript.
- **Backend**: Python, PythonDjango.
- **Database**: MySQL,PostgreSQL.

WORK FLOW :



https://drive.google.com/file/d/1c-2k7_O19hOD2IVkWRvQCgjxQmZjl7_M/view?usp=sharing

USER INTERFACE :



https://drive.google.com/file/d/1qUs3pCP_JhKpS-H4uNVJuOoO7ycrRkdq/view?usp=sharing_