**Project Spec**

**Description:**

This project is to build a **personal portfolio website** using the Flask framework. The website will showcase the admin's professional projects, personal information, and an image gallery. The website will have two primary types of users:

1. Admin: Will have the ability to add, edit, or delete content such as projects, blog posts, and images directly on the website.
2. Viewers: Regular visitors who can view the portfolio content but cannot make changes.

Key Features:

* Admin Control Panel: Admin can log in to manage the content.
* Content Management: Admin can add or remove projects, descriptions, and pictures.
* Viewer Interface: Users can view the content, but their interaction will be limited to reading and viewing images.
* Flask for Backend: Flask will serve as the backend framework, handling routing, rendering templates, and managing the content.
* Database: SQLite (or another simple relational database) will store user credentials, projects, and images.
* Front-End: HTML, CSS, and possibly JavaScript for interactive elements (consider using Bootstrap for a responsive design).

External Mechanisms:

* Flask for routing and server-side logic.

GUI:

* Admin Panel: Simple web-based GUI to manage content.
* Viewer Interface: Public-facing website where visitors can explore the portfolio.

**Task Vignettes:**

Task 1: Admin Login

Vignette:  
The admin navigates to the login page, enters their username and password, and logs into the dashboard. From the dashboard, the admin can see a list of all existing projects and images, with options to add, edit, or delete content.

Technical Details:

* User credentials are stored in the database.
* Admin login is handled by Flask-Login.
* Password encryption

**Task 2: Adding a New Project**

Vignette:  
From the dashboard, the admin should be able to add projects. A form is displayed, asking for project details (title, description, image). The admin fills in the form and submits it, and the new project appears on the website's "Projects" page.

Technical Details:

* Form submission with Flask-WTF.
* File upload handling for images.
* Database entries updated with project details (title, description, image).

**Task 3: Viewing Projects (Viewer Flow)**

Vignette:  
A viewer visits the portfolio website. They navigate to the "Projects" section and can see a grid of projects with titles and descriptions. Clicking on a project shows a detailed view of the project, including more information and associated images.

Technical Details:

* Each project has a unique URL endpoint (e.g., /project/<id>).
* Pagination for larger portfolios.

**Technical Flow:**

Data Flow:

Inputs are given at several stages in the user flow such as admin inputting their personal details, photos, projects.

Flask will interact with database for CRUD operations.

Viewer interface will get the data from the database and display it on the front end/GUI.

Blocks:

Content Management:

This block will handle functions like adding, updating or deleting the content from the database.

Frontend:

This block will have HTML templates for the front-end.

Data Types:

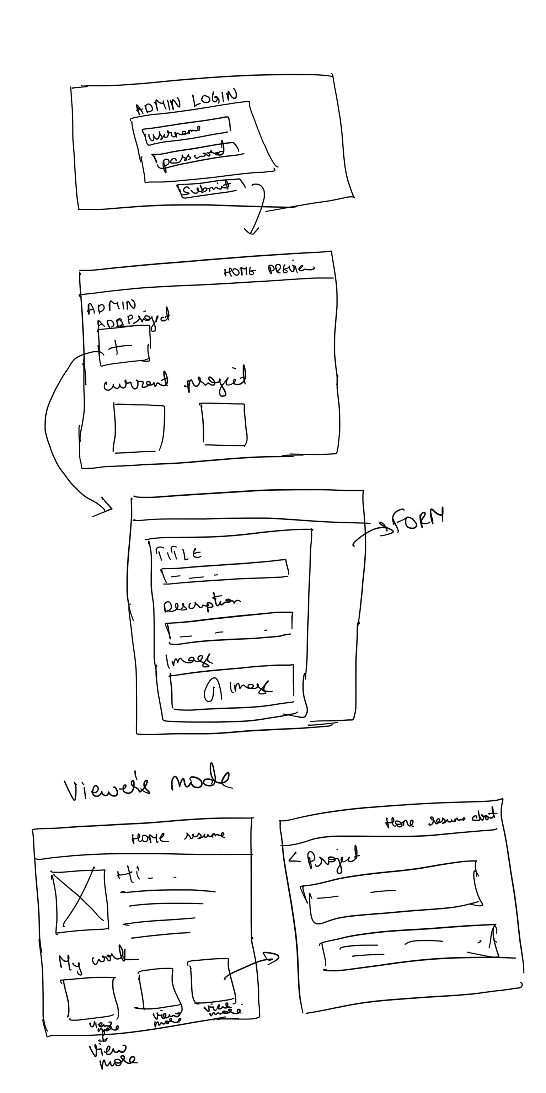
Each Projects can be stored as JPEG, strings.

All Projects together can be stored as a list of list (List of JPEGS, strings)

User Interaction:

Admin: Admin panel with input forms for adding and deleting content.

Viewer: Simple navigation to view projects and images.



**Self-Assessment:**

The original sketch only described the portfolio itself. In the final spec, I have added an admin interface and explained about the content management system.

I am moderately confident to build the GUI that I have written as I can only code moderately.

I might need help in managing/setting the database and need suggestions to choose which database to use.

I am familiar with SQL and read about SQLite and MangoDB. Please suggest me on what database to use.

I might need help on admin login section.

**Link to the git repository:** <https://github.com/deekshareddy57/PortfolioWebsite_HCI584/tree/main>