**Python mini project report**

**Title-Certificate Butler**

Mark Lopes-9913

Vivian Ludrick-9914

Introduction:

The project aims to develop a certificate generation system utilizing Vite + Svelte for frontend and Django with SQLite for backend. It begins with requirement gathering and technology stack selection, followed by frontend and backend development phases covering UI design, API implementation, data processing, image generation, zip file creation, and frontend-backend integration. The system will streamline certificate management for competitions, including features for form submission, history access, and efficient response handling.

Methodology:

1. Project Planning:

* Requirement Gathering: Understand the user needs and define the features required for the application.
* Technology Stack Selection: Choose Vite + Svelte for frontend development and Django with SQLite for backend development based on project requirements and familiarity.

1. Frontend Development:

* Environment Setup: Set up the development environment for Vite + Svelte.
* UI/UX Design: Design the user interface, including the homepage, form for certificate generation, and history tab.
* Form Implementation: Create the form to capture user inputs for certificate generation.

1. Backend Development:

* Environment Setup: Set up Django and SQLite for backend development.
* Database Design: Define the database schema with tables: Certificate\_templates, Participants, User, and Zipped\_file.
* API Development: Implement APIs to handle incoming data from the frontend, including storing certificate templates, participant details, and generating zip files.
* Integration with Pillow: Integrate the Pillow library for image generation and manipulation.

1. Data Processing:

* CSV Parsing: Develop functionality to parse the uploaded CSV file to extract participant details.
* Code Generation: Implement logic to generate unique codes for certificate templates and associate participants with the generated template code.

1. Image Generation:

* Certificate Rendering: Use Pillow library to render participant details on the certificate templates.
* Image Storage: Store the generated images in the storage and create public links for access.

1. Zip File Creation:

* Certificate Compilation: Compile individual certificate images into a zip file for each competition.
* Storage and Linking: Store the generated zip files and link them to the corresponding competition IDs.

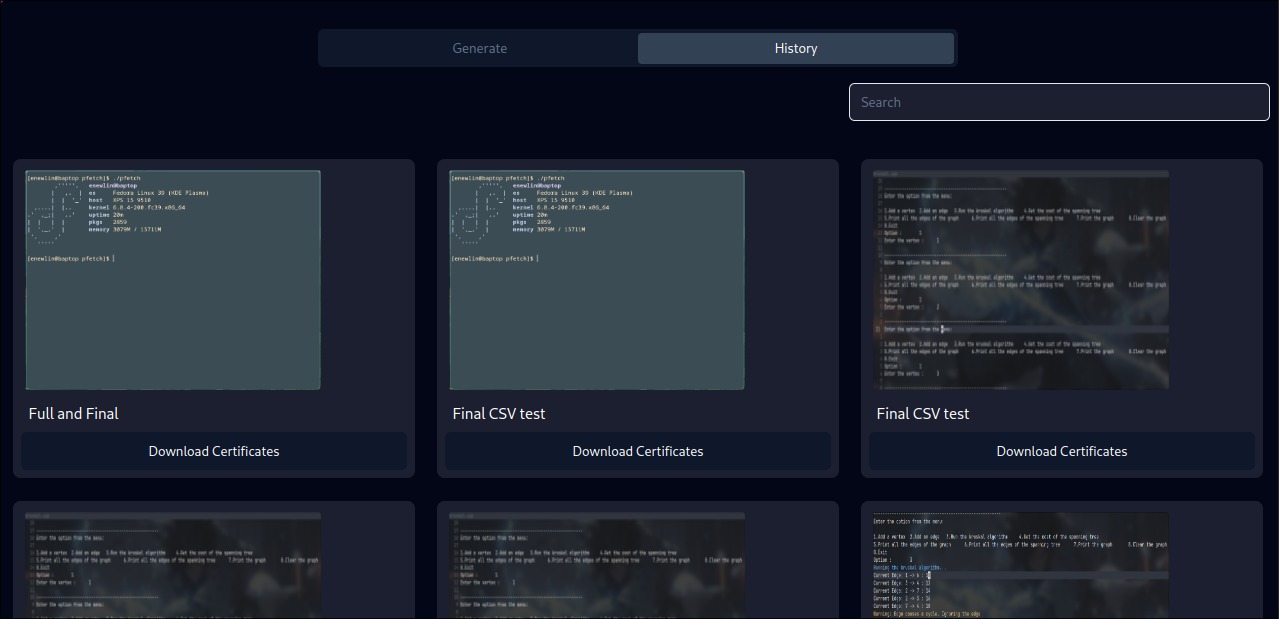
1. Frontend-Backend Integration:

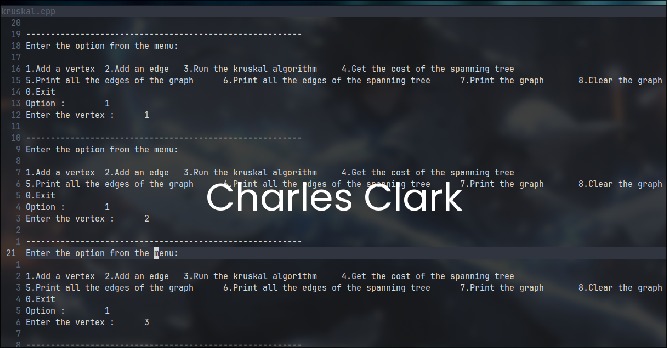
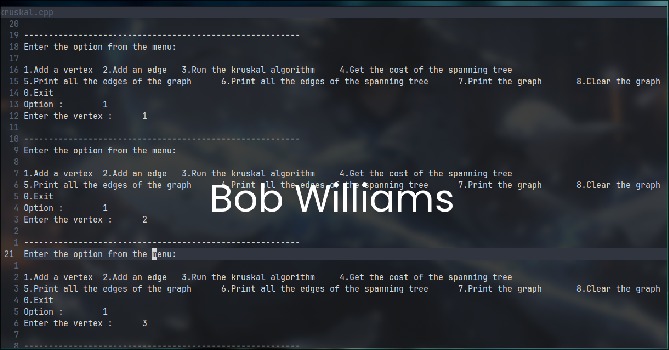
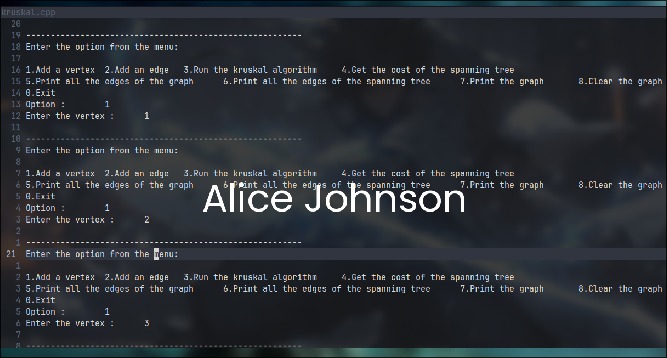
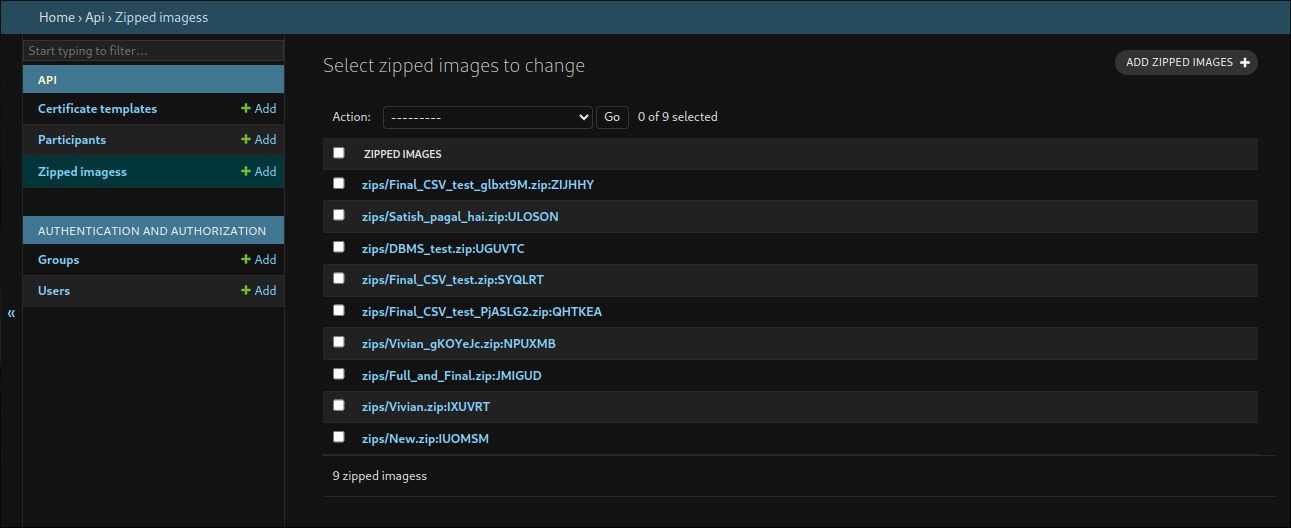
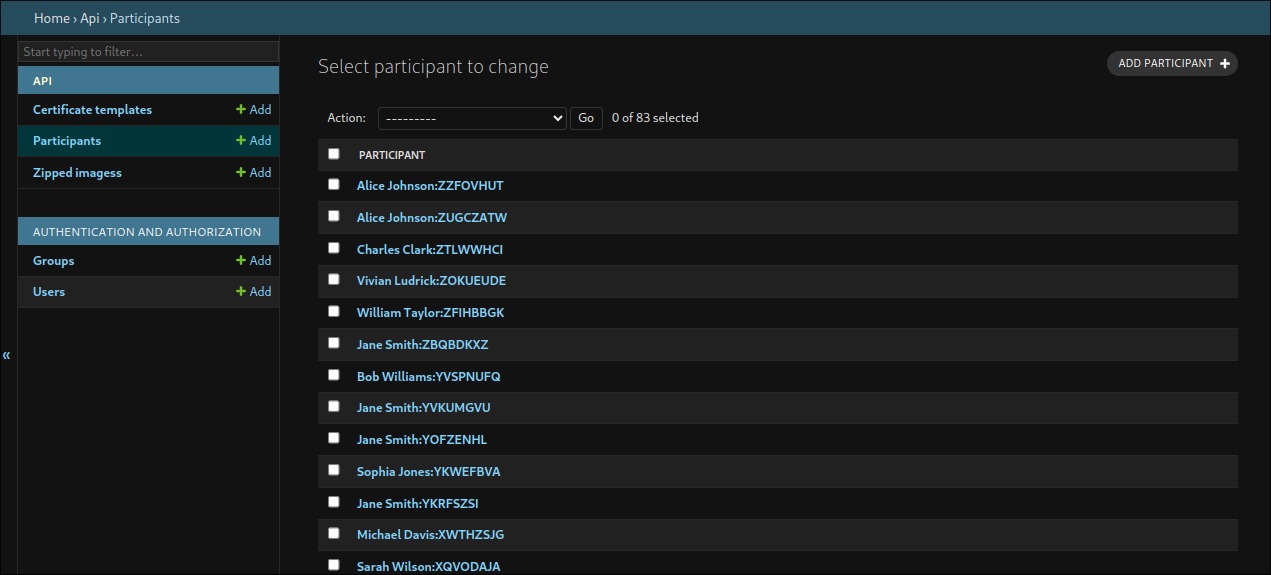
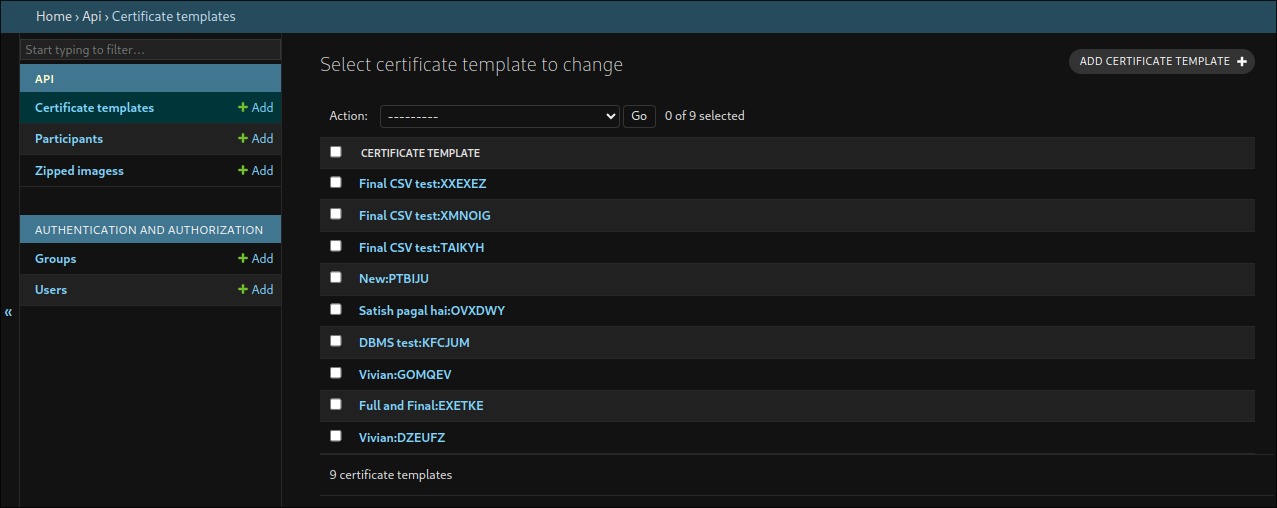
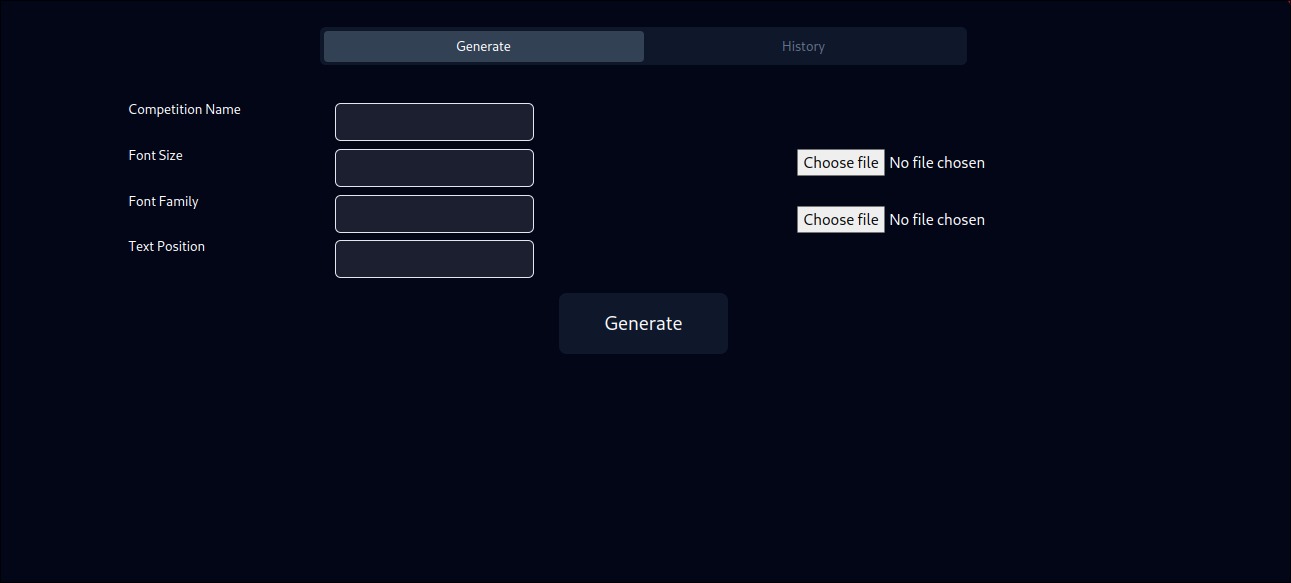
* API Integration: Connect the frontend with the backend through APIs to facilitate data exchange.
* Response Handling: Implement logic to handle responses from the backend, particularly the public links for downloading zip files.

1. History Access:

* Implement a history tab on the homepage.Allow users to view and download previously generated certificates for past competitions.

Result:





Conclusion:

﻿In conclusion, the certificates generation system developed offers a robust solution for successfully managing certificate for numerous competitions or occasions. By leveraging a cautiously selected era stack and following a structured development method, the device gives a person-pleasant interface, seamless data exchange between frontend and backend, and complete functionalities which includes shape submission, records get entry to, and zip report creation. With its potential to streamline certificates control processes, the project contributes to enhancing efficiency and person enjoy in managing certificate for numerous occasions.