**Software Implementation and Testing Document**

**For**

**Group <33>**

Version 1.0

**Authors**:

Manar Mohsin

Ayah Halloum

Felipe Ferreyra-Lima

Katherine Lu

Jonathan Alcineus

# Programming Languages (5 points)

The languages used for this project are Python and TypeScript. Python was used for the entire of the backend, such as managing data, implementing logic, and managing the server. Typescript was used for the frontend, such as handling user interactions and managing application state, creating registering and login components. The reason for choosing python is because of its vast libraries, and simplified code syntax.

# Platforms, APIs, Databases, and other technologies used (5 points)

We did not use any Platforms like cloud services or web servers. The backend API was developed using the Django REST framework, and the API testing tool used was postman. The database we use was SQLite3. On the frontend Next.js was used for rendering server-side and routing.

# Execution-based Functional Testing (10 points)

For the backend, we performed manual function testing using Postman to validate the API endpoints and business logic. We used a postman to send various requests (GET, POST, PUT, DELETE) to the API endpoints, ensuring that they returned the expected JSON responses. We tested the authentication and authorization logic by attempting to access protected endpoints with and without valid tokens. On the frontend we performed basic observational testing. This includes verifying the navigation flow was functional and basic user interactions were responsive.

# Execution-based Non-Functional Testing (10 points)

The response time for the backend was within acceptable limits for the development environment. We tested the backend multiple times by sending requests and they were successful. We also monitored the response times of the database queries, and ensured that they were performing as expected. Security testing was done by testing for proper authentication and authorization by attempting to access protected endpoints with invalid or missing tokens. Due to time limitations we did not conduct all testing required, we will do some in the future. For the frontend the website was responsive. We tested navigation flow, form submissions, and error handling. We simulated user interaction and evaluated the application's ease of use.

# Non-Execution-based Testing (10 points)

We have not conducted any code reviews throughout our code. However we did have informal discussions about the code during the development. While not a structured approach, these conversations helped us improve the overall understanding of the codebase. Due to the time constraints and the project scope, we were unable to implement a more formal non-execution testing process, but we will in the future when everyone is on page and have a good understanding of the overall structure of the project.