Keanu Muller

ST10306640

SOEN6222

POE Part 2

**Question 1:**

**Selected architecture -**

**Model-View-Controller (MVC) Architecture Pattern**

The MVC architecture is a commonly used software design pattern that splits an application into 3 main components - The Model, the View and the Controller. Modularisation is ensured through these separate parts, allowing for individual developing, testing and maintenance of each component, which is useful in relation to the agile development approach we are required to use.

1. **Model**:

This represents the logic and data of the application. It manages the information related to business logic, including tree-planting initiatives, user accounts and donations. In my solution, this will deal with tasks such as events, donation tracking and volunteer processing. Any data changes are reflected in the Model, and these updates are seen in the View.

1. **View**:

This represents the data to the user visually. This includes the formatting of the UI for tasks like event sign ups or making donations. The View gets info from the Controller, and creates an easy to understand format for users, which in this case, may be a web interface allowing users to view project updates, upcoming events and progress on donations.

1. **Controller**:

This is the middle man between the Model and the View. It gets user input, processes it, and handles the Model for data updating. After updating, this interacts with the View to show the now updated data. If, for example, a user donates, the Controller updates donation info found in the Model, and lets the View display a confirmation message.

**Why Choose MVC?**

The MVC architecture is ideal for this project for several reasons:

* **Separation of Concerns**:

Having the Model, View and Controller be distinct parts allows for independent development and maintenance. This will be crucial in this assignment, as there will be separated functionality like donation tracking, volunteer management and progress monitoring for initiatives, and these can be built separately without affecting the rest of the system.

* **Scalability**:

As the organisation becomes bigger, so will the user count, donations and initiatives. MVC being modular allows for independent scalability through the addition of Models or Views without having existing ones be affected. If, for example, the organisation at a later stage chooses to add additional types of projects, the Model can easily be upgraded without causing disruption to the system.

* **Testability**:

MVC allows for easier and independent testing of all components, allowing developers to ensure parts of the system work before integrating it to the system as a whole. Given that progress tracking and donation management functionalities are crucial for this system, the use of MVC enables these components to work correctly before being implemented.

* **Support for Agile Development**:

MVC easily aligns with agile principles, enabling iterative development and regular feedback from stakeholders. The Views can be modified easily based on provided feedback, and the Model to add new business requirements, all without an entire system rework.

In conclusion, MVC creates a scalable, flexible and maintainable architecture, aligning with the non-profits software needs. The ability to distinguish between the data, UI and business logic makes it a solid choice for handling the different aspects of the organisation's operations.

**Question 2:**

Add diagram here from pc!