10/25/2024

SOEN 6222

DESIGN DOCUMENT

Sajana Bidesi
ST 10249843

Table of Contents

ARCHITECTURE PATTERN 2

1. What is an Architecture Pattern? 2
2. How Does an Architecture Pattern Work? 2
3. Why and How Would MVC Be Used for Helping Hands? 3

UML DIAGRAM 4

FUNCTIONAL (SOFTWARE) REQUIREMENTS 5

REFERENCES 6

ARCHITECTURE PATTERN

1. What is an Architecture Pattern?

An architecture pattern is a widely recognized solution for addressing common software development challenges while promoting separation of concerns. It provides a blueprint for organizing code, defining component relationships, and managing data flow within an application (Interserver, 2016). These language-agnostic patterns can be applied across various programming environments, allowing developers to create well-structured, testable, and maintainable code, thereby streamlining the development process. (Sheldon, 2023)

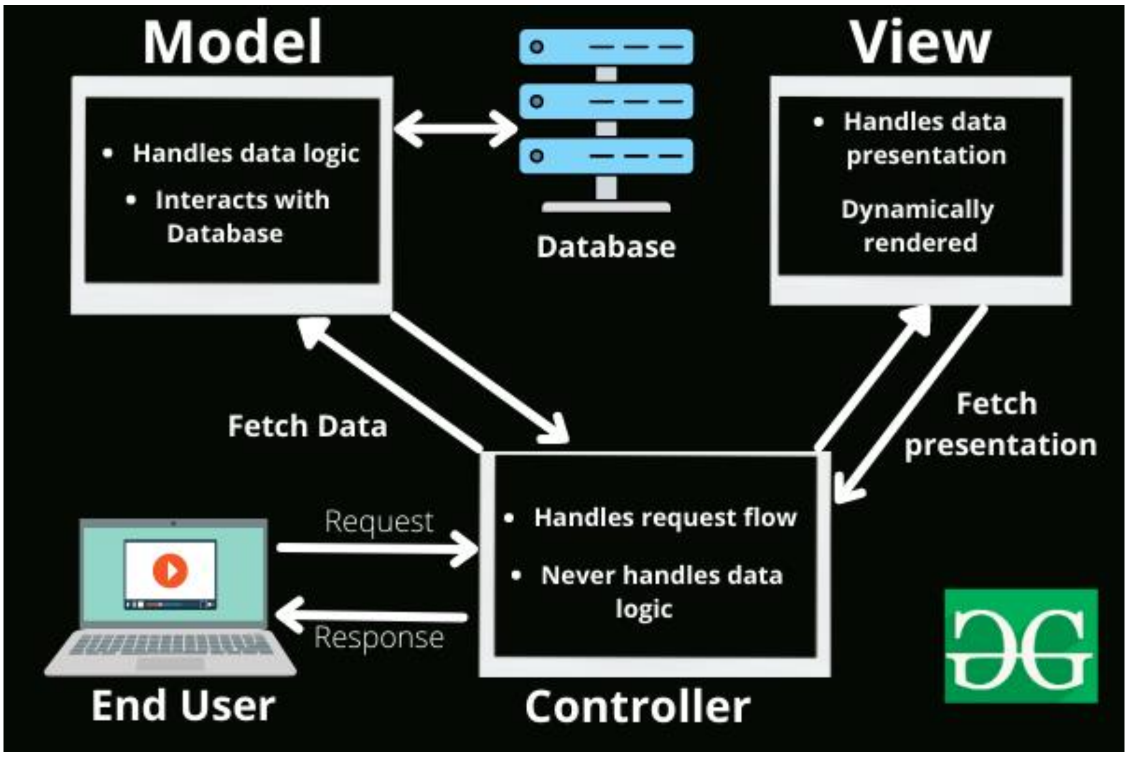
In this project, I have chosen the MVC (Model-View-Controller) pattern to leverage these benefits. The MVC approach clearly separates data, user interface, and logic, facilitating modular development, easier testing, and improved scalability.

2. How Does an Architecture Pattern Work?

Architecture patterns segment an application into distinct components, each with a specific role. In the MVC pattern:

- Model: Manages data logic, including storage, retrieval, and business rules.
- View: Serves as the user interface, displaying data and capturing user interactions.
- Controller: Acts as an intermediary, processing user inputs from the View, updating the Model, and directing the data presentation back to the View.

These structured interactions create a clear division of responsibilities, enhancing modularity and allowing for independent development, code reuse, and simplified testing and debugging. (Sheldon, 2023)



https://i.sstatic.net/hchsD.png

3. Why and How Would MVC Be Used for Helping Hands?

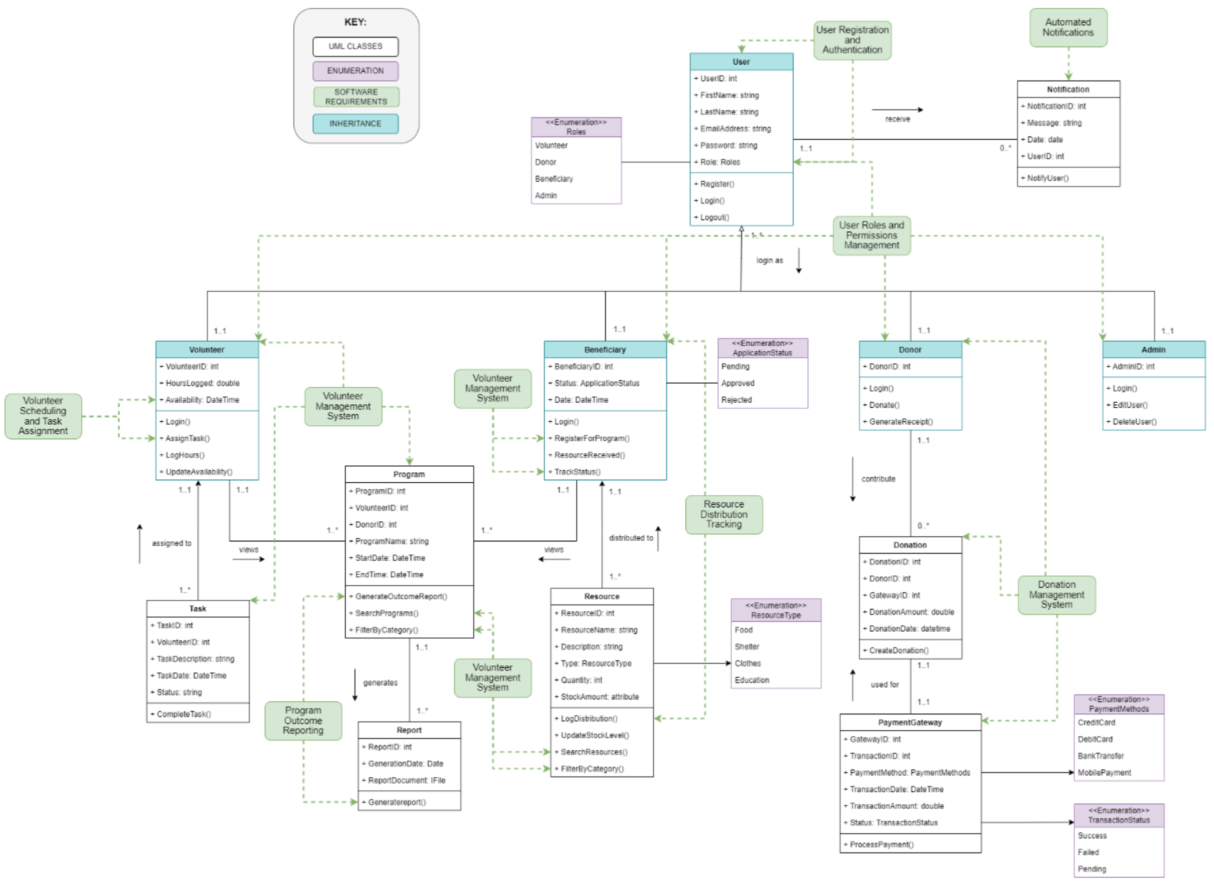
Implementing MVC for the Helping Hands project enhances organization and scalability in managing volunteers, donations, resources, tasks and beneficiaries. The Model Layer manages data logic and business rules for entities such as volunteers, donations, and beneficiaries, etc., often interfacing with a relational database for structured storage. The View Layer provides specific user interfaces for different roles, such as volunteers, donors, beneficiaries, and admin, allowing for personalized experiences without altering core logic. Meanwhile, the Controller Layer processes user inputs, interacts with the Model for data storage and retrieval, and manages workflows, including notifying users or assigning tasks based on availability. (Interserver, 2016)

Adopting the MVC architecture offers several benefits. It enables faster development through a clear separation of concerns (Interserver, 2016), allowing for more efficient collaboration among multiple developers working on different components. The structure also facilitates easier updates, as changes can be implemented with minimal disruption to the overall system. Additionally, the modular nature of MVC simplifies debugging, making it easier to isolate and fix issues.

However, there are some challenges associated with MVC. The architecture can be complex and may pose difficulties for new developers to understand. Additionally, adherence to strict rules and conventions can introduce a learning curve. Despite these minor drawbacks, the advantages of MVC significantly outweigh the disadvantages (Interserver, 2016). This architecture enables the Helping Hands system to evolve modularly, supporting new features like volunteer scheduling and donation tracking while aligning with agile methodologies for responsive and maintainable development.

UML DIAGRAM

https://tinyurl.com/j4ubuy8z



FUNCTIONAL (SOFTWARE) REQUIREMENTS

(AS PER PART 1 - AMMENDED WITH UML SOFTWARE REQUIREMENTS SATISFIED)

Functional Requirement	Requirement Name	UML INDICATION
User Registration and Authentication	The system will allow users (volunteers, staff, donors) to create accounts and log in securely.	- User class
Volunteer Management System	The system will track volunteer availability, skills, and tasks, and allow volunteers to update their profiles.	- Volunteer class - Program class - Task class
Donation Management System	The system will track donations, generate receipts, and store donor information.	- Donor class - Donation class - PaymentGateway class
Resource Distribution Tracking	The system will log and track the distribution of resources (food, clothing, etc.) to beneficiaries.	- Resource class - Beneficiary class
Program Outcome Reporting	The system will generate reports on program outcomes (e.g., families helped, volunteer hours logged).	- Program class - Report class
Volunteer Scheduling and Task Assignment	The system will assign tasks and schedules based on volunteer availability and display them on a dashboard.	- Volunteer, availability(attribute) - AssignTask(method)
Beneficiary Registration and Tracking	The system will allow beneficiaries to register for programs and track their application statuses.	- Beneficiary class
Automated Notifications for Volunteers and Donors	The system will send email notifications to volunteers (tasks) and donors (confirmation, thank you).	- Notification class
Search and Filter Functionality for Programs and Resources	The system will allow users to search and filter programs or resources by categories like location or type of support.	- Program class - Resource class
User Roles and Permissions Management	The system will control user access based on roles (admin, staff, volunteers, donors) to limit functionalities accordingly.	- User class - Admin class - Donor class - Beneficiary class - volunteer class

REFERENCES

Interserver (2016). *What Is MVC? Advantages and Disadvantages of MVC - Interserver Tips*. [online] Interserver Tips. Available at: https://www.interserver.net/tips/kb/mvc-advantages-disadvantages-mvc/.

Sheldon, R. (2023). *What Is model-view-controller (MVC)? | Definition from TechTarget*. [online] WhatIs.com. Available at: https://www.techtarget.com/whatis/definition/model-view-controller-MVC.