ADONAI

Informative Website for SMYM Mukkoottuthara Organization Management and User Engagement

Mini Project Report

Submitted by

Alan Antony

Reg. No.: AJC19MCA-I004

In Partial fulfillment for the Award of the Degree of

INTEGRATED MASTER OF COMPUTER APPLICATIONS

(INMCA)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2023-2024

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, “**ADONAI”** is the bona fide work of **ALAN ANTONY (Regno: AJC19MCA-I004)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Guide Name

Ms. Nimmy Francis

Coordinator Name

Ms. Meera Rose Mathew

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report **“ADONAI”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

Date:	**ALAN ANTONY**
KANJIRAPPALLY	**Reg: AJC19MCA-I004**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Ms. Nimmy Francis** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

ALAN ANTONY

ABSTRACT

.

The mini project "Adonai" is centered on the development of an informative website for the SMYM Mukkoottuthara organization. Its primary objective is to enhance user engagement and streamline the organization's management. The website accommodates various user roles, including administrators, normal users, and guest users. Administrators play a crucial role in maintaining the website's relevance by overseeing content, user accounts, and features. Normal users, once approved, gain access to engage with the platform's various features, fostering interaction and participation. Guest users have open access to explore the organization's information and support its initiatives through donations.

"Adonai" aims to create a comprehensive online platform that provides valuable resources and engagement opportunities for users of all roles. By enhancing the organization's online presence, this project seeks to attract new members, supporters, and volunteers. Efficient website management is at its core, where administrators ensure smooth content updates and interactions. This not only benefits administrators but also enhances the user experience for all others, thus contributing to the overall efficiency of the organization. In summary, "Adonai" is a valuable initiative that strengthens the organization's online presence, engagement, and management, catering to the needs of diverse user roles and the broader community it serves.

The mini project will serve as the foundation for the ADONAI, Main project, laying the groundwork for further extensions and additional functionalities

CONTENT

SL. NO	TOPIC	PAGE NO
1	INTRODUCTION	01
1.1	PROJECT OVERVIEW	02
1.2	PROJECT SPECIFICATION	03
2	SYSTEM STUDY	04
2.1	INTRODUCTION	05
2.2	EXISTING SYSTEM	05
2.3	DRAWBACKS OF EXISTING SYSTEM	07
2.4	PROPOSED SYSTEM	08
2.5	ADVANTAGES OF PROPOSED SYSTEM	09
3	REQUIREMENT ANALYSIS	10
3.1	FEASIBILITY STUDY	11
3.1.1	ECONOMICAL FEASIBILITY	11
3.1.2	TECHNICAL FEASIBILITY	11
3.1.3	BEHAVIORAL FEASIBILITY	12
3.1.4	FEASIBILITY STUDY QUESTIONNAIRE	12
3.2	SYSTEM SPECIFICATION	13
3.2.1	HARDWARE SPECIFICATION	13
3.2.2	SOFTWARE SPECIFICATION	13
3.3	SOFTWARE DESCRIPTION	14
3.3.1	DJANGO	14
3.3.2	PYTHON	14
3.3.3	POSTGRESQL	14
4	SYSTEM DESIGN	15
4.1	INTRODUCTION	16
4.2	UML DIAGRAM	16
4.2.1	USE CASE DIAGRAM	17
4.2.2	SEQUENCE DIAGRAM	19
4.2.3	STATE CHART DIAGRAM	20
4.2.4	ACTIVITY DIAGRAM	20
4.2.5	CLASS DIAGRAM	21
4.2.6	OBJECT DIAGRAM	22

4.2.7	COMPONENT DIAGRAM	22
4.2.8	DEPLOYMENT DIAGRAM	23
4.2.9	COLLABORATION DIAGRAM	24
4.3	USER INTERFACE DESIGN USING FIGMA	26
4.4	DATABASE DESIGN	28
5	SYSTEM TESTING	34
5.1	INTRODUCTION	35
5.2	TEST PLAN	36
5.2.1	UNIT TESTING	36
5.2.2	INTEGRATION TESTING	37
5.2.3	VALIDATION TESTING	37
5.2.4	USER ACCEPTANCE TESTING	37
5.2.5	AUTOMATION TESTING	38
5.2.6	SELENIUM TESTING	38
6	IMPLEMENTATION	46
6.1	INTRODUCTION	47
6.2	IMPLEMENTATION PROCEDURE	47
6.2.1	USER TRAINING	48
6.2.2	TRAINING ON APPLICATION SOFTWARE	48
6.2.3	SYSTEM MAINTENANCE	48
7	CONCLUSION & FUTURE SCOPE	49
7.1	CONCLUSION	50
7.2	FUTURE SCOPE	50
8	BIBLIOGRAPHY	52
9	APPENDIX	54
9.1	SAMPLE CODE	55
9.2	SCREEN SHOTS	67

List of Abbreviation

IDE	**-**	Integrated Development Environment
HTML	**-**	Hyper Text Markup Language.
CSS	**-**	Cascading Style Sheet
SQL	**-**	Structured Query Language
UML	**-**	Unified Modeling Language
VE	**-**	Virtual Environment

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The mini project "Adonai" is dedicated to the creation of an informative website for the SMYM Mukkoottuthara organization, with the primary goal of enhancing user engagement and optimizing organizational management. The website is designed to accommodate various user roles, including administrators, normal users, and guest users. Administrators take on the comprehensive responsibility of managing the website, ensuring its content remains current, moderating user interactions, and maintaining the integrity of user profiles. In contrast, normal users, once approved, gain access to engage with the website's numerous features, which may include participating in quizzes, polls, and viewing detailed reports of events. Guest users have the freedom to explore information about the organization and make contributions, fostering inclusivity and support. The overarching aim of the project is to construct a versatile platform that delivers valuable resources and engagement opportunities to users of varying roles, thereby enhancing the organization's online presence and management.

The mini project encompasses several vital modules. The admin module focuses on user management, the maintenance of blood donor lists, and parish directories, gallery management, generating detailed reports for conducted events, creating quizzes and polls, and managing donations. Normal users have the ability to create accounts, access information, engage in quizzes and polls, view comprehensive event reports, and make donations. Guest users can access information about the organization and also contribute financially. The project introduces various modules, including registration for user account creation, a login system for admin, users, and accountants, maintenance of blood donor lists and parish directories, a career guidance forum allowing users to seek advice from experts for informed career decisions, a donation module equipped with a secure payment gateway, and reports to provide detailed insights into events conducted, managed by administrators, and viewed by users. In essence, "Adonai" aspires to be a dynamic and user-friendly platform that streamlines management and engagement, catering to a diverse set of users while bolstering the organization's online presence.

1.2 PROJECT SPECIFICATION

The various system specification that has been used in developing both the frontend and the back end of the project are being discussed below.

1.2.1 Front-End Technologies:

HTML, CSS, BOOTSTRAP, JAVASCRIPT are utilized to implement the frontend.

HTML (Hyper Text Markup Language): HTML is used to format text documents on the web, providing the structure and content of web pages.

CSS (Cascading Style Sheets): CSS is employed for styling and formatting web documents, enhancing the visual presentation of the platform.

Bootstrap: Bootstrap, an open-source front-end framework, is utilized to design and develop responsive websites and web applications, ensuring a consistent and mobile-friendly user experience.

JavaScript (JS): JavaScript, a dynamic programming language, enhances user interactivity and functionality, primarily used for client-side scripting in web browsers.

1.2.2 Front-End Technologies:

The back end is implemented using Python Flask, Django and PostgreSQL which is used to design the databases.

Python Flask: The back end of the Adonai platform is implemented using Python
Flask, a lightweight web framework, offering essential tools and features for building web applications efficiently.

Django: Django, a high-level Python web framework that simplifies web development. Django could be integrated into the project for enhanced web application development capabilities.

PostgreSQL: PostgreSQL, a powerful open-source relational database system, securely stores and manages essential data related to users, blood donors, images, parish members, and transactions for the Adonai platform. It ensures data integrity, reliability, and scalability, contributing to a robust database system.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis involves the process of gathering and comprehending data, pinpointing issues, and utilizing this information to suggest enhancements to the system. It's essentially a problem-solving endeavor that necessitates extensive communication between system users and developers. System analysis is an integral phase in any system development process. It involves a holistic view of the system, the identification of inputs, and a thorough examination to locate areas of concern. Proposed solutions are then presented for review. The proposal is subject to revision upon user request until the user is content with the outcome.

2.2 EXISTING SYSTEM

In the case of "Adonai" for SMYM Mukkoottuthara organization, there is no existing digital system. The organization have relied on manual and non-digital methods for communication and management. The "Adonai" project represents a transition from these traditional processes to an online platform, aimed at significantly improving efficiency, accessibility, and user engagement for the organization.

2.2.1 NATURAL SYSTEM STUDIED

"In the development of the 'Adonai' website for SMYM Mukkoottuthara organization, a thorough study of the organization's existing natural system was conducted, encompassing various facets of its operations and user interactions. The natural system under study involves a conventional and predominantly physical approach to management and engagement. Various key components were explored:

1. **User Engagement and Management:** The organization operates through traditional methods, with user approvals and registration handled manually. New members are required to complete physical registration forms and submit them for approval. User engagement primarily occurs through in-person meetings held on specific days.

2. **Data Accessibility Challenges:** Challenges faced by the organization revolve around data accessibility. Valuable resources like the blood donor list were traditionally hard to access, requiring members or relevant personnel to contact the executive team for information. The

user approval system was initially slow, hindering the onboarding of new members.

3. **Content Management:** The management of the gallery and program reports was historically less efficient. Reports were typically presented during meetings following program events, creating delays in sharing information.

4. **Integration of Secure Payment Gateway:** A significant enhancement brought by the 'Adonai' project is the integration of a secure payment gateway, enabling users to make donations to the organization online. This innovation simplifies and secures the donation process.

In summary, the 'Adonai' project serves to digitize and streamline various aspects of the organization's management, improve data accessibility, and enhance user engagement. The integration of a secure payment gateway further modernizes and secures the donation process. By studying the existing natural system, the 'Adonai' project aims to offer a more efficient and user-centric experience within the organization, addressing existing challenges while boosting its online presence."

2.2.2 DESIGNED SYSTEM STUDIED

"In the process of developing the 'Adonai' website for SMYM Mukkoottuthara organization, a comprehensive examination of the designed system was conducted. This study encompasses the architecture, features, and functionalities that are integral to the platform. The designed system study involves several key components:

1. **System Architecture:** The study explores the architectural design of 'Adonai,' outlining its components, their interactions, and the overall structure of the system. It highlights how data flows between different modules and layers, ensuring a smooth and efficient operation.
2. **User Roles and Permissions:** An essential aspect of the designed system is the definition of user roles, including administrators, normal users, and guest users. The study details their respective permissions and responsibilities, ensuring controlled access and user engagement.
3. **Registration and User Management:** 'Adonai' introduces a user registration module, allowing users to create accounts with specific credentials. For normal users, registration

requires approval from administrators, streamlining the user onboarding process.

4. **Blood Donors List and Parish Directory:** The designed system facilitates the management and accessibility of essential data, such as the blood donor list and a comprehensive directory of parish members. This ensures crucial information is readily available.
5. **Gallery Management:** The platform offers content creators tools and features to upload, edit, and manage their work, fostering a supportive ecosystem for artists and contributors.
6. **Event Reports and Donations:** The study covers the creation of detailed reports for events, allowing administrators to add event reports and users to view them. Additionally, it incorporates a donation module equipped with a secure payment gateway, enabling users to make contributions seamlessly.
7. **Career Guidance Forum:** To assist users in making informed career decisions, the system introduces a career guidance forum, enabling them to seek advice from experts within the organization.
8. **User Interface and Experience:** The study addresses the design of the user interface, emphasizing user-friendliness and ease of navigation. This includes features for image customization, search functionality, and categorization.
9. **Security Measures:** Security is a paramount consideration, and the study details the security measures implemented to protect user data and financial transactions, ensuring a secure user experience.
10. **Testing and Quality Assurance:** Rigorous testing procedures are carried out to ensure the platform's functionality, performance, and user experience meet the highest standards.

By studying the designed system, the 'Adonai' project gains insights into how the envisioned platform will operate, the features it will offer, and how it addresses user needs and challenges. This knowledge serves as the foundation for the development and implementation of the 'Adonai' website, offering a more efficient and user-centric experience within the organization while enhancing its online presence."

2.3 DRAWBACKS OF EXISTING SYSTEM

- Manual and Time-Consuming Processes
- Limited Data Accessibility
- Inefficient User Engagement
- Data Security and Accessibility
- Lack of Real-Time Information Sharing

- Limited Fundraising Opportunities
- Insecure Transactions

2.4 PROPOSED SYSTEM

"The proposed system, 'Adonai,' represents a transformative solution designed to alleviate the challenges faced by the SMYM Mukkoottuthara organization and introduce a range of improvements. This section outlines the key aspects of the proposed system:

1. **Efficient User Management:** 'Adonai' streamlines the user management process, eliminating manual paperwork for new member registration and approvals. A structured user hierarchy, including administrators, normal users, and guest users, ensures defined roles and responsibilities.

2. **Data Accessibility and Centralized Repository:** The platform provides a centralized repository for crucial information such as the blood donor list and the parish directory. This data is readily accessible to authorized personnel, reducing the need for direct contact with the executive team.

3. **Streamlined User Engagement:** 'Adonai' introduces digital tools and features for efficient user engagement. It offers a virtual space for members to participate in discussions, access valuable resources, and contribute to the organization's initiatives.

4. **Real-Time Information Sharing:** The system enables the immediate publication of gallery content and event reports. Reports are available to users as soon as programs are conducted, ensuring timely information sharing.

5. **Enhanced Communication:** Communication among members and with the executive team is facilitated through digital means. This reduces delays and enhances the efficiency of conveying important information or updates.

6. **Secure Payment Gateway:** The system integrates a secure payment gateway, enabling users to make donations to the organization with confidence and ensuring secure financial transactions.

7. **User-Friendly Interface:** 'Adonai' is designed with a user-friendly interface, making navigation and interaction seamless and intuitive.

8. **Advanced Features for Member Engagement:** The platform offers a range of features, including quiz and poll creation, event participation, and user forums, enhancing member engagement and interaction.

9. **Content Management for Members:** Normal users have a dedicated space to create accounts, access information, participate in quizzes and polls, view event reports, and make donations.

10. **Security Measures:** Security is a paramount consideration in 'Adonai,' with robust measures in place to protect user data, financial transactions, and the integrity of content.

11. **Documentation and Support:** Comprehensive documentation, including user guides, ensures that members can efficiently use the platform, and administrators can maintain it effectively.

In summary, 'Adonai' serves as a dynamic and user-centric platform that addresses the limitations of the existing system. It streamlines operations, enhances data accessibility, fosters efficient user engagement, and introduces a secure and user-friendly online environment for the SMYM Mukkoottuthara organization, ultimately boosting its online presence and management capabilities."

2.5 ADVANTAGES OF PROPOSED SYSTEM

- Efficiency and time savings in user registration and approval
- Centralized data repository for easy information access
- Enhanced user engagement through virtual spaces and interactive features
- Real-time sharing of event updates and reports
- Improved communication tools within the platform
- Secure financial transactions via an integrated payment gateway
- User-friendly interface for easy navigation and interaction
- Advanced user engagement features like quizzes, polls, and user forums
- Content management tools for members
- Robust security measures to protect data and transactions
- Comprehensive documentation and user guides for ease of use and system maintenance.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

A feasibility study is a quick evaluation done to see if a project or business venture is feasible and worth pursuing. In order to detect potential difficulties and opportunities, it entails assessing technical, economic, legal, operational, and scheduling aspects. The study offers key information that stakeholders can use to decide whether to move forward with the project, change its scope, or stop working on it altogether. A successful feasibility study shows that the project is feasible, financially sustainable, and compliant with legal standards, whereas a failure indicates that more research or other choices may be required. Overall, a feasibility study is an essential planning tool that assists companies in determining the viability of a project and in making wise decisions prior to allocating resources.

3.1.1 Economical Feasibility

A feasibility study that focuses on determining the financial sustainability of a proposed project or business venture must consider economic feasibility as a key component. To ascertain whether the project is financially viable and lucrative entails examining the predicted costs, potential revenue sources, and anticipated return on investment (ROI).

- The cost of the hardware and software?

  ✓*The resources are already available.*

- What is the estimated cost to develop and implement the website with the additional functionalities?

  ✓*All resources such as libraries and frameworks used are open source and free. The website itself can be hosted locally or on a free tier of hosting on various cloud platforms*

3.1.2 Technical Feasibility

Technical feasibility A feasibility study's technical feasibility assessment determines if a proposed project or business venture is technically feasible and whether the necessary technology and resources are already in place or can be acquired in order to carry out the project successfully. Understanding if the project can be produced, launched, and maintained utilizing existing or practical technological solutions requires this examination.

● Do stakeholders need to have expertise in the technologies used?

✓*No*

● Is the required technology and infrastructure readily available or easily obtainable to support the implementation website

✓*Yes*

● How compatible is the proposed platform with various web browsers and devices (e.g., desktops, tablets, mobile phones)?

✓*All modern browsers and devices support the platform.*

● Are there any technical constraints related to budget limitations that may impact the development and deployment of certain functionalities?

✓*No*

3.1.3 Behavioral Feasibility

Behavioral feasibility is an evaluation conducted during a feasibility study to assess whether the proposed project or business venture is socially and culturally acceptable and whether the intended users or stakeholders will readily adopt and embrace the changes brought about by the project.

● Are the defined user roles (admin, normal user, accountant, guest user) appropriate and relevant to the organization's needs?

✓*Yes, the defined user roles are relevant to organizations need and have specific roles.*

● What features or functionalities do potential normal users find most engaging and valuable?

✓*The career guidance forum, blood donors list, parish directory, info regarding events and program reports.*

3.1.4 Feasibility Study Questionnaire

● What are the different user roles that need to be supported by the website?

✓*Admin, Normal User, Accountant, Guest User).*

● What are the essential functionalities and features required for each user role?

✓*User management, blood donors list, parish directory, event management, quizzes, polls, financial management, career interest analyzer, etc.*

- Describe the desired user registration and approval process for normal users?

  ✓*Normal users must register and wait for the approval from the admin to create account.*

- How would you like to manage the waiting period for admin approval?

  ✓*Once a user register, there should be option for receiving an email to admin regarding the user registration and admin can approve the request after verification.*

- What are the specific financial management features required for the accountant role?

  ✓ *Uploading and saving receipts, recording credit and debit details, generating excel sheets of transactions.*

- Are there any specific resources that you can provide for the website development?

  ✓*Yes, blood donors list, parish directory and images for gallery can be provided.*

- Are there any additional functionalities or features that you would like to see on the website?

  ✓*An option for generating virtual id card, option for viewing and uploading program reports.*

- How would you like to provide user support and assistance?

  ✓*Through a contact form, contact forms will allow users to submit specific queries or issues through a form on the website. So, we can then respond to these inquiries via email or phone.*

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- 12th Gen Intel(R) Core (TM) i3-1240P
RAM	- 4 GB or Above
Hard disk	- 500GB or Above

3.2.2 Software Specification

Front End	-	HTML, CSS, JS, Bootstrap
Back End	-	Django, Python
Database	-	PostgreSQL Client
Client on PC	-	Windows 7 and above.
Technologies used	-	JS, HTML5, AJAX, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 Django:

Django is a high-level, open-source web framework written in Python, ideal for creating robust and scalable web applications. It adheres to the Model-View-Controller (MVC) architectural pattern, ensuring code cleanliness and maintainability. Django simplifies database interactions with its built-in Object-Relational Mapping (ORM) system. Its extensive, well-documented library accelerates development, covering features like authentication, URL routing, and templating. Additionally, Django prioritizes security, guarding against common web vulnerabilities, and forming a solid foundation for building secure web applications.

3.3.2 Python:

Python is a high-level, dynamically-typed, and interpreted programming language recognized for its readability and user-friendliness. It accommodates multiple programming paradigms, encompassing procedural, object-oriented, and functional approaches. Python boasts an extensive standard library with modules and packages suited for diverse tasks, spanning web development to scientific computing. Its unique syntax, which utilizes indentation for code structure, encourages clean and uniform coding practices. Python's broad community support and a plethora of third-party libraries render it an adaptable and versatile option for an array of applications, such as web development, data analysis, machine learning, and automation.

3.3.3 PostgreSQL:

PostgreSQL is a robust, open-source relational database management system (RDBMS) renowned for its reliability and advanced features. It excels in handling complex data structures and relationships, making it a preferred choice for various applications. PostgreSQL supports SQL queries, enabling seamless data retrieval and manipulation. Its extensibility and support for custom functions and data types allow developers to tailor the database to specific project needs. PostgreSQL's strong emphasis on data integrity, transactions, and concurrency control ensures the consistency and security of data. Its active community and continuous development guarantee ongoing support, enhancements, and a bright future for this powerful RDBMS.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

System design is the crucial phase where a new system solution takes shape. This phase is dedicated to the detailed implementation of a feasible system, with a strong focus on translating design specifications into performance requirements. The system design encompasses two distinct phases:

1. Logical Design: During the logical design phase, the analyst meticulously outlines the system's inputs (sources), outputs (destinations), databases (data stores), and procedures (data flows) in a format aligned with user requirements. This phase involves specifying the user's needs at a level that effectively determines how information flows in and out of the system and how data resources are managed. Key tools used during the logical design include data flow diagrams and database design.
2. Physical Design: Following the logical design phase, the system progresses to physical design or coding. The physical design stage brings the system to life by defining precise design specifications, detailing what the candidate system must accomplish. Programmers then create the necessary code, enabling the system to receive user input, process the data, and generate the required output, either in the form of hard copy reports or display on the screen.

This approach to system design allows for a seamless transition from logical planning to the actual implementation of the system, ensuring that it aligns with user needs and requirements.

4.2 UML DIAGRAM

A UML diagram is a visual representation of a system that leverages the Unified Modeling Language (UML). It serves the purpose of depicting a system, including its primary actors, roles, actions, artifacts, or classes. The goal is to provide a clearer understanding of the system, support modifications, facilitate maintenance, and aid in documentation.

UML diagrams are broadly categorized into two groups, each with its subcategories:

- Structural Diagrams
- Behavioral Diagrams

Structural Diagrams: These diagrams capture the static aspects of a system, representing elements that form the core structure and remain relatively stable. Static components include classes, interfaces, objects, components, and nodes. The four structural diagram types are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

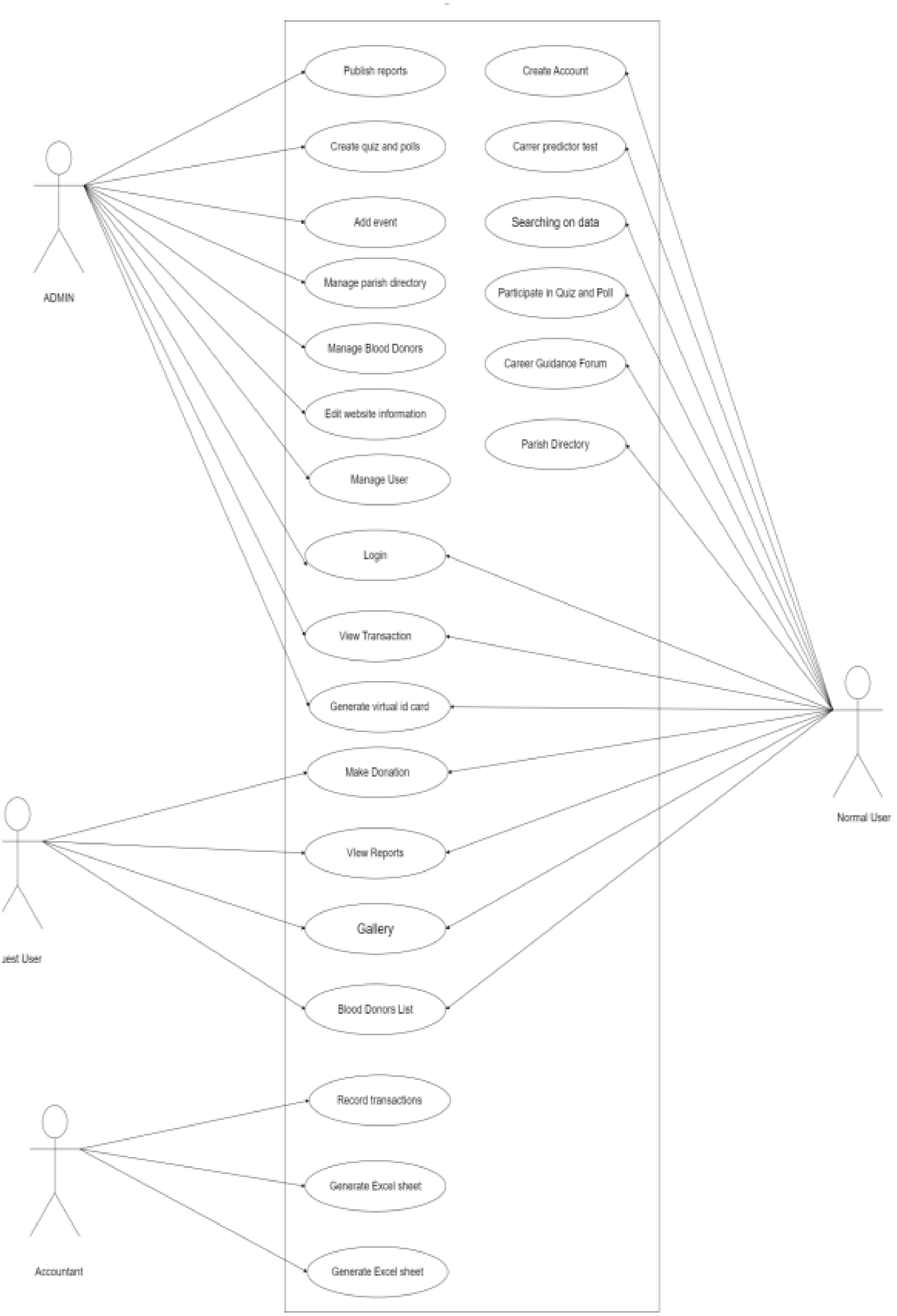
Behavioral Diagrams: Behavioral diagrams in UML focus on the dynamic aspects of a system. They showcase the interactions and behaviors of various components or objects within the system. These diagrams help in understanding how the system functions during its runtime. The four primary types of behavioral diagrams are:

- Use Case Diagram
- Activity Diagram
- Sequence Diagram
- State Machine Diagram

4.2.1 USE CASE DIAGRAM

A use case diagram is a visual representation within Unified Modeling Language (UML) that portrays interactions between a system and its external actors or users. It displays different use cases, representing distinct functions or actions the system can execute, and elucidates how these use cases are instigated and executed by the actors. Use case diagrams provide insights into a system's functional requirements and the roles of various entities in accomplishing particular objectives or tasks within the system.

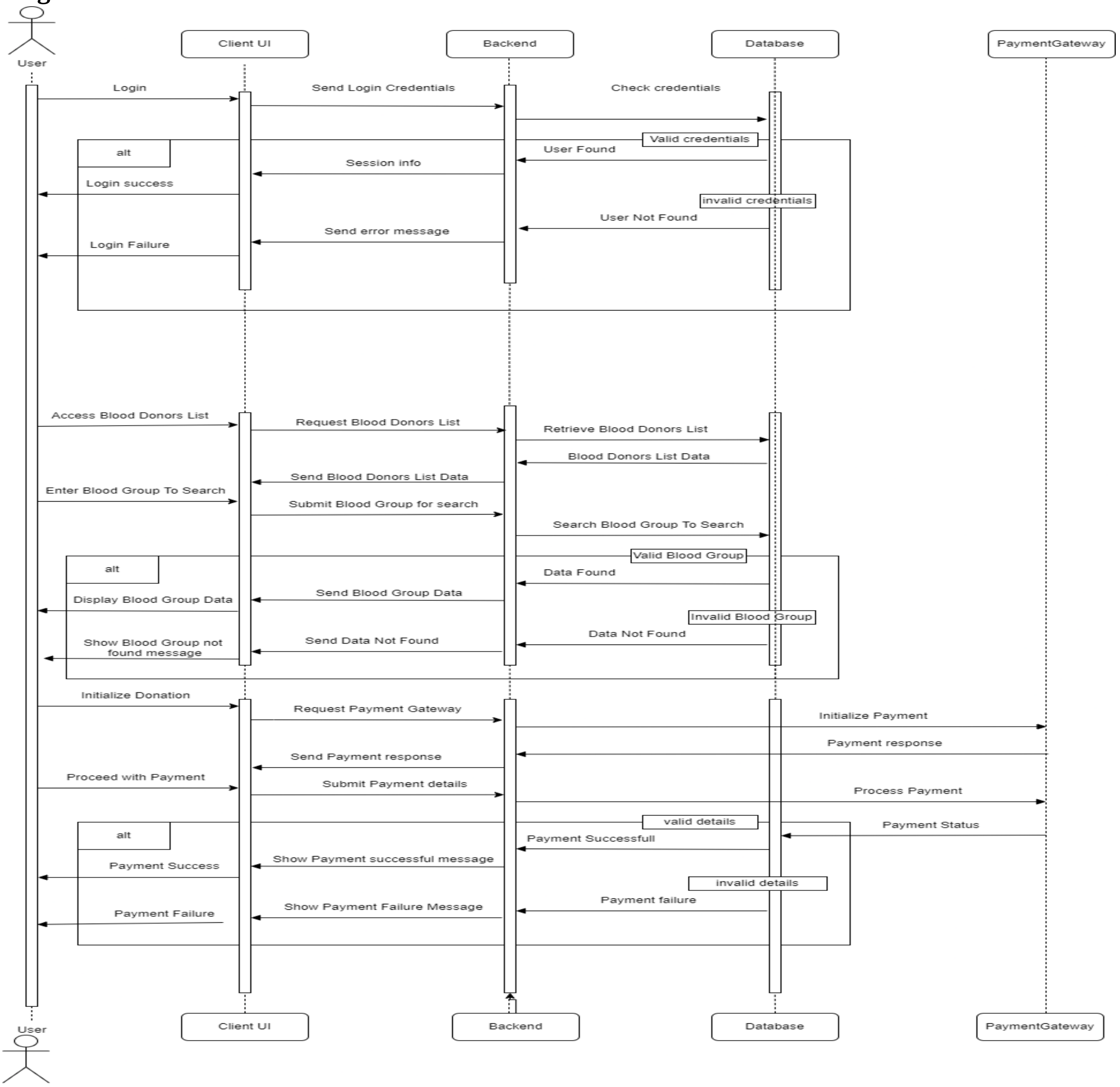
Diagram:


Publish reports
Create Account
Create quiz and polls
Carrer predictor test
ADMIN
Add event
Searching on data
Manage parish directory
Participate in Quiz and Poll
Manage Blood Donors
Career Guidance Forum
Edit website information
Parish Directory
Manage User
Login
View Transaction
Generate virtual id card
Normal User
Make Donation
View Reports
uest User
Gallery
Blood Donors List
Record transactions
Generate Excel sheet
Accountant
Generate Excel sheet

4.2.1 SEQUENCE DIAGRAM

A sequence diagram is a visual representation used in Unified Modeling Language (UML) to illustrate the interactions and communication between various objects or components within a system. It provides a chronological depiction of the order and timing of operations and messages passed between these objects, helping to understand the dynamic behavior of the system during its execution. Sequence diagrams are valuable for modeling and visualizing how different elements within a system collaborate and respond to specific events or interactions, aiding in system analysis and design.

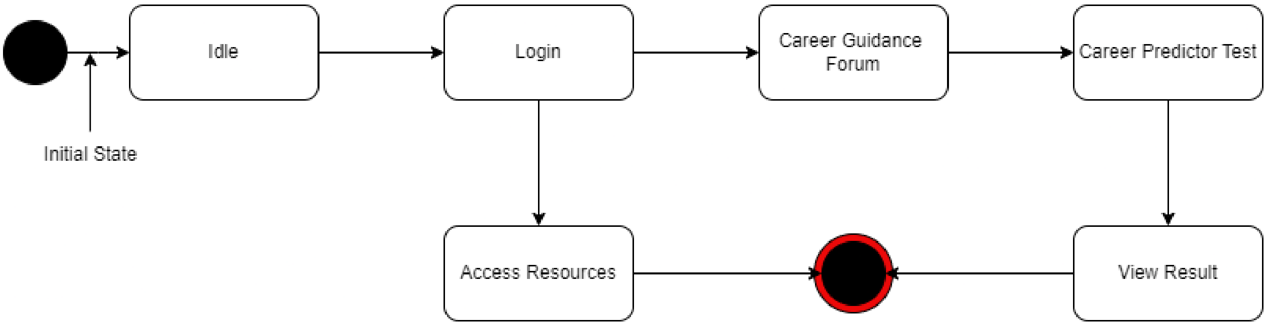
Diagram:



4.2.2 State Chart Diagram

A State Chart Diagram is a visual representation in Unified Modeling Language (UML) that illustrates the various states an object or system can be in and the transitions between these states. It is a way to model the behavior of an entity, showing the conditions and events that cause it to change from one state to another. State Chart Diagrams help in understanding how objects or systems respond to different events and under what conditions they transition from one state to another.

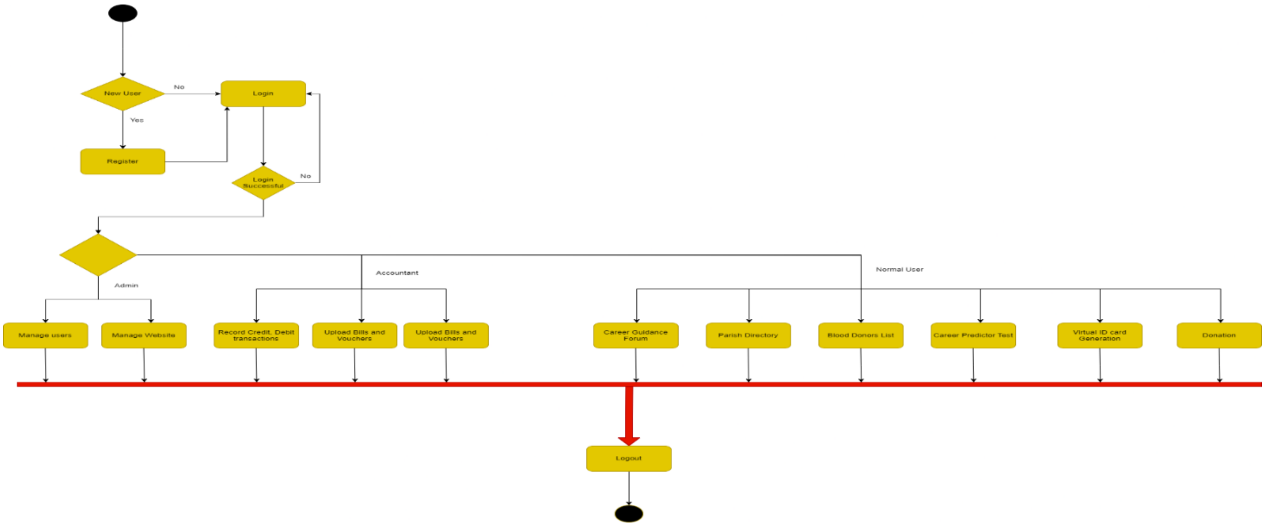
Diagram:



4.2.2 Activity Diagram

An activity diagram is a graphical representation used in Unified Modeling Language (UML) to illustrate the dynamic aspects of a system's behavior. It presents a visual flowchart-like depiction of the activities or actions within the system and the sequence in which they occur. Activity diagrams help in modeling and understanding complex workflows, processes, or business logic by showing how different activities are linked, the conditions under which they are executed, and the order in which they take place.

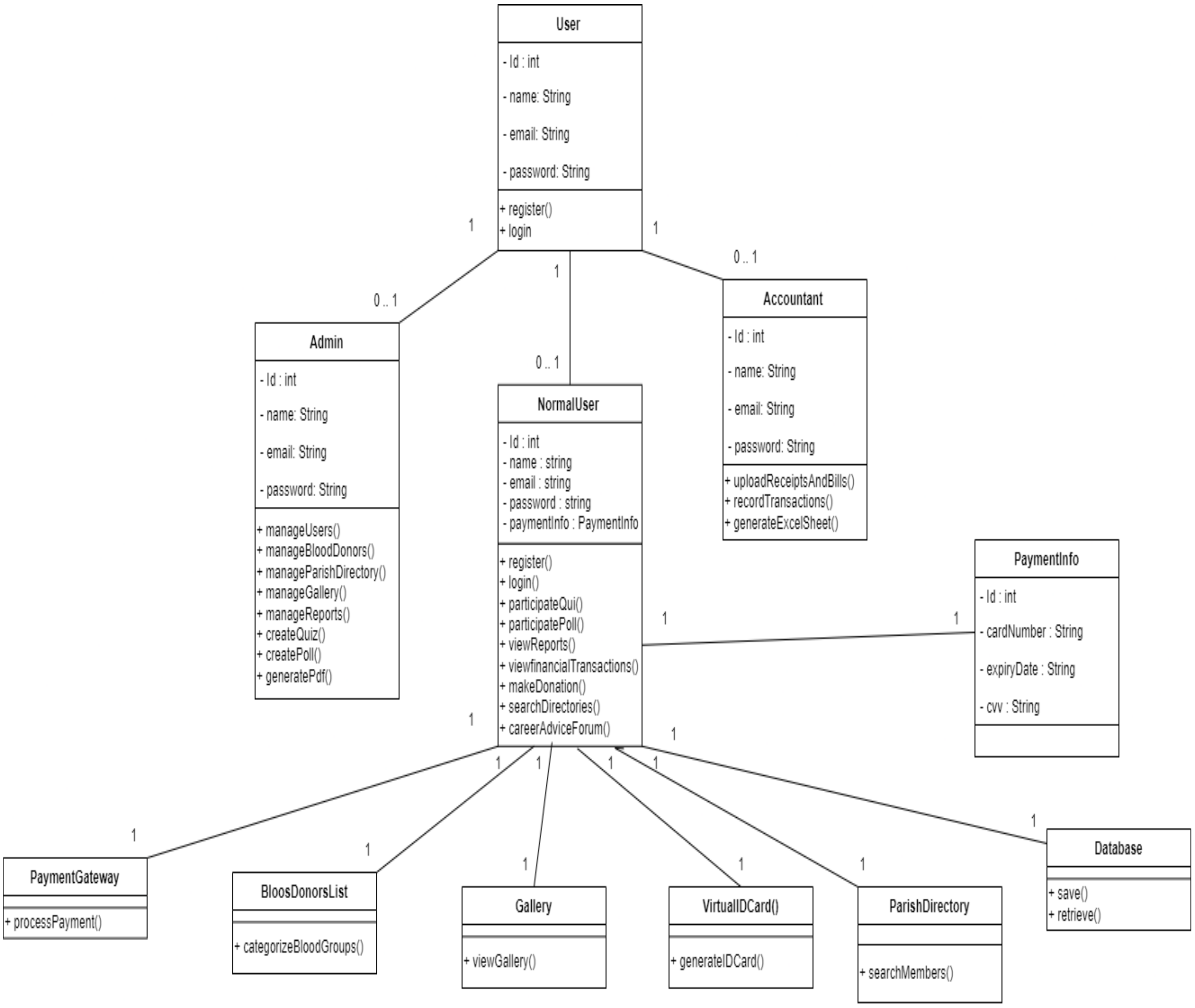
Diagram:



4.2.3 Class Diagram

A class diagram is a visual representation used in Unified Modeling Language (UML) to illustrate the structure of a system by depicting classes, their attributes, and relationships between classes. It serves to provide a clear and concise view of the static structure of a system, highlighting the various objects or components, their properties, and the associations or connections between them. Class diagrams are valuable tools for understanding the design and organization of a system, facilitating communication among developers and stakeholders, and aiding in the planning and development of software systems.

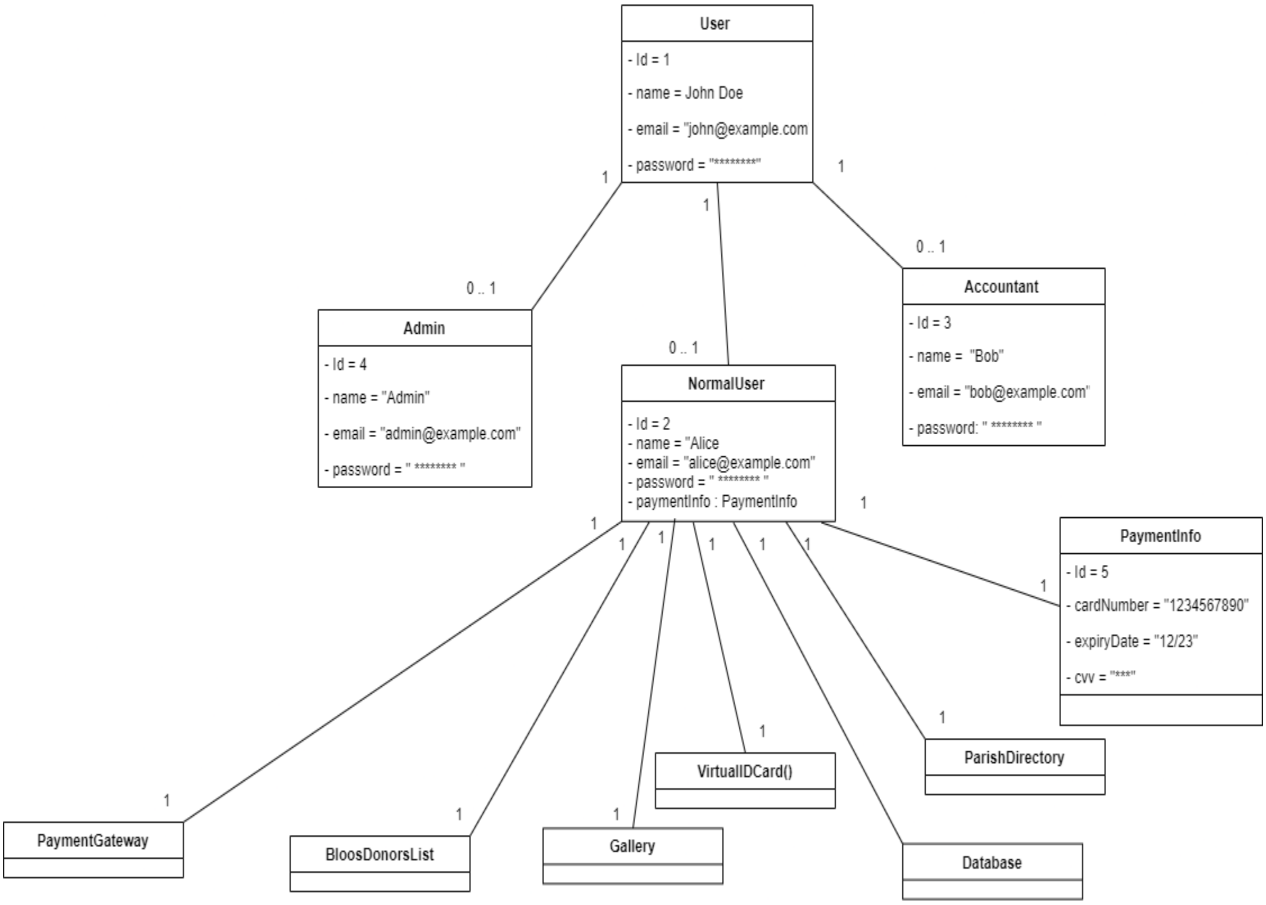
Diagram:



4.2.4 Object Diagram

An object diagram is a visual representation in Unified Modeling Language (UML) that focuses on depicting a specific instance or occurrence of objects and their relationships within a system. It provides a snapshot view of the system at a particular moment, showing the objects' attributes, values, and associations in a particular context. Object diagrams help in understanding the real world manifestation of classes and their interactions in a specific scenario.

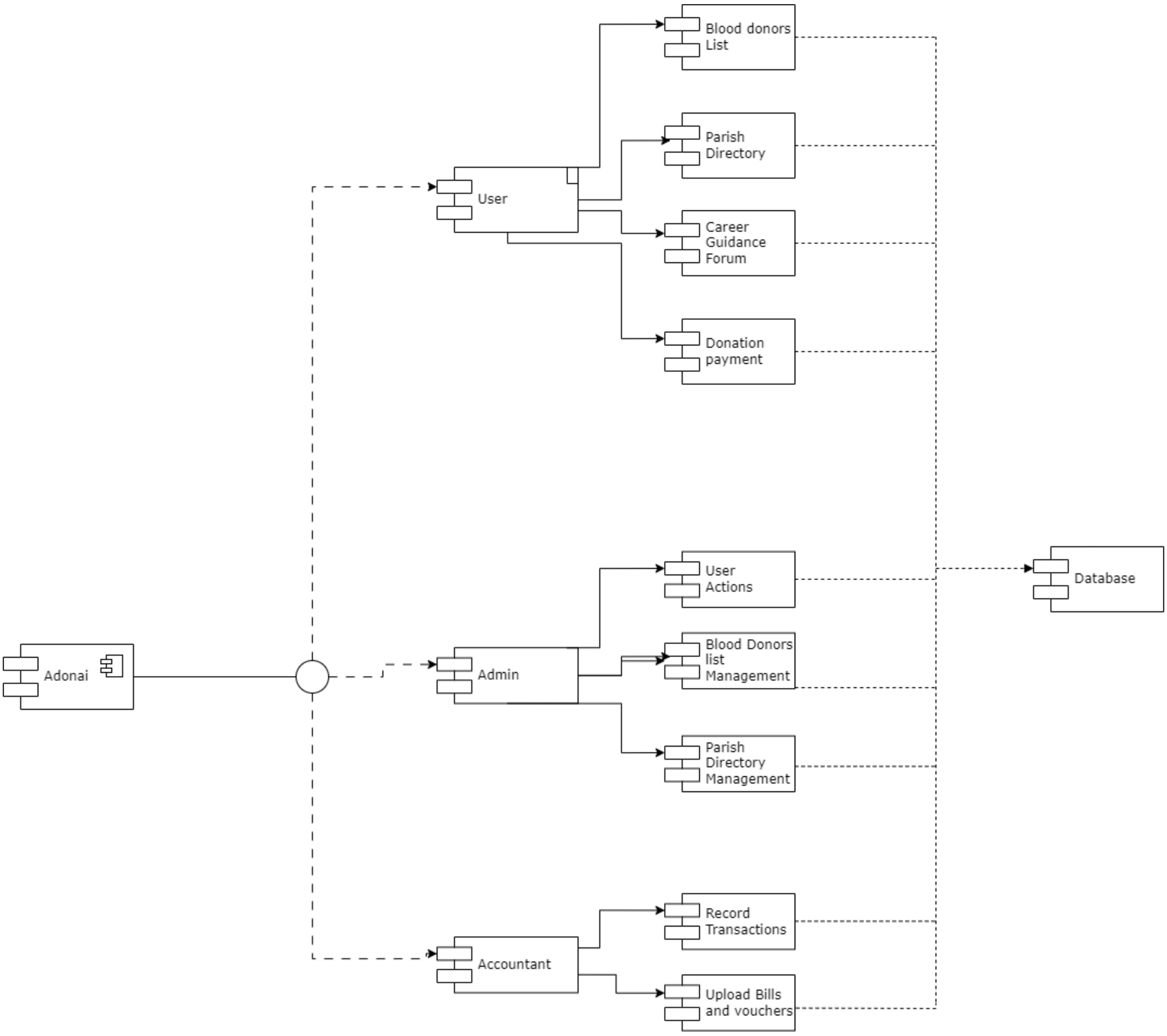
Diagram:



4.2.5 Component Diagram

A component diagram is a visual representation in Unified Modeling Language (UML) that illustrates the architectural structure of a system by breaking it down into components or modules. These components represent the physical and logical building blocks of the system, such as software modules, libraries, or executable files. Component diagrams demonstrate how these components interact with one another, showcasing their relationships, dependencies, and connections. They provide a clear and concise view of the system's internal organization and the distribution of responsibilities among its components.

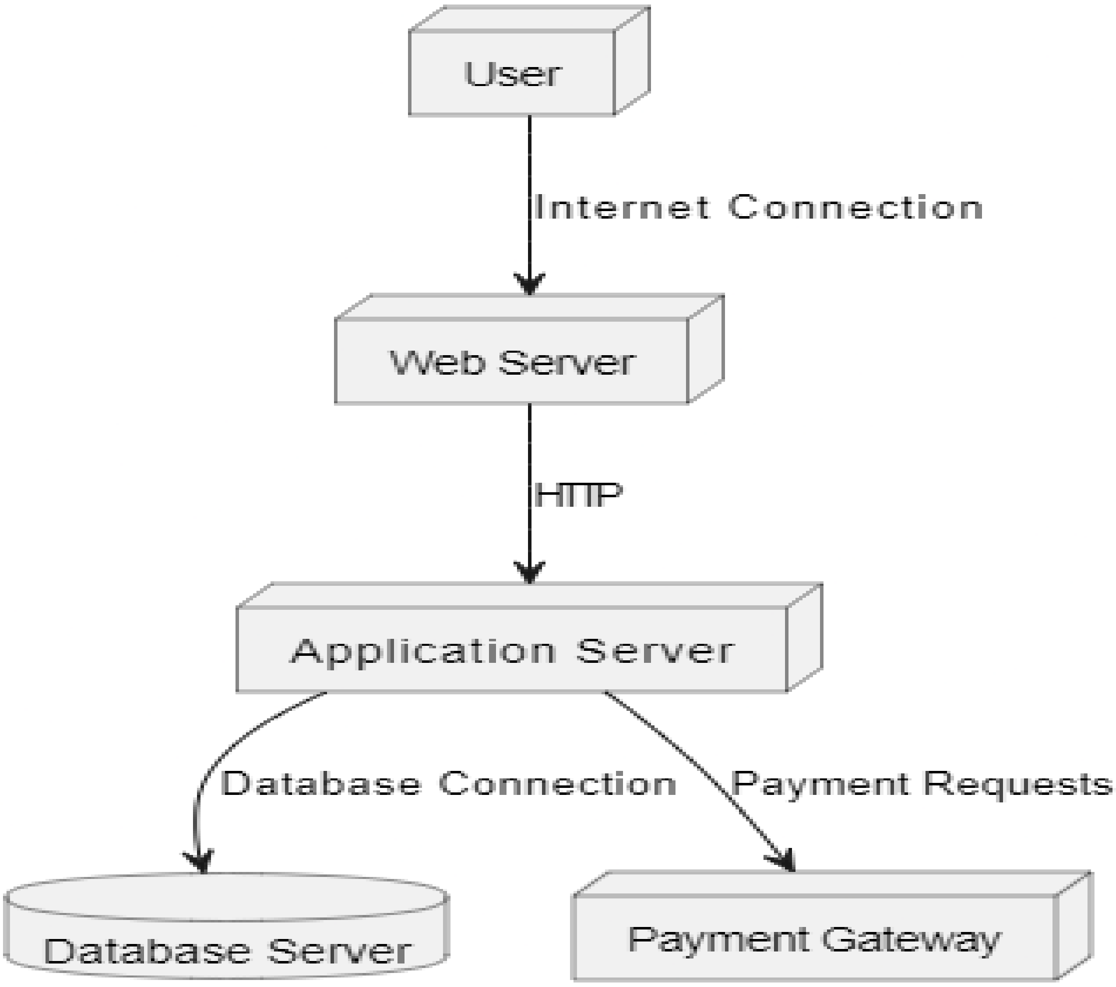
Diagram



4.2.8 Deployment Diagram

A deployment diagram, in the context of Unified Modeling Language (UML), is a visual representation that illustrates the physical arrangement of hardware components and software artifacts within a system or application. This diagram highlights how various nodes, which can be computers, servers, or other hardware devices, interact to support the execution of software components and services. Deployment diagrams are valuable for understanding the distribution and configuration of a system's components in a real-world, physical environment, aiding in system design and maintenance.

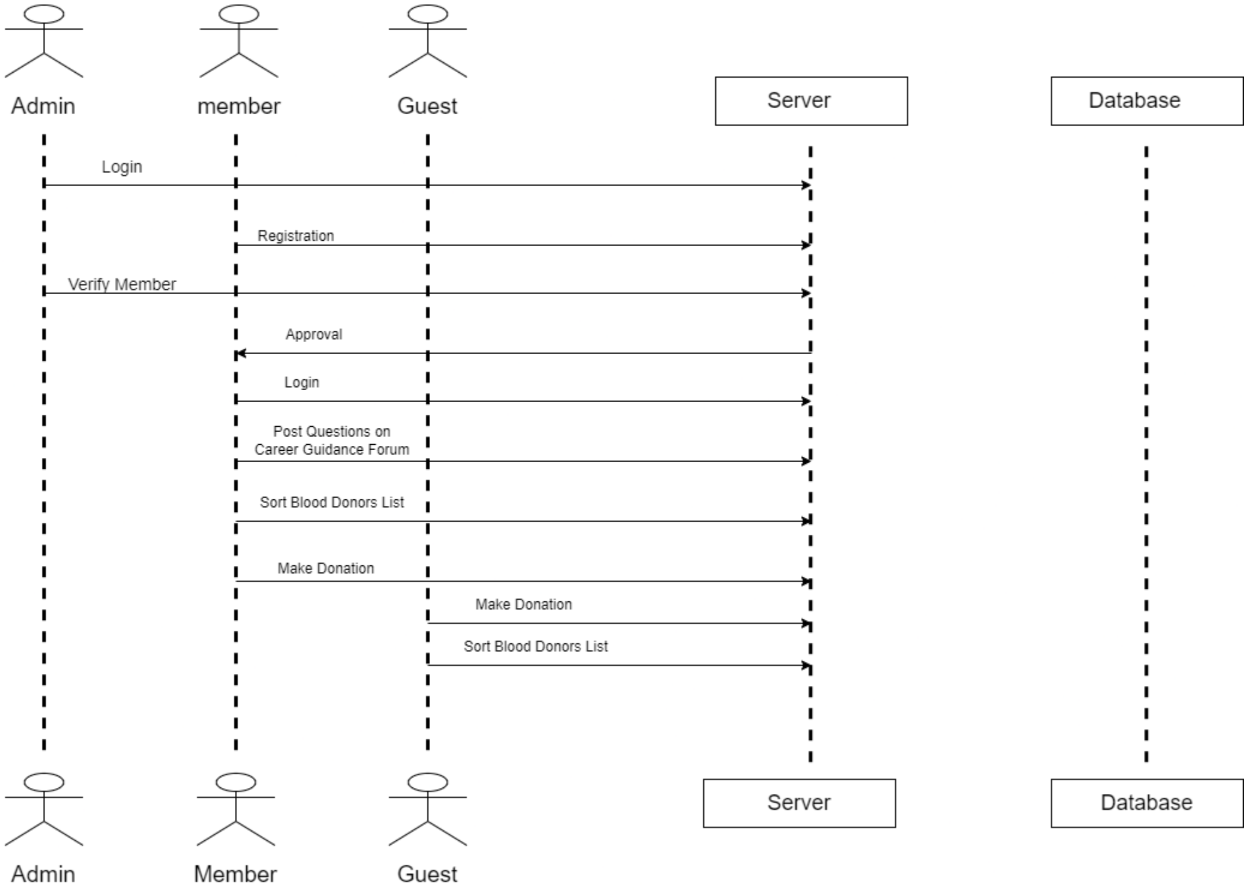
Diagram



4.2.9 Collaboration Diagram

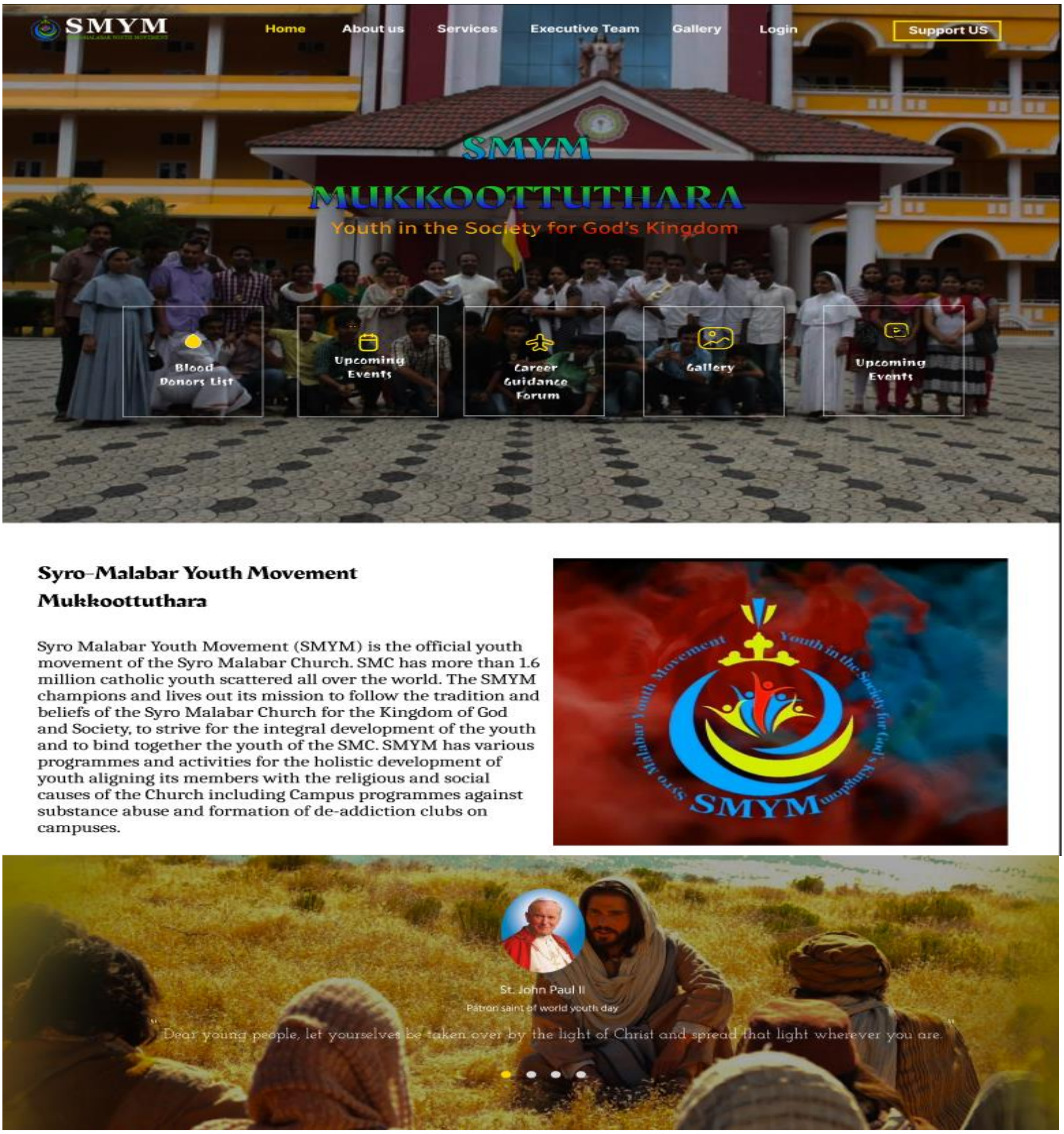
A collaboration diagram, in the context of Unified Modeling Language (UML), is a graphical representation that depicts the interactions and relationships between various objects or components within a system. This diagram illustrates how these objects collaborate to achieve specific tasks or functions. It emphasizes the flow of messages and interactions between objects, aiding in the understanding of the dynamic behavior of the system during runtime.

Diagram


Admin
member
Guest
Server
Database
Login
Registration
Verify Member
Approval
Login
Post Questions on
Career Guidance Forum
Sort Blood Donors List
Make Donation
Make Donation
Sort Blood Donors List
Admin
Member
Guest
Server
Database

4.3 USER INTERFACE DESIGN USING FIGMA

Form Name: Home Page

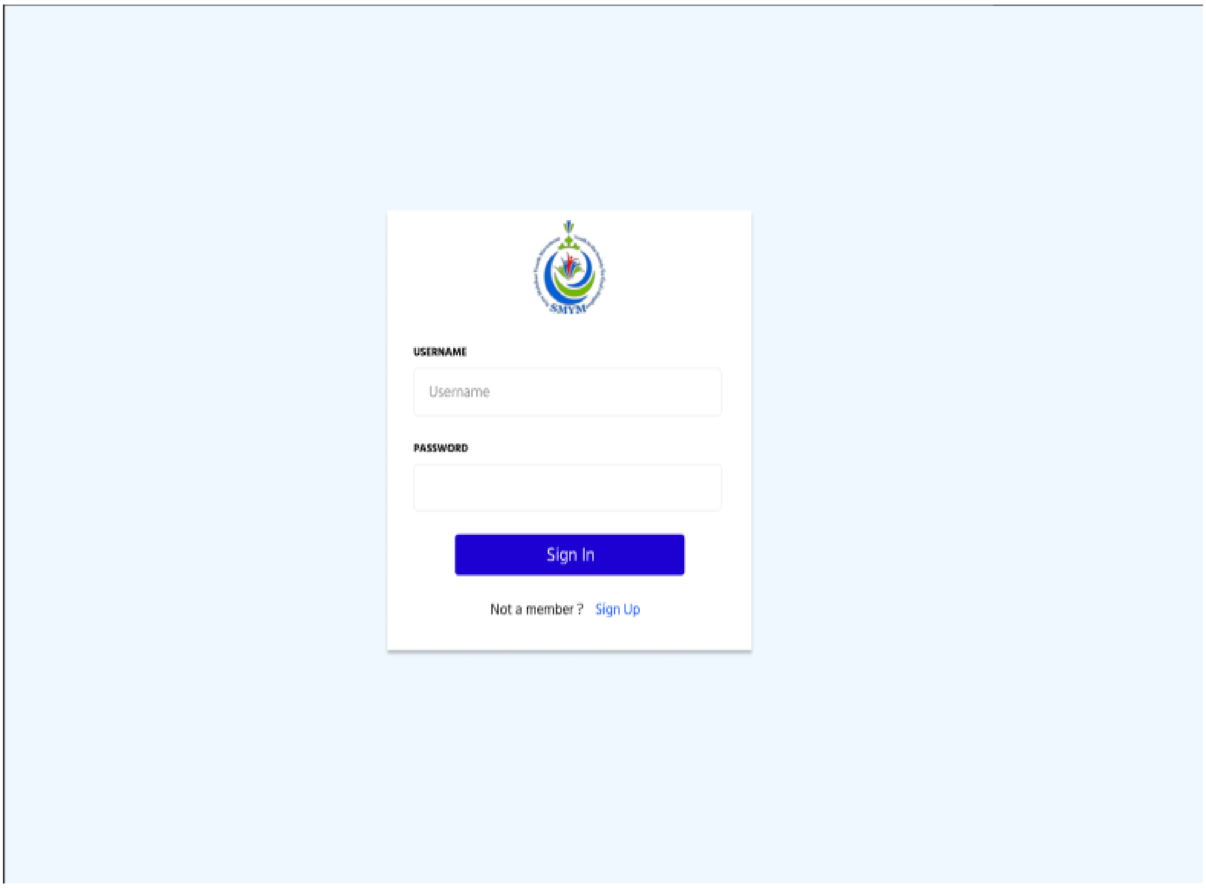


Syro-Malabar Youth Movement Mukkoottuthara

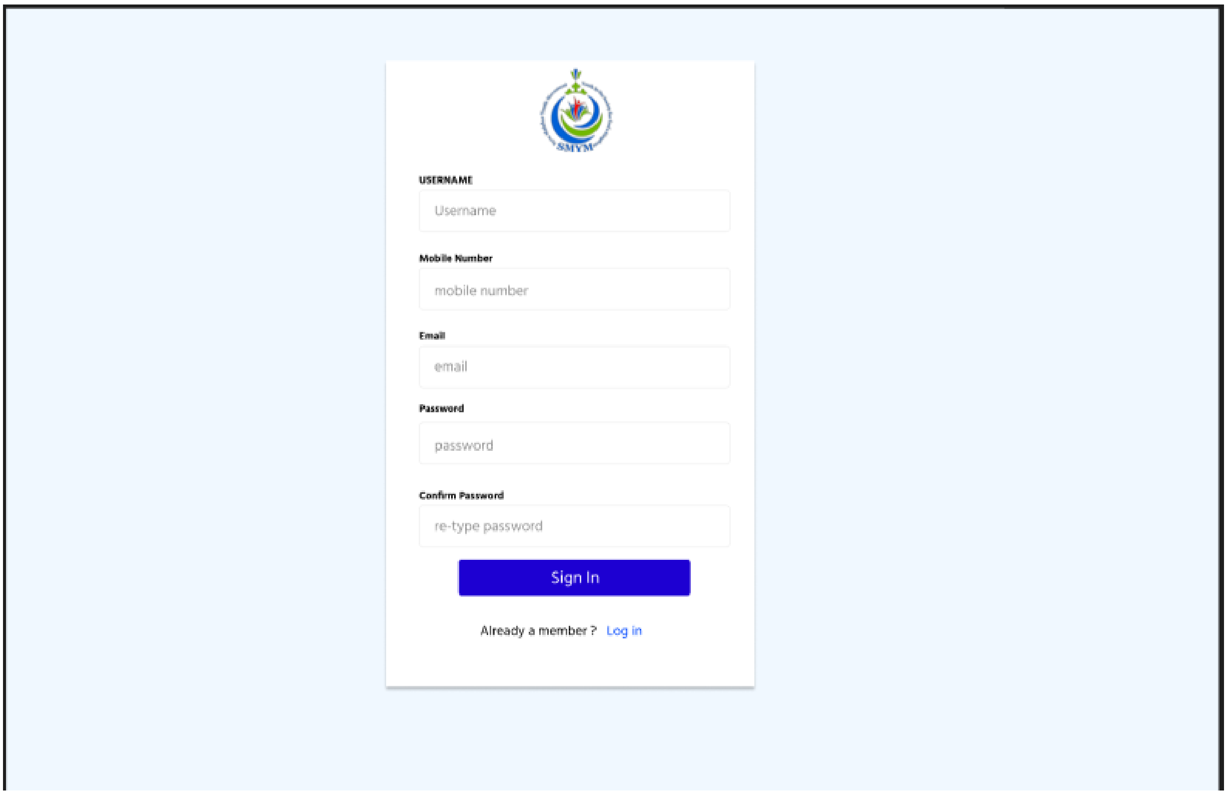
Syro Malabar Youth Movement (SMYM) is the official youth movement of the Syro Malabar Church. SMC has more than 1.6 million catholic youth scattered all over the world. The SMYM champions and lives out its mission to follow the tradition and beliefs of the Syro Malabar Church for the Kingdom of God and Society, to strive for the integral development of the youth and to bind together the youth of the SMC. SMYM has various programmes and activities for the holistic development of youth aligning its members with the religious and social causes of the Church including Campus programmes against substance abuse and formation of de-addiction clubs on campuses.

" Dear young people, let yourselves be taken over by the light of Christ and spread that light wherever you are. "

Form Name: Login Page



Form Name: Registration Page



4.4 DATABASE DESIGN

4.4.1 Relational Database Management System (RDBMS)

A relational database management system (RDBMS) is software specifically designed for the storage, organization, querying, and retrieval of data within a relational database. It acts as an intermediary between users and applications and the underlying database, offering essential administrative capabilities for overseeing data storage, access, and system performance.

4.4.2 Normalization

Normalization is the procedure of structuring data within a database. This involves the creation of tables and the establishment of relationships between those tables, all in accordance with predefined principles. The objective is twofold: to safeguard data integrity and enhance database flexibility by eliminating redundancy and inconsistent dependencies. The normalization process consists of three primary forms:

First Normal Form:

- Eliminate repeating groups in individual tables.
- Create separate tables for each set of related data.
- Identify each set of related data with a primary key.

Second Normal Form:

- Create separate tables for sets of values that apply to multiple records.
- Establish relationships between these tables using a foreign key.

Third Normal Form:

- Eliminate fields that do not depend on the key.

3.5 Normal Form (3.5NF):

- 3.5NF is a more advanced stage of normalization that builds upon the principles of Third Normal Form (3NF).
- In 3.5NF, the focus is on eliminating transitive dependencies between non-prime attributes (attributes that are not part of the primary key) in a relation.
- It ensures that there are no indirect relationships between non-prime attributes, which can lead to data anomalies.
- Achieving 3.5NF involves further breaking down tables and creating additional relationships to remove any non-prime attribute dependencies that are not directly related to the primary key.

This process of normalization ensures that data is structured efficiently, avoiding duplication and ensuring the database's stability and flexibility.

4.4.3 Sanitization

Validation involves assessing whether input data adheres to a defined set of criteria, ensuring it meets specific requirements, while sanitization is the process of modifying input to guarantee its validity. Combining these two techniques provides a comprehensive defense for your application. For instance, you may transform all single quotation marks into double quotation marks (sanitize) and subsequently validate that all quotation marks have been successfully changed to double quotation marks.

Validation checks encompass various aspects, including length, format, range, and allowable characters. For instance, in cases where an application anticipates positive integer input, validation is necessary to confirm that any string input exclusively contains the digits 0 through 9. These practices collectively bolster data quality and security within your application.

4.4.4 Indexing

Indexing is a method employed to enhance the performance of a database by reducing the number of disk accesses needed during query processing. It's a data structure technique designed for swift data retrieval in a database. Indexes are constructed using specific database columns, typically comprising two main components:

1. Search Key: This is the first column in the index and holds a copy of the primary key or candidate
key of the table. These values are organized in sorted order, facilitating rapid access to corresponding data. It's worth noting that the data may or may not be stored in sorted order.

2. Data Reference or Pointer: The second column contains a set of pointers that store the addresses of disk blocks where specific key values can be located.

Indexes are a critical part of database optimization, allowing for efficient data retrieval and query processing.

4.5 TABLE DESIGN

1.Tbl_users

Column Name	Data Type	Constraint
UserID	INT	Primary Key
FirstName	VARCHAR(50)	Not Null
MiddleName	VARCHAR(50)	
LastName	VARCHAR(50)	Not Null
HouseName	VARCHAR(100)	Not Null
PrayerGroup	VARCHAR(100)	Not Null
DateOfBirth	DATE	Not Null
Gender	CHAR(1)	Not Null
Email	VARCHAR(100)	Not Null, Unique
Password	VARCHAR(100)	Not Null
PhoneNumber	VARCHAR(20)	Not Null
ApprovalStatus	BOOLEAN	Not Null
RoleID	INT	Foreign Key (Roles.RoleID)

2. Table: Roles

Column Name	Data Type	Constraint
RoleID	INT	Primary Key
RoleName	VARCHAR(50)	Not Null

3. Table: BloodDonors

Column Name	Data Type	Constraint
DonorID	INT	Primary Key
Name	VARCHAR(100)	Not Null
Age	INT	Not Null
Gender	CHAR(1)	Not Null
Weight	DECIMAL(5,2)	Not Null

BloodGroup	VARCHAR(5)	Not Null
PhoneNumber	VARCHAR(20)	Not Null

4. Table: ParishDirectory

Column Name	Data Type	Constraint
DirectoryID	INT	Primary Key
Name	VARCHAR(100)	Not Null
HouseName	VARCHAR(100)	Not Null
PhoneNumber	VARCHAR(20)	Not Null

5. Table: Gallery

Column Name	Data Type	Constraint
ImageID	INT	Primary Key
ImageDescription	VARCHAR(500)	Not Null

6. Event

Column Name	Data Type	Constraint
EventID	INT	Primary Key
EventDate	DATE	Not Null
EventHeading	VARCHAR(200)	Not Null
EventDesc1	TEXT	Not Null
EventDesc2	TEXT	Not Null
Time	TIME	Not Null
Venue	VARCHAR(200)	Not Null

7. Table: FinanceTransactions

Column Name	Data Type	Constraint
TransactionID	INT	Primary Key
TransactionDescription	VARCHAR(200)	
Credit	DECIMAL(10,2)	

Debit	DECIMAL(10,2)	
Date	DATE	Not Null
ReceiptsAndVoucherImage	VARCHAR(200)	

8. Table: Quizzes

Column Name	Data Type	Constraint
QuizID	INT	Primary Key
QuizName	VARCHAR(100)	Not Null
QuizDescription	TEXT	

9. Table: CareerForumPosts

Column Name	Data Type	Constraint
PostID	INT	Primary Key
UserID	INT	Foreign Key (Users.UserID)
QuestionText	TEXT	Not Null
PostDate	DATE	

10. Table: CareerForumAnswers

Column Name	Data Type	Constraint
AnswerID	INT	Primary Key
PostID	INT	Foreign Key (CareerForumPosts.PostID)
UserID	INT	Foreign Key (Users.UserID)
AnswerText	TEXT	Not Null
AnswerDate	DATE	

11. Table: Donations

Column Name	Data Type	Constraint
DonationID	INT	Primary Key
UserID	INT	Foreign Key (Users.UserID), Nullable
GuestName	VARCHAR(100)	
DonationDate	DATE	Not Null
Amount	DECIMAL(10,2)	Not Null

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

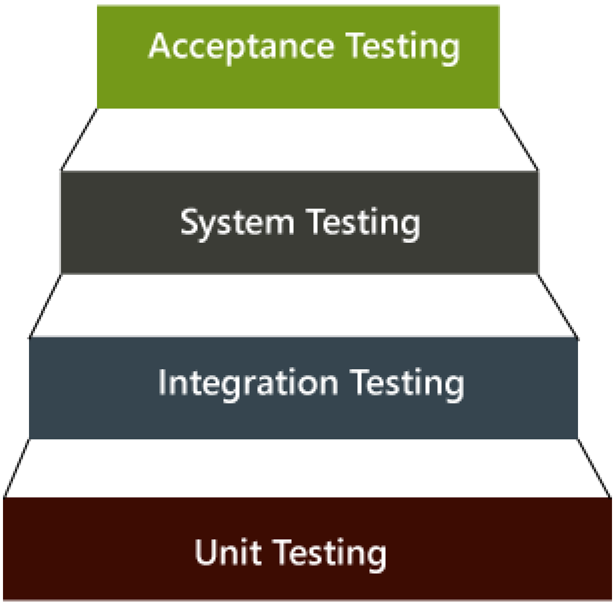
Explanation:

System Testing is a critical phase of software testing that assesses the complete and fully integrated software product. Its primary purpose is to validate the end-to-end system specifications. Typically, the software is just one component of a larger computer-based system, and it interfaces with other software and hardware systems. System Testing comprises a series of diverse tests aimed at thoroughly exercising the entire computer-based system.

In the realm of software testing, there are two main categories:

1. Black Box Testing: This approach focuses on testing the software without examining its internal code or workings. System Testing is classified under black box testing because it assesses the software's external behavior from the user's perspective.
2. White Box Testing: In contrast, white box testing delves into the internal workings and code of a software application.

System Testing, as a black box testing method, primarily evaluates the external functionality of the software, emphasizing how it behaves from the user's standpoint.



5.2 TEST PLAN

Explanation

The test plan for website cross-browser testing serves the following objectives:

- To specify the tools utilized during the testing process.
- To communicate the testing scope, schedule, and environmental requirements to the relevant stakeholders.
- To outline the methodology for conducting the tests.

Test Items:

This test plan outlines the objectives and scope of cross-browser testing for the wallpaper site project. It covers the testing of both the frontend customer-facing website and the back-end admin platform. The testing will be carried out on the latest stable versions of Chrome, Firefox, Safari, and Microsoft Edge, ensuring compatibility with major browsers. Furthermore, the testing will include both Windows and Mac machines to validate cross-platform functionality.

Features to Be Tested:

The testing scope encompasses the following key features:

1. Directing to the website's index page.
2. Navigating to the user login page
3. Logging in on the user login page.
4. Navigating to the website's index page.
5. Redirecting to user management page
6. Activating a suspended user
7. Logging out of the session.

5.2.1 Unit Testing

Explanation:

Unit Testing is a software testing technique focused on evaluating individual software components, which may include groups of program modules, usage procedures, and operating processes. The primary objective of this testing method is to ascertain the suitability of these components for their intended use. It's a process where each independent module is rigorously tested by the developer to identify and address any issues. Unit Testing primarily assesses the functional correctness of these independent modules.

Unit Testing is defined as a type of software testing in which individual components of a software product are examined. This testing takes place during the application development phase, and the individual component under assessment can be either an individual function or a procedure. Typically, Unit Testing is performed by developers themselves. It serves as the initial level of testing within the Software Development Life Cycle (SDLC) or V Model, conducted before integration testing. Although developers are the primary performers of Unit Testing, quality assurance engineers may also engage in this process as needed.

5.2.2 Integration Testing

Explanation

Integration Testing is a form of testing in which software modules are logically combined and tested collectively as a group. In a typical software project, various software modules are developed by different programmers. The primary goal of integration testing is to identify and reveal defects that may arise in the interactions between these software modules when they are integrated.

5.2.3 Validation Testing or System Testing

Explanation

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment. It answers to the question, are we building the right product?

5.2.4 Output Testing or User Acceptance Testing

Explanation

User Acceptance Testing (UAT) is a critical phase of software testing conducted by the end user or client. Its primary aim is to validate and ensure the readiness of the software system before deploying it to the production environment. UAT occurs in the final stages of testing, following functional, integration, and system testing. It serves as a crucial step in verifying that the software meets the user's requirements and expectations, thereby ensuring its suitability for deployment.

5.2.5 Automation Testing

Explanation

Automation testing is a vital process aimed at assessing software and other technological products to verify their adherence to stringent requirements. Essentially, it serves as a validation to ensure that the equipment or software functions precisely as intended. This form of testing diligently identifies and addresses bugs, defects, and potential issues that may arise during the product development phase. While some testing, such as manual regression or functional testing, can be executed manually, the advantages of automation testing are notable. Automation testing can be scheduled to run at any time, employs scripted sequences to thoroughly evaluate the software, and generates comprehensive reports on the findings. These reports can be compared with data from prior test runs to monitor changes and progress. Automation developers typically use programming languages such as C#, JavaScript, and Ruby to create and execute automated test scripts.

5.2.6 Selenium Testing

Explanation

Selenium is a crucial open-source tool for automated testing, highly valuable for web application developers. Selenium automation testing refers to the practice of conducting tests using Selenium. It's important to note that Selenium is not a singular tool; instead, it comprises a collection of tools, each serving distinct needs in Selenium automation testing.

Example:

Test Case 1

Code:

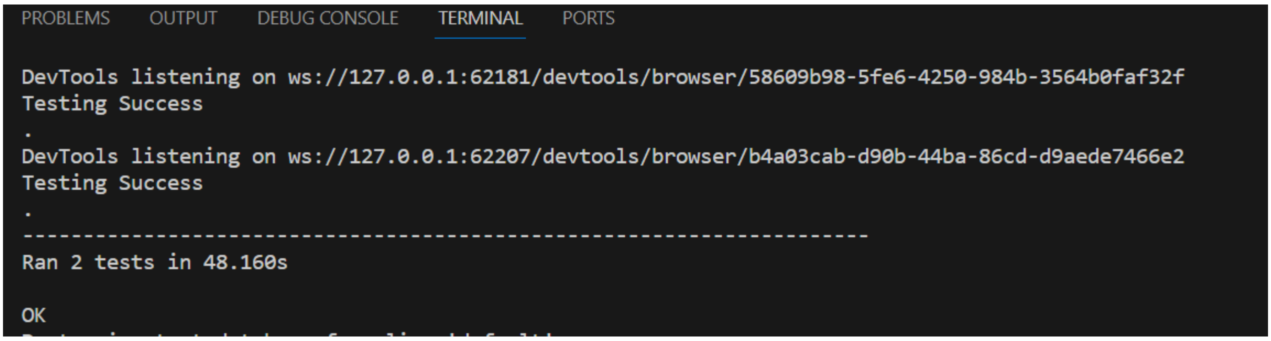
#register

```
from django.test import TestCase
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import Select
import time

class Hosttest(TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.implicitly_wait(10)
        self.live_server_url = 'http://127.0.0.1:8000/register'

    def tearDown(self):
        self.driver.quit()

    def test_django_project_functionalities(self):
        driver = self.driver
        driver.get(self.live_server_url)
        driver.maximize_window()
        time.sleep(1)

        # Fill out the registration form
        fname = driver.find_element(By.ID, "fname")
        fname.send_keys("John")
        mname = driver.find_element(By.ID, "Mname")
        mname.send_keys("Middle")
        lname = driver.find_element(By.ID, "lname")
        lname.send_keys("Doe")
        lname = driver.find_element(By.ID, "lname")
        lname.send_keys("Doe")
        house_name = driver.find_element(By.ID, "Hname")
        house_name.send_keys("Main Street")

        # Select the prayer group (e.g., 'Gethsemane')
        prayer_group_dropdown = Select(driver.find_element(By.ID, "prayer-group"))
        prayer_group_dropdown.select_by_value("Gethsemane")

        dob = driver.find_element(By.ID, "dob")
        dob.send_keys("21-10-2001")

        # Select the gender (e.g., 'Male')
        gender_dropdown = Select(driver.find_element(By.ID, "gender"))
        gender_dropdown.select_by_value("male")

        email = driver.find_element(By.ID, "email")
        email.send_keys("john123@example.com")

        password = driver.find_element(By.ID, "password")
        password.send_keys("@jhon1234")

        confirm_password = driver.find_element(By.ID, "cpassw")
        confirm_password.send_keys("@jhon1234")

        phone = driver.find_element(By.ID, "mob")
        phone.send_keys("8746353727")

        submit_button = driver.find_element(By.XPATH, "//button[contains(text(), 'Sign Up')]")
        submit_button.click()
        time.sleep(2)



        print("Testing Success")

if __name__ == '__main__':
    import unittest
    unittest.main()
```

#login, sort, logout

```
from django.test import TestCase
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import Select
import time

class Hosttest(TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.implicitly_wait(10)
        self.live_server_url = 'http://127.0.0.1:8000/login'

    def tearDown(self):
        self.driver.quit()

    def test_django_project_functionalities(self):
        driver = self.driver
        driver.get(self.live_server_url)
        driver.maximize_window()
        time.sleep(1)

        username = driver.find_element(By.CSS_SELECTOR, "input#username")
        username.send_keys("alanantony96696@gmail.com")
        password = driver.find_element(By.CSS_SELECTOR, "input#password")
        password.send_keys("@Lan4493")
        time.sleep(3)
        submit_button = driver.find_element(By.CSS_SELECTOR, "button#submit")
        submit_button.click()
        time.sleep(2)

        # Find the element with ID "services"
        services_link = driver.find_element(By.LINK_TEXT, "Services")
        services_link.click()
        time.sleep(4)

        # Navigate to the blood donor page
        blood_donors_link = driver.find_element(By.CSS_SELECTOR, "#blood")
        blood_donors_link.click()
        time.sleep(2)

        select_element = driver.find_element(By.ID, "filter-blood-group-select")
        select = Select(select_element)
        select.select_by_value("A+")
        # Logout from the session
        time.sleep(5)

        logout_button = driver.find_element(By.CSS_SELECTOR, "a#logout")
        logout_button.click()
        time.sleep(2)

        print("Testing Success")

if __name__ == '__main__':
    import unittest
    unittest.main()
```

Eg.Screenshot



```
DevTools listening on ws://127.0.0.1:62181/devtools/browser/58609b98-5fe6-4250-984b-3564b0faf32f
Testing Success
.
DevTools listening on ws://127.0.0.1:62207/devtools/browser/b4a03cab-d90b-44ba-86cd-d9aede7466e2
Testing Success
.
----------------------------------------------------------------------
Ran 2 tests in 48.160s

OK
```

Eg.Test Report

Test Case 1

Project Name: Adonai					
Login Test Case					
Test Case ID: Test_1			**Test Designed By:**		
Test Priority(Low/Medium/High): Medium			**Test Designed Date: 19-10-2023**		
Module Name: Registration			**Test Executed By: Ms. Nimmy Francis**		
Test Title : Adonai User Registration			**Test Execution Date: 19-10-2023**		
Description: Testing the registration module					
Pre-Condition : User want a valid email					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigate to registration page		User want to register successfully	User is successfully registered and navigate to login page	Pass
2	Provide First name	John			
3	Provide Middle name	Joy			
4	Provide Last name	Doe			
5	Provide House name	Main Street			

6	Select a prayer group	Gethsemane			
7	Provide Date of birth	21-10-2001			
8	Select Gender	Male			
9	Provide Email	john123@example.com			
10	Provide Password	@jhon1234			
11	Reenter the password	@jhon1234			
12	Provide Phone number	8746353727			
13	Click on submit button				
Post-Condition: User want to navigate to login page after registration					

<table>
<tr><td colspan="6"><b>Test Case 2</b></td></tr>
<tr><td colspan="6"><b>Project Name: Adonai</b></td></tr>
<tr><td colspan="6"><b>Login Test Case</b></td></tr>
<tr><td colspan="3"><b>Test Case ID: Test_1</b></td><td colspan="3"><b>Test Designed By:</b></td></tr>
<tr><td colspan="3"><b>Test Priority(Low/Medium/High): Medium</b></td><td colspan="3"><b>Test Designed Date: 19-10-2023</b></td></tr>
<tr><td colspan="3"><b>Module Name</b>: Login</td><td colspan="3"><b>Test Executed By: Ms. Nimmy Francis</b></td></tr>
<tr><td colspan="3"><b>Test Title :</b> User Login Test</td><td colspan="3"><b>Test Execution Date: 19-10-2023</b></td></tr>
<tr><td colspan="3"><b>Description:</b> Testing the user login module</td><td colspan="3"></td></tr>
<tr><td colspan="6"><b>Pre-Condition:</b> User is approved by the admin and have a valid email and password</td></tr>
<tr><td><b>Step</b></td><td><b>Test Step</b></td><td><b>Test Data</b></td><td><b>Expected Result</b></td><td><b>Actual Result</b></td><td><b>Status(Pass/ Fail)</b></td></tr>
</table>

1	Navigate to Login page		User want to login and navigate to index page	User navigated to index page with successful login	Pass
2	Provide Username	alanantony96696@gmail.com			
3	Provide Password	@Lan4493			
4	Click the login button				
Post-Condition: User is navigated to the index page					

Test Case 3

Project Name: Adonai					
Login Test Case					
Test Case ID: Test_1			**Test Designed By:**		
Test Priority(Low/Medium/High): Medium			**Test Designed Date: 19-10-2023**		
Module Name: Blood Donor List			**Test Executed By: Ms. Nimmy Francis**		
Test Title: Blood Donor Sorting			**Test Execution Date: 19-10-2023**		
Description: Testing the Blood donor list sorting					
Pre-Condition: Select a blood group to sort out the data					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigate to Blood Donors Page		Want to redirect to blood donors page	Redirected successfully to blood donors page	Pass
2	Select blood group	A+			
Post-Condition: Blood Donors with A+ blood group is shown					

CHAPTER 6

IMPLEMENTATION

6.1INTRODUCTION

Explanation

Implementation is a pivotal project phase where the theoretical design evolves into a functional system. It's a critical juncture in securing user confidence in the system's efficacy and accuracy. Key elements during this stage include user training and documentation. The conversion process typically aligns with user training or follows shortly thereafter. Implementation essentially means bringing a new system design into operational use, signifying the transition from design to reality. This phase places the most significant workload, disruption, and impact on the user department. If implementation lacks careful planning and control, it can lead to chaos and confusion. Implementation encompasses all activities necessary for transitioning from the existing system to the new one. The new system might be entirely fresh, replacing a manual or automated system, or it could be a modification of an existing setup. Effective implementation is paramount in delivering a dependable system that meets organizational requirements. System implementation denotes the process of putting the developed system into actual use. It only occurs after thorough testing, ensuring adherence to specifications. It involves evaluating system feasibility. The complexity of the system in question correlates with the depth of effort needed for successful implementation, covering three main aspects: education and training, system testing, and changeover.

The implementation phase encompasses the following key tasks:

- Thorough planning to ensure a smooth transition.
- In-depth investigation of the system and its constraints.
- Designing methods and strategies to achieve the changeover seamlessly.

6.2 IMPLEMENTATION PROCEDURES

Explanation

The implementation of software refers to the final installation of the package in its actual environment, ensuring it meets the needs of its intended users and operates as expected. In many organizations, software development projects are commissioned by individuals who will not be the end users. In the initial stages, there may be doubts about the software. To ensure a smooth transition and minimize resistance, it's essential to:

- Ensure active users are aware of the benefits of the new system.
- Build confidence in the software among users.
- Provide proper guidance to users, making them comfortable with the application.

- Inform users that the server program must be running for them to access the system's results. Without the server object being active, the intended process won't occur.

6.2.1 User Training

Explanation

User training plays a crucial role in preparing users for system testing and the transition to a new system. It is essential for individuals involved in the computer-based system to have confidence in their roles to achieve the expected benefits. As systems become more intricate, the importance of training grows. Through user training, individuals learn how to input data, handle error messages, query databases, execute routines for generating reports, and perform other essential functions within the system.

6.2.2 Training on the Application Software

Explanation

Following the essential basic computer awareness training, the user's next step involves training on the new application software. This training aims to instill the fundamental principles of using the new system, including screen navigation, screen design, on-screen assistance, handling data entry errors, validation checks, and methods for rectifying data entry mistakes. The training then delves into user-specific information, tailoring the program to meet the unique needs of each user or user group. It's important to note that this training may vary among different user groups and across various hierarchical levels.

6.2.3 System Maintenance

Explanation

Maintenance is a critical aspect of the system development process. During the maintenance phase of the software development cycle, a software product is actively utilized and serves its intended purpose. It is crucial to ensure that a system, once successfully implemented, receives proper maintenance. System maintenance is a fundamental component of the software development life cycle, primarily aimed at making the system adaptable to changes in its environment. It's important to note that software maintenance goes beyond mere error identification and correction; it encompasses a broader spectrum of activities necessary for the continued effectiveness and relevance of the software product.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The "Adonai" project has been successfully executed, ushering in a transformative platform that revolutionizes the management and user engagement for the SMYM Mukkoottuthara organization. This informative website caters to different user roles, enhancing efficiency and expanding the organization's online presence. Administrators are equipped with tools for user management, blood donors list and parish directory maintenance, gallery management, event report management, quiz and poll creation, and donation management. Normal users benefit from the ability to create accounts, access information, participate in quizzes and polls, view detailed event reports, and make donations. Guest users can easily access organization information and contribute through donations.

The project's user-centric approach, combined with modern technologies, enables a seamless and efficient experience for both administrators and users. With features like a secure payment gateway, career guidance forums, and an interactive platform, "Adonai" not only streamlines management but also offers valuable resources and engagement opportunities. This successful project reflects a significant step toward strengthening the organization's online presence and providing a digital hub that fosters efficient management and meaningful user engagement.

7.2 FUTURE SCOPE

The future scope for the "Adonai" project holds exciting potential for enhancing the user experience and expanding the platform's feature set. These opportunities encompass several key areas, mirroring the dynamic evolution of the platform:

1. Enhanced Data Searching and Personalization: Adonai can implement advanced filtering and sorting options to enable users to discover information with precision. Personalized search results aligned with individual preferences can contribute to a more engaging and individualized user experience.

2. Machine Learning Integration: Leveraging machine learning algorithms can enable the platform to offer personalized recommendations to users based on their unique preferences and past interactions. This may involve collaborative filtering and providing suggestions based on a user's previous engagements.

3. Dynamic Community Space: Adonai can create a vibrant community space where users and content creators interact, share ideas, and engage with each other. Features like forums, comments, and direct messaging can facilitate these interactions, fostering a sense of community.

4. User Following and Updates: Allowing users to follow their favorite content creators and receive updates on their latest contributions can add an extra layer of engagement and connection.

5. User Feedback Mechanisms: The incorporation of a robust feedback system can enable users to provide comments, ratigs, and reviews for various aspects of the platform. This feedback is invaluable for refining the platform and highlighting popular content.

6. Voice Search Integration: Implementing a voice search assistant can empower users to search for information using voice commands, enhancing accessibility and convenience. Natural language processing can further refine this feature, ensuring precise and user-friendly voice-based searches.

The future scope for Adonai aims to enrich user interactions, provide tailored experiences, and remain at the forefront of technological trends in the continually evolving realm of online engagement and information management. The implementation of the main project abstract, which includes advanced features and functionalities, will further elevate Adonai's capabilities and user engagement.

CHAPTER 8
BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009.
- Roger S Pressman, “Software Engineering”, 1994.
- PankajJalote, “Software engineering: a precise approach”, 2006.
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- https://www.geeksforgeeks.org/
- https://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
- http://agilemodeling.com/artifacts/useCaseDiagram.html

CHAPTER 9

APPENDIX

9.1 Sample Code

Registration for new members:

```
from django.http import JsonResponse

from django.shortcuts import get_object_or_404, render

from django.shortcuts import render, redirect

from django.contrib.auth.models import User

from django.urls import reverse


def register(request):

    if request.method == 'POST':

        email = request.POST['email']

        password = request.POST['password']

        first_name = request.POST['fname']

        last_name = request.POST['lname']

        Mname = request.POST['Mname']

        Hname = request.POST['Hname']

        prayerGroup = request.POST['prayerGroup']

        dob = request.POST['dob']

        gender = request.POST['gender']

        mob = request.POST['mob']


        # Check if a user with the same email already exists

        if User.objects.filter(email=email).exists():

            messages.error(request, 'Email already exists. Please use a different email.')

            return render(request, 'register.html')


        user = User.objects.create_user(username=email, email=email, first_name=first_name,
last_name=last_name, password=password)


        registration = Registration(

            user=user,

            middle_name=Mname,

            house_name=Hname,

            prayer_group=prayerGroup,

            date_of_birth=dob,
            gender=gender,
            phone_number=mob,
        )
        registration.fname = first_name
        registration.lname = last_name
        registration.save()
        return redirect('login')
    return render(request, 'register.html')
```

Login Page:

```
from django.contrib.auth import login, authenticate
from django.contrib import messages
from django.shortcuts import render, redirect
def login_view(request):
    if request.method == 'POST':
        email = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=email, password=password)
        if user is not None:
            if user.is_superuser:
                login(request, user)
                return redirect('index')
            elif user.registration.is_active and user.registration.is_approved:
                login(request, user)
                return redirect('index')
            elif not user.registration.is_active:
                messages.error(request, 'User account is not active.')
            else:
                messages.error(request, 'User is not approved yet. Please wait for approval.')
        else:
            messages.error(request, 'Invalid username or password. Please try again.')

    return render(request, 'login.html')
```

Home page:

```
def index(request):
    return render(request, "index.html")
```

Member Approval/ Deactivate from Admin dashboard:

Approve user

```
from django.core.mail import send_mail
from django.conf import settings

def approve_user(request, user_id):
    if request.user.is_superuser:
        user_profile = get_object_or_404(Registration, user__id=user_id)

        user_profile.is_approved = True
        user_profile.is_active = True  # Set the user as active
        user_profile.save()


        subject = 'Account Approval for SMYM Mukkoottuthara'
        # message = f'Dear{user_profile.user.first_name}, ,We are thrilled to inform you that your registration request for Adonai has been approved by our admin team, and your account is now ready for use.You can now access all the exciting features and content available on our platform by logging in with your registered email address and password.We are committed to providing you with a seamless and enjoyable experience on Adonai, and we look forward to having you as an active member of our community.If you have any questions, encounter any issues, or need assistance with anything related to our website, please do not hesitate to contact our support team at smymmukkoottuthara@gmail.com.Thank you for choosing Adonai. We hope you have a fantastic time exploring our platform.Best regards,President SMYM Mukkoottuthara Adonai{ user_profile.user.email }'
        message = f'Dear {user_profile.user.username},\n\n' \
          'We are thrilled to inform you that your registration request for Adonai has been approved by our admin team, and your account is now ready for use.\n\n' \
          'You can now access all the exciting features and content available on our platform by logging in with your registered email address and password.\n\n' \
          'To get started, please click the following link to log in:\n' \
          '"http://127.0.0.1:8000/login" : Log In to Adonai\n\n' \
          'We are committed to providing you with a seamless and enjoyable experience on Adonai, and we look forward to having you as an active member of our community.\n\n' \
          'If you have any questions, encounter any issues, or need assistance with anything related to our website, please do not hesitate to contact our support team at smymmukkoottuthara@gmail.com.\n\n' \
          'Thank you for choosing Adonai. We hope you have a fantastic time exploring our platform.\n\n' \
          'Best regards,\n' \
          'President \n'\
        'SMYM Mukkoottuthara \n'
        'Adonai'
        from_email = settings.EMAIL_HOST_USER
        recipient_list = [ user_profile.user.email ]

        send_mail(subject, message, from_email, recipient_list)

        return redirect('user_admin')
    else:
        # If the current user is not a superuser, redirect them to the login page
        return redirect('login')
```

Delete User

```
from django.core.mail import send_mail
from django.conf import settings
from django.shortcuts import render, redirect, get_object_or_404
from django.contrib.auth.models import User
from .models import Registration

def delete_user(request, user_id):
    user_profile = get_object_or_404(Registration, user__id=user_id)
    if request.method == 'POST':
        comment = request.POST.get('comments', '')
        user_profile.is_active = False
        user_profile.comments = comment
        user_profile.save()

         # Compose the email body
        subject =  'Account Suspension Notification'
        message = f'Dear {user_profile.user.first_name},\n' \
                  'We hope this message finds you well. We regret to inform you that your account on [Website Name] has been suspended temporarily due to the following reason:\n\n' \
                  f'Suspension Reason: {user_profile.comments}\n' \
                  f'Suspension Date: {timezone.now()}\n\n' \
                  'Your account will remain suspended until further notice. During this time, you will not be able to access your account or use the platform\'s features.\n\n' \
                  'If you believe this suspension is in error or have any questions regarding the suspension, please reach out to our support team at smymmukkoottuthara@gmail.com for assistance. We will do our best to address your concerns and provide clarification on the situation.\n\n' \
                  'We take account suspensions seriously and strive to maintain a safe and enjoyable environment for all users. We appreciate your understanding and cooperation in this matter.\n\n' \
                  'Thank you for your attention to this notification, and we hope to resolve this issue promptly.\n\n' \
                  'Best regards,\n' \
                  'President\n' \
                  'SMYM Mukkoottuthara\n' \
                  'smymmukkoottuthara@gmail.com'

        # Send the email
        from_email = settings.EMAIL_HOST_USER
        recipient_list = [user_profile.user.email]

        send_mail(subject, message, from_email, recipient_list)

        return redirect('user_admin')

    return render(request, 'user_admin.html', {'user_profile': user_profile})
```

Dashboard

```
from django.shortcuts import render, redirect
from django.contrib.auth.decorators import login_required
from django.contrib.auth.models import User
from .models import Registration

@login_required
def user_admin(request):
    if request.user.is_superuser:
        # Get all user profiles
        user_profiles = Registration.objects.select_related('user').all()

        # Separate profiles into three lists based on approval status
        pending_approval_users = [profile for profile in user_profiles if not profile.is_approved and
profile.is_active]
        approved_users = [profile for profile in user_profiles if profile.is_approved and profile.is_active]
        rejected_users = [profile for profile in user_profiles if not profile.is_active]

        context = {
            'pending_approval_users': pending_approval_users,
            'approved_users': approved_users,
            'rejected_users': rejected_users,
        }
        return render(request, 'user_admin.html', context)
    else:
        return redirect('login')
```

Blood Donor Form

```
from django.contrib.admin.views.decorators import staff_member_required
from django.shortcuts import render, redirect
from .models import Donor

def blood_admin(request):
    donors = Donor.objects.filter(is_deleted=False)
    is_superuser = request.user.is_superuser if request.user.is_authenticated else False

    if request.method == 'POST' and is_superuser:
        name = request.POST.get('funame')
        age = request.POST.get('age')
        gender = request.POST.get('gender')
        blood_group = request.POST.get('blood-group')
        contact = request.POST.get('mob')

        donor = Donor(
            name=name,
            age=age,
            gender=gender,
            blood_group=blood_group,
            contact=contact
        )
        donor.save()
        # messages.success(request, 'Donor data submitted successfully.')
        return redirect('blood_admin')

    return render(request, 'blood_admin.html', {'donors': donors, 'is_superuser': is_superuser})

from django.shortcuts import get_object_or_404, redirect, reverse  # Import the reverse function
from .models import Donor

def soft_delete_donor(request, donor_id):
    donor = get_object_or_404(Donor, pk=donor_id)
    donor.is_deleted = True
    donor.save()

    # Use reverse to get the URL for the blood_admin page
    return redirect(reverse('blood_admin'))
```

Parish Directory Page

```
from django.urls import reverse
from django.http import HttpResponseRedirect
from django.shortcuts import render, redirect, get_object_or_404
from django.db import IntegrityError
from .models import PrayerGroup, ParishDirectory
from django.db.models import Q

def parish_admin(request):
    error_message = None

    if request.method == 'POST':
        if 'new_group_name' in request.POST:
            new_group_name = request.POST.get('new_group_name')
            if new_group_name:
                # Check if the group name already exists (case-insensitive)
                if PrayerGroup.objects.filter(Q(name__iexact=new_group_name) &
~Q(is_deleted=True)).exists():
                    error_message = "A prayer group with this name already exists."
                else:
                    try:
                        PrayerGroup.objects.create(name=new_group_name)
                    except IntegrityError:
                        error_message = "An error occurred while creating the prayer group."
        else:
            form_name = request.POST.get('funame')
            form_house_name = request.POST.get('Hname')
            form_contact = request.POST.get('mob')
            prayer_group_id = request.POST.get('prayer_group')

            if form_name and form_house_name and form_contact and prayer_group_id:
                prayer_group = PrayerGroup.objects.get(pk=prayer_group_id)
                ParishDirectory.objects.create(
                    name=form_name,
                    house_name=form_house_name,
                    contact=form_contact,
                prayer_group=prayer_group
            )

    elif request.method == 'GET':
        error_message = None  # Initialize the error message
        if 'soft_delete_group' in request.GET:
            group_id = request.GET.get('soft_delete_group')
            try:
                group = PrayerGroup.objects.get(pk=group_id)
                group.is_deleted = True
                group.save()

                # Soft delete all related Parish Members
                ParishDirectory.objects.filter(prayer_group=group).update(is_deleted=True)

            except PrayerGroup.DoesNotExist:
                error_message = "The selected prayer group does not exist."

            # Redirect back to the parish_admin page after group deletion
            return HttpResponseRedirect(reverse('parish_admin'))

        elif 'soft_delete_member' in request.GET:
            member_id = request.GET.get('soft_delete_member')
            try:
                member = ParishDirectory.objects.get(pk=member_id)
                member.is_deleted = True
                member.save()

            except ParishDirectory.DoesNotExist:
                error_message = "The selected parish member does not exist."

            # Redirect back to the parish_admin page after member deletion
            return HttpResponseRedirect(reverse('parish_admin'))

    prayer_groups = PrayerGroup.objects.filter(is_deleted=False)
    parish_members = ParishDirectory.objects.select_related('prayer_group').filter(is_deleted=False)
    deleted_prayer_groups = PrayerGroup.objects.filter(is_deleted=True)
    deleted_parish_members = ParishDirectory.objects.filter(is_deleted=True)

    return render(request, 'parish_admin.html', {
        'prayer_groups': prayer_groups,
        'parish_members': parish_members,
        'deleted_prayer_groups': deleted_prayer_groups,
        'deleted_parish_members': deleted_parish_members,
        'error_message': error_message,
    })

from django.shortcuts import redirect
from django.contrib import messages

def retrieve_deleted_entity(request, entity_type, entity_id):
    retrieve_error_message = None  # Initialize the error message

    try:
        if entity_type == 'prayer_group':
            group = PrayerGroup.objects.get(pk=entity_id)
            group.is_deleted = False
            group.save()

            # Set the is_deleted flag to False for related Parish Members
            ParishDirectory.objects.filter(prayer_group=group).update(is_deleted=False)

        elif entity_type == 'parish_member':
            member = ParishDirectory.objects.get(pk=entity_id)

            # Check if the related prayer group is deleted
            if member.prayer_group.is_deleted:
                retrieve_error_message = "The chosen participant's prayer group is not active or dosen't exist"
            else:
                member.is_deleted = False
                member.save()
        else:
            raise ValueError("Invalid entity type.")

    except (PrayerGroup.DoesNotExist, ParishDirectory.DoesNotExist, ValueError) as e:
        retrieve_error_message = str(e)  # Convert the exception to a string

    # Store the error message in the Django messages framework
    if retrieve_error_message:
        messages.error(request, retrieve_error_message)

    # Redirect back to the 'parish_admin' page
    return redirect('parish_admin')
```

Career Guidance Page

```
from django.shortcuts import render, redirect, get_object_or_404
from django.contrib.auth.decorators import login_required
from django.contrib import messages

@login_required
def career_forum(request):
    return render(request, 'career_forum.html')

from django.shortcuts import render, redirect
from django.http import JsonResponse
from .models import Question, Answer
from django.contrib.auth.decorators import login_required

@login_required
def post_question(request):
    if request.method == 'POST':
        title = request.POST.get('questionTitle')
        description = request.POST.get('questionDescription')
        additional_details = request.POST.get('additionalDetails')
        user = request.user

        question = Question.objects.create(
            title=title,
            description=description,
            additional_details=additional_details,
            posted_by=user
        )

        # Optionally, you can redirect to the career_forum page after posting
        return redirect('career_forum')
    else:
        return JsonResponse({'success': False})
```

Post an answer

```
from django.shortcuts import render, redirect
from django.http import JsonResponse
from .models import Question, Answer
from django.contrib.auth.decorators import login_required

@login_required
def post_answer(request):
    if request.method == 'POST':
        # print(answer_text)
        answer_text = request.POST.get('answerText')
        question_id = request.POST.get('questionId')
        user = request.user
        # print('answer_text')
        question = Question.objects.get(pk=question_id)

        answer = Answer.objects.create(
            question=question,
            answer_text=answer_text,
            posted_by=user
        )

        # Optionally, you can redirect to the career_forum page after posting
        return redirect('career_forum')
    else:
        return JsonResponse({'success': False})



from django.shortcuts import render
from .models import Question, Answer
from .models import CareerResourcePerson

def career_forum(request):
    # Fetch and prepare the list of questions and answers from your database
    # questions = Question.objects.all()
    # answers = Answer.objects.filter(is_deleted=False)
    questions = Question.objects.filter(is_deleted=False).select_related('posted_by__registration').all()
    answers = Answer.objects.filter(is_deleted=False).select_related('posted_by__registration')
    resource_persons = CareerResourcePerson.objects.all()
    context = {
        'questions': questions,
        'answers': answers,
        'resource_persons': resource_persons,
    }
    return render(request, 'career_forum.html', context)
```

Delete answer

```
from django.shortcuts import get_object_or_404, redirect
from django.http import JsonResponse
from .models import Question, Answer
from django.http import JsonResponse

def soft_delete_answer(request, answer_id):
    try:
        answer = Answer.objects.get(pk=answer_id)
        answer.is_deleted = True
        answer.save()
        return redirect('career_forum')
    except Answer.DoesNotExist:
        return JsonResponse({'success': False, 'error_message': 'Answer not found'})
```

Delete Question

```
from django.http import JsonResponse

def soft_delete_question(request, question_id):
    try:
        question = Question.objects.get(pk=question_id)
        question.is_deleted = True
        question.save()
        return redirect('career_forum')  # Replace 'career_forum' with the URL name of your forum page
    except Question.DoesNotExist:
        return JsonResponse({'success': False, 'error_message': 'Question not found'})
```

Edit Comment

```
from django.http import JsonResponse

def edit_comment(request, answer_id):
    if request.method == 'POST':
        edited_text = request.POST.get('edited_text')

        try:
            # Find the comment by its ID and update the text
            comment = Answer.objects.get(pk=answer_id)
            comment.answer_text = edited_text
            comment.save()

            return JsonResponse({'success': True})
        except Answer.DoesNotExist:
            return JsonResponse({'success': False, 'error': 'Comment not found'})
    else:
        return JsonResponse({'success': False, 'error': 'Invalid request method'})
```

Edit Question

```
from django.shortcuts import get_object_or_404
from django.http import JsonResponse

from .models import Question

def edit_question(request, question_id):
    # Assuming you have a Question model with fields 'title' and 'description'
    question = get_object_or_404(Question, pk=question_id)

    if request.method == 'POST':
        edited_title = request.POST.get('edited_title')
        edited_description = request.POST.get('edited_description')

        # Update the question with the new data
        question.title = edited_title
        question.description = edited_description
        question.save()

        # Return a JSON response indicating success
        return JsonResponse({'success': True})
        print(success)

    # Return a JSON response indicating failure (if the request method is not POST)
    return JsonResponse({'success': False})
```

Report Comment

```
from django.shortcuts import get_object_or_404
from django.http import JsonResponse

def report_comment(request, answer_id):
    answer = get_object_or_404(Answer, id=answer_id)
    reason = request.POST.get('reason', '')

    if reason:
        # Create an AnswerReport object and save it to the database
        report = AnswerReport(answer=answer, reporter=request.user, reason=reason)
        report.save()
        return JsonResponse({'success': True})
    else:
        return JsonResponse({'success': False})

from django.shortcuts import render
from django.db.models import Count, F
from .models import AnswerReport, Answer

def reported_comments(request):
    # Annotate each answer with the count of reports
    reported_answers =
Answer.objects.annotate(report_count=Count('answerreport')).filter(report_count__gt=0)

    return render(request, 'reported_comments.html', {'reported_answers': reported_answers})
```

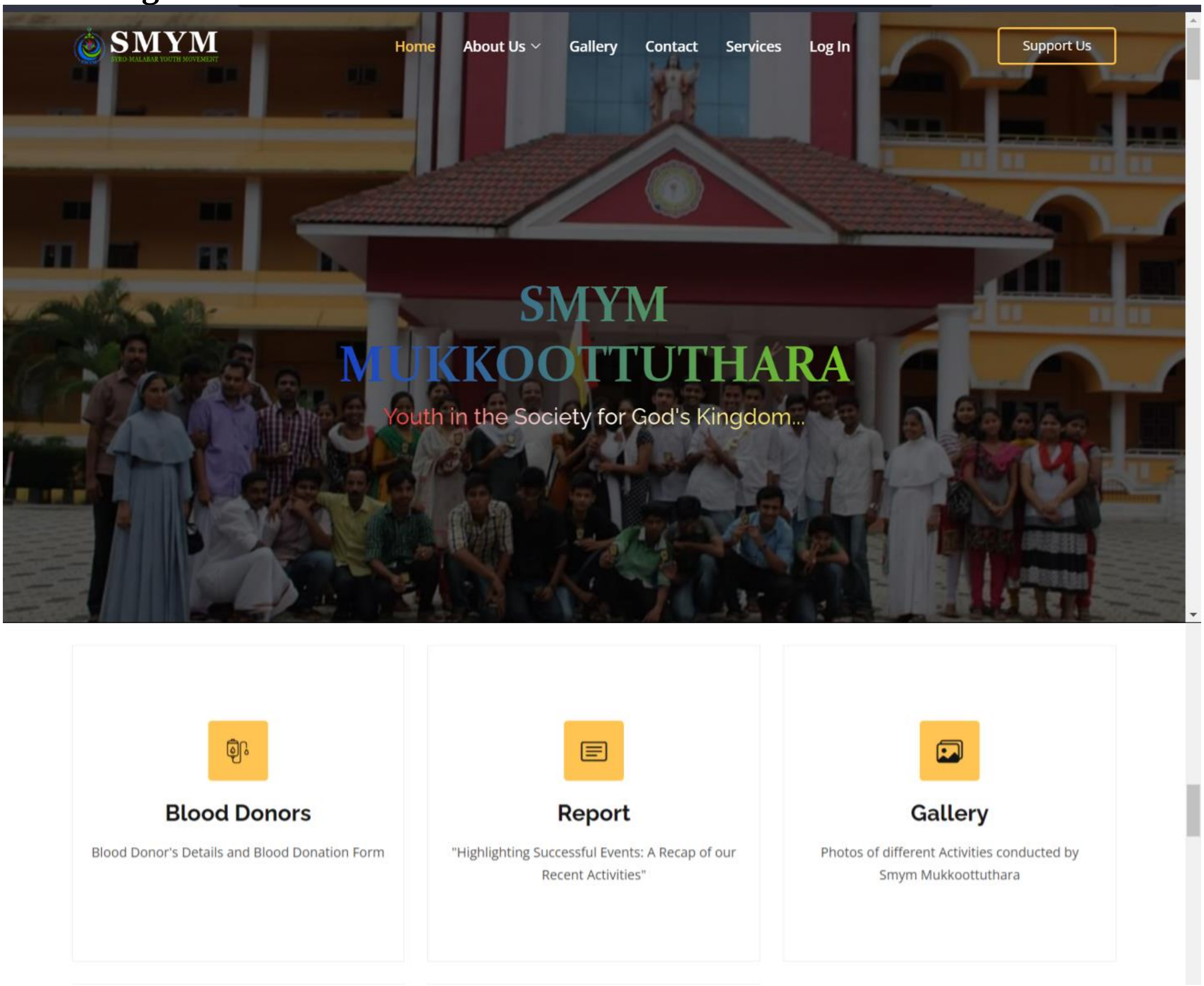
Delete Reported Answer

```
from django.shortcuts import get_object_or_404, redirect
from django.http import JsonResponse
from .models import Question, Answer, AnswerReport
from django.http import JsonResponse
from django.core.mail import send_mail
from django.contrib.auth.decorators import login_required

@login_required  # Ensure the view is accessible only to logged-in users
def soft_delete_reported_answer(request, answer_id):
    try:
        answer = Answer.objects.get(pk=answer_id)

        # Check if the answer is reported
        if AnswerReport.objects.filter(answer=answer).exists():
            # Soft delete the answer
            answer.is_deleted = True
            answer.save()



            removal_reasons = "\n\n".join(report.reason for report in
AnswerReport.objects.filter(answer=answer))

            # Send an email notification to the answer's author
            subject = 'Answer Deleted due to reports'
            message = f'Dear {answer.posted_by.username},\n\n' \
                'We hope this message finds you well. We would like to inform you that one of your recent
answers on [Platform/Website Name] has been removed due to reports from other users. We take the
quality and appropriateness of content on our platform seriously, and this action has been taken to ensure a
safe and respectful environment for all users.\n\n' \
                'Removed Answer Details:\n\n' \
                f'- Question: {answer.question.title}\n' \
                f'- Date Posted: {answer.posted_date_time}\n\n' \
                f'Reason(s) for Removal:\n\n{removal_reasons}\n\n' \
                'We encourage our users to follow our community guidelines and terms of service to maintain a
positive and constructive atmosphere on our platform. If you have any questions or concerns regarding the
removal of your answer or would like further clarification, please do not hesitate to reach out to our support
team at smymmukkoottuthara@gmail.com\n\n' \
                'Your contributions to our community are valued, and we appreciate your understanding of this
situation. We look forward to your continued participation and the sharing of valuable insights on
Adonai.\n\n' \
                'Thank you for being a part of our community.\n\n' \
                'Best regards,\n' \
                'President, Smym Mukkoottuthara\n' \
                'Admin, Adonai\n'

            from_email = settings.EMAIL_HOST_USER
            recipient_list = [answer.posted_by.email]

            send_mail(subject, message, from_email, recipient_list)

            return redirect('career_forum')  # Redirect to the appropriate page
        else:
```

```
        return JsonResponse({'success': False, 'error_message': 'Answer is not reported'})
    except Answer.DoesNotExist:
        return JsonResponse({'success': False, 'error_message': 'Answer not found'})
```

9.1 Screen Shots

Home Page



Blood Donors

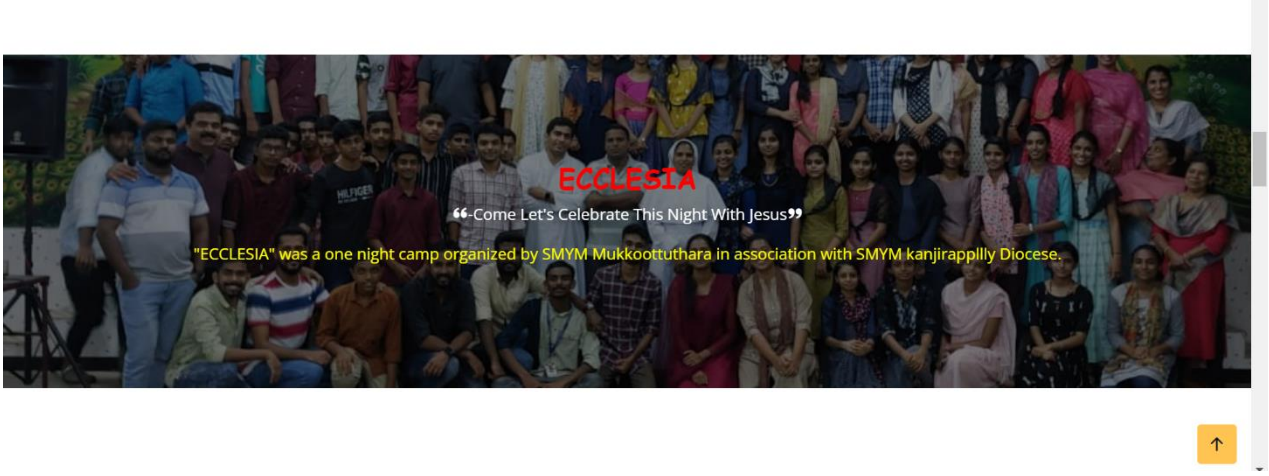
Blood Donor's Details and Blood Donation Form

Report

"Highlighting Successful Events: A Recap of our Recent Activities"

Gallery

Photos of different Activities conducted by Smym Mukkoottuthara

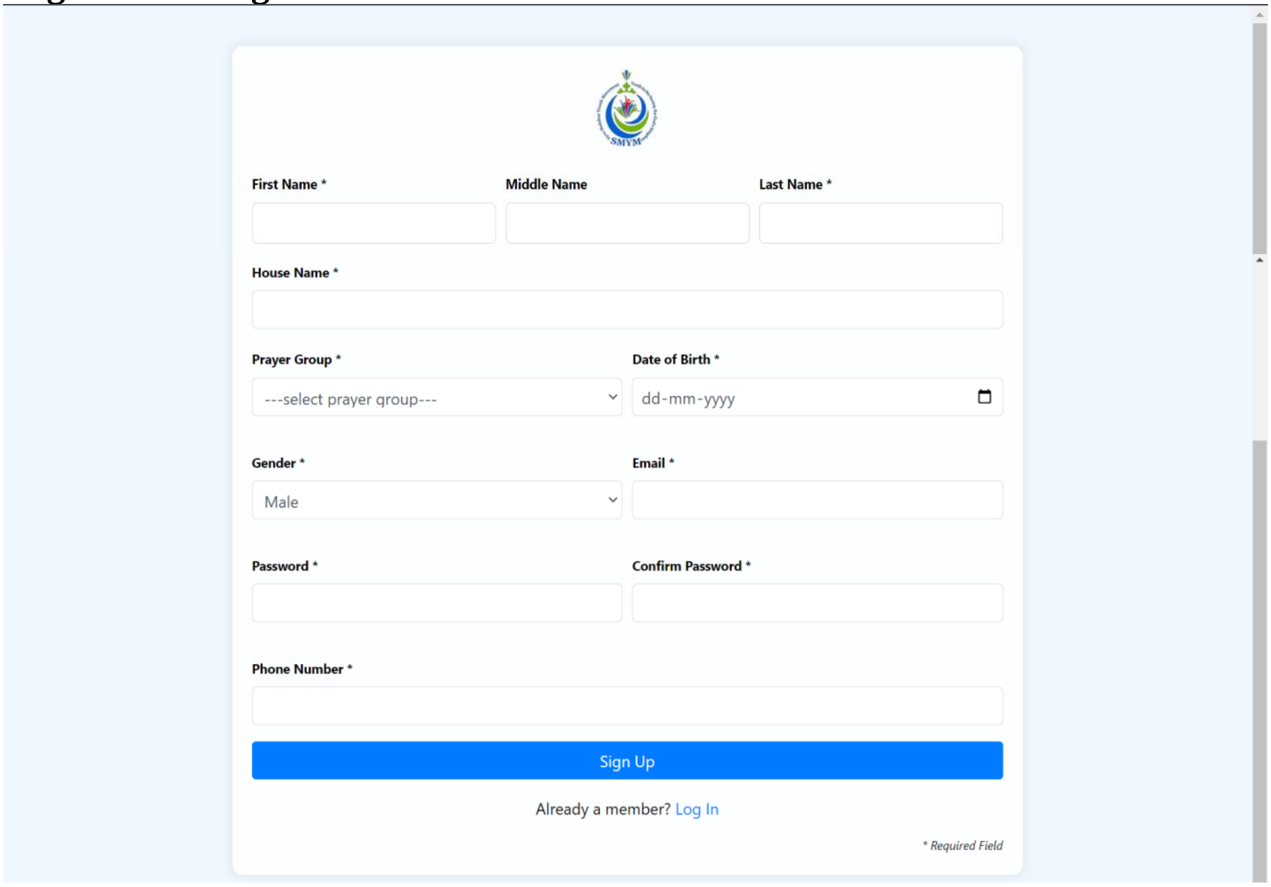


ECCLESIA

❝-Come Let's Celebrate This Night With Jesus❞

"ECCLESIA" was a one night camp organized by SMYM Mukkoottuthara in association with SMYM kanjirappllly Diocese.

Registration Page:



First Name *

Middle Name

Last Name *

House Name *

Prayer Group *

---select prayer group---

Date of Birth *

dd-mm-yyyy

Gender *

Male

Email *

Password *

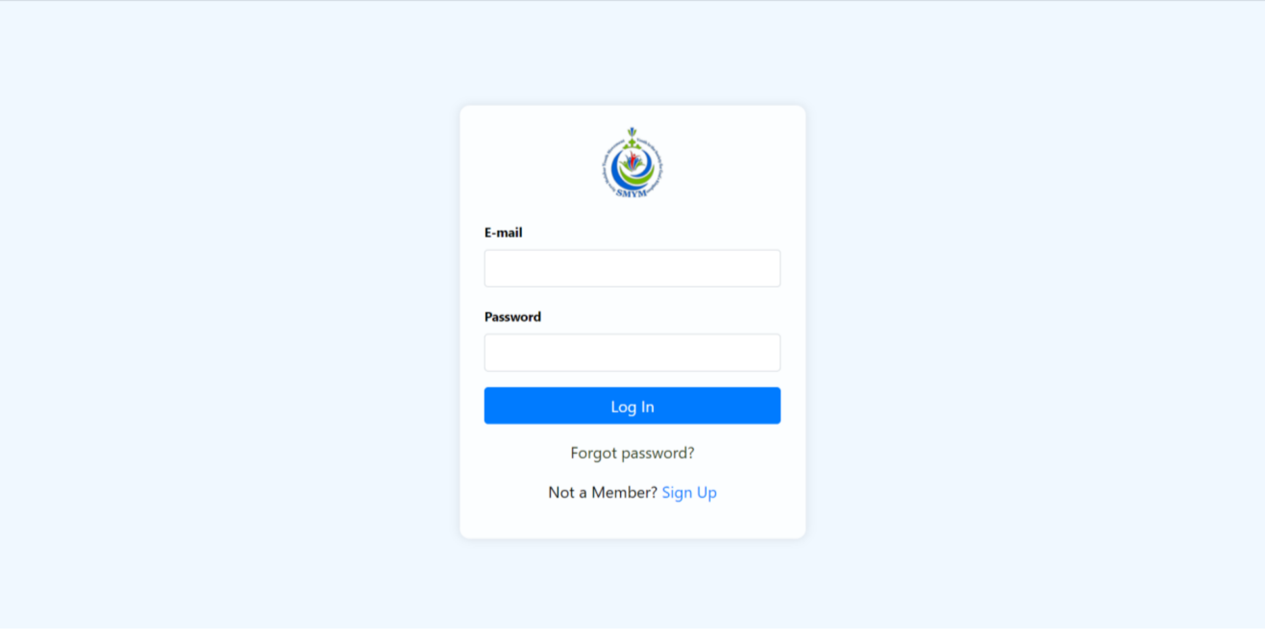
Confirm Password *

Phone Number *

Already a member? Log In

** Required Field*

Login Page



E-mail

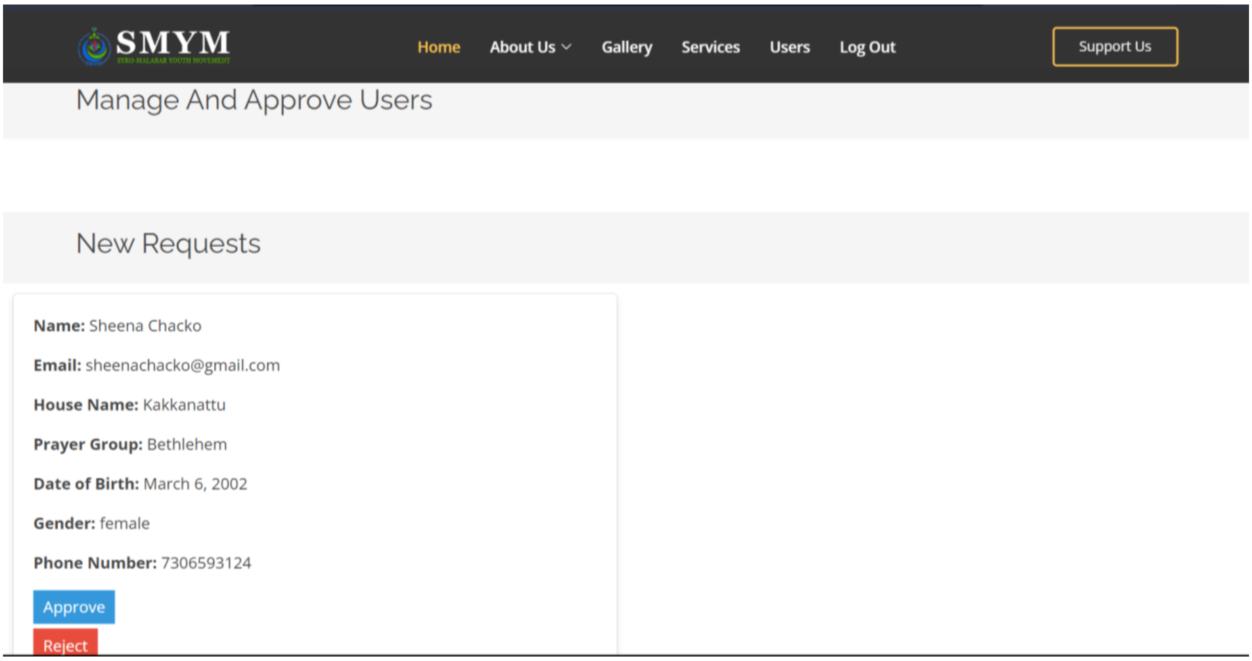
Password

Log In

Forgot password?

Not a Member? Sign Up

User Approval/Rejection Page



Manage And Approve Users

New Requests

Name: Sheena Chacko

Email: sheenachacko@gmail.com

House Name: Kakkanattu

Prayer Group: Bethlehem

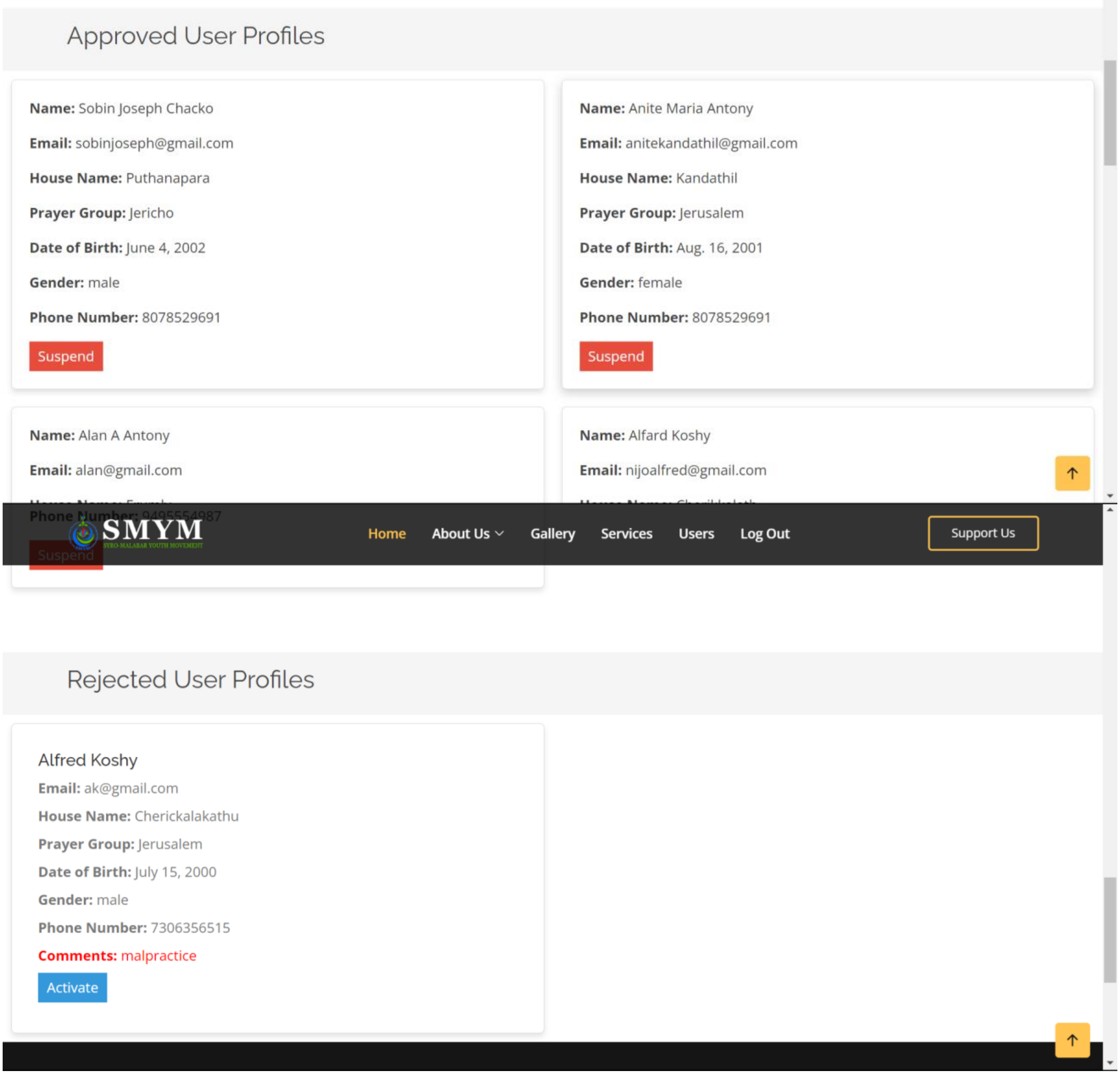
Date of Birth: March 6, 2002

Gender: female

Phone Number: 7306593124

Approve

Reject



Approved User Profiles

Name: Sobin Joseph Chacko

Email: sobinjoseph@gmail.com

House Name: Puthanapara

Prayer Group: Jericho

Date of Birth: June 4, 2002

Gender: male

Phone Number: 8078529691

Name: Anite Maria Antony

Email: anitekandathil@gmail.com

House Name: Kandathil

Prayer Group: Jerusalem

Date of Birth: Aug. 16, 2001

Gender: female

Phone Number: 8078529691

Name: Alan A Antony

Email: alan@gmail.com

Phone Number: 9495554987

Name: Alfard Koshy

Email: nijoalfred@gmail.com

Rejected User Profiles

Alfred Koshy

Email: ak@gmail.com

House Name: Cherickalakathu

Prayer Group: Jerusalem

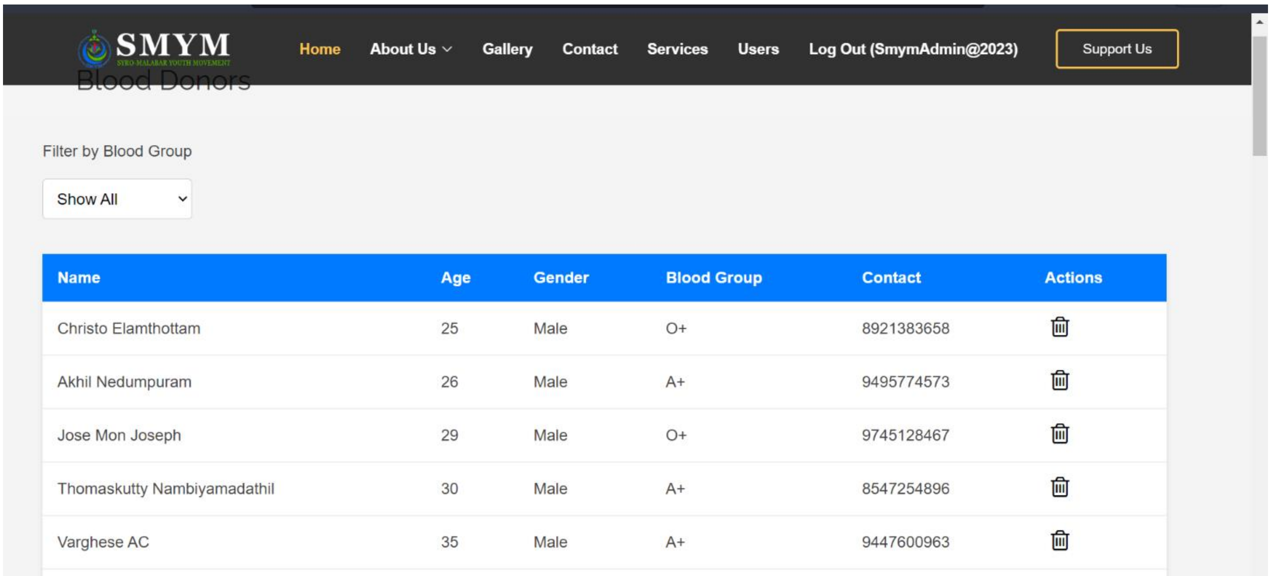
Date of Birth: July 15, 2000

Gender: male

Phone Number: 7306356515

Comments: malpractice

Blood Donors List

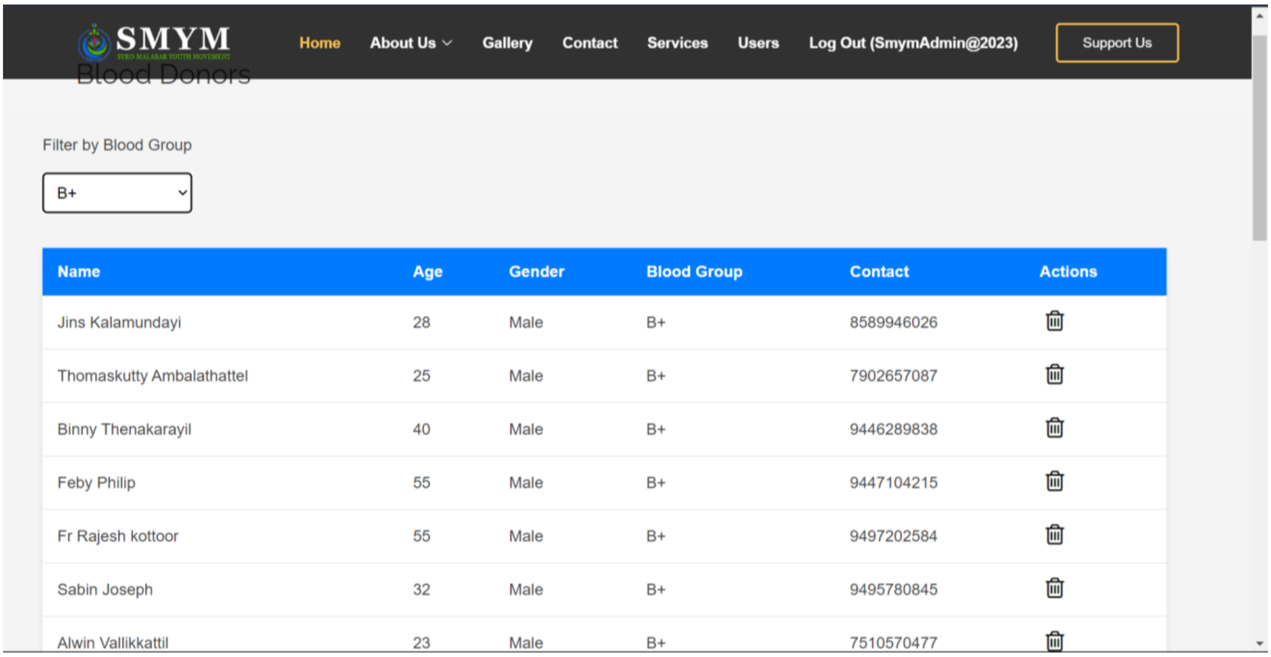


Blood Donors

Filter by Blood Group

Show All

Name	Age	Gender	Blood Group	Contact	Actions
Christo Elamthottam	25	Male	O+	8921383658	🗑
Akhil Nedumpuram	26	Male	A+	9495774573	🗑
Jose Mon Joseph	29	Male	O+	9745128467	🗑
Thomaskutty Nambiyamadathil	30	Male	A+	8547254896	🗑
Varghese AC	35	Male	A+	9447600963	🗑

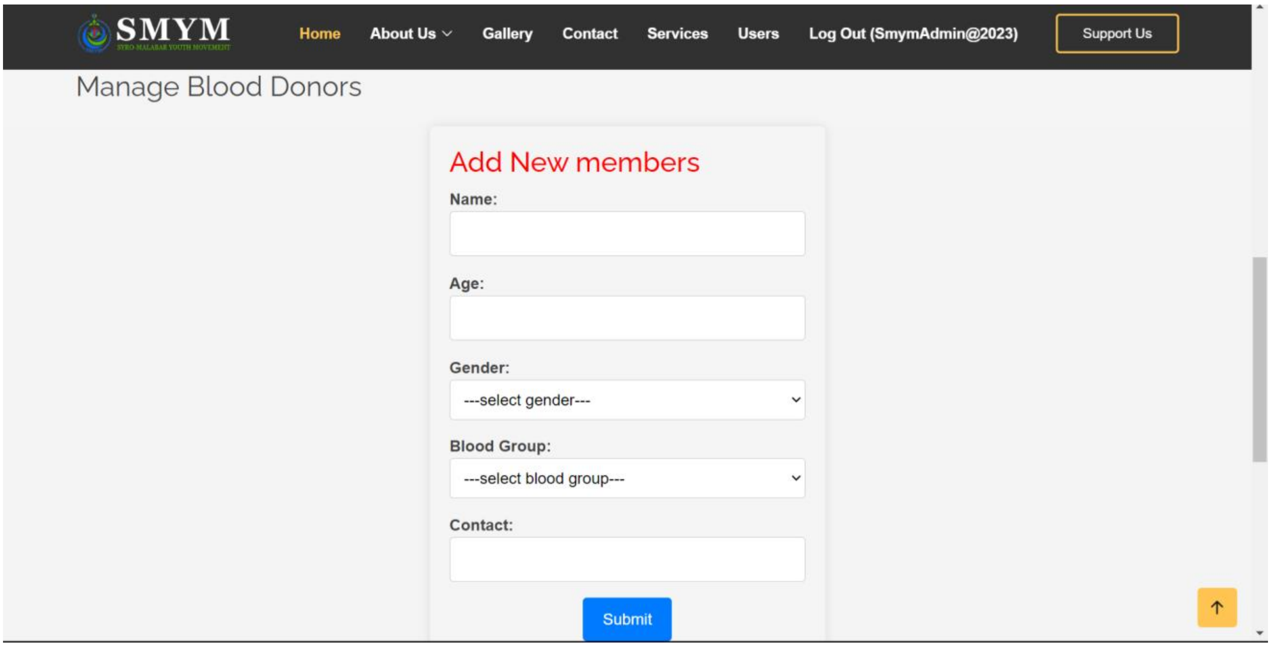


Blood Donors

Filter by Blood Group

B+

Name	Age	Gender	Blood Group	Contact	Actions
Jins Kalamundayi	28	Male	B+	8589946026	🗑
Thomaskutty Ambalathattel	25	Male	B+	7902657087	🗑
Binny Thenakarayil	40	Male	B+	9446289838	🗑
Feby Philip	55	Male	B+	9447104215	🗑
Fr Rajesh kottoor	55	Male	B+	9497202584	🗑
Sabin Joseph	32	Male	B+	9495780845	🗑
Alwin Vallikkattil	23	Male	B+	7510570477	🗑



Manage Blood Donors

Add New members

Name:

Age:

Gender:

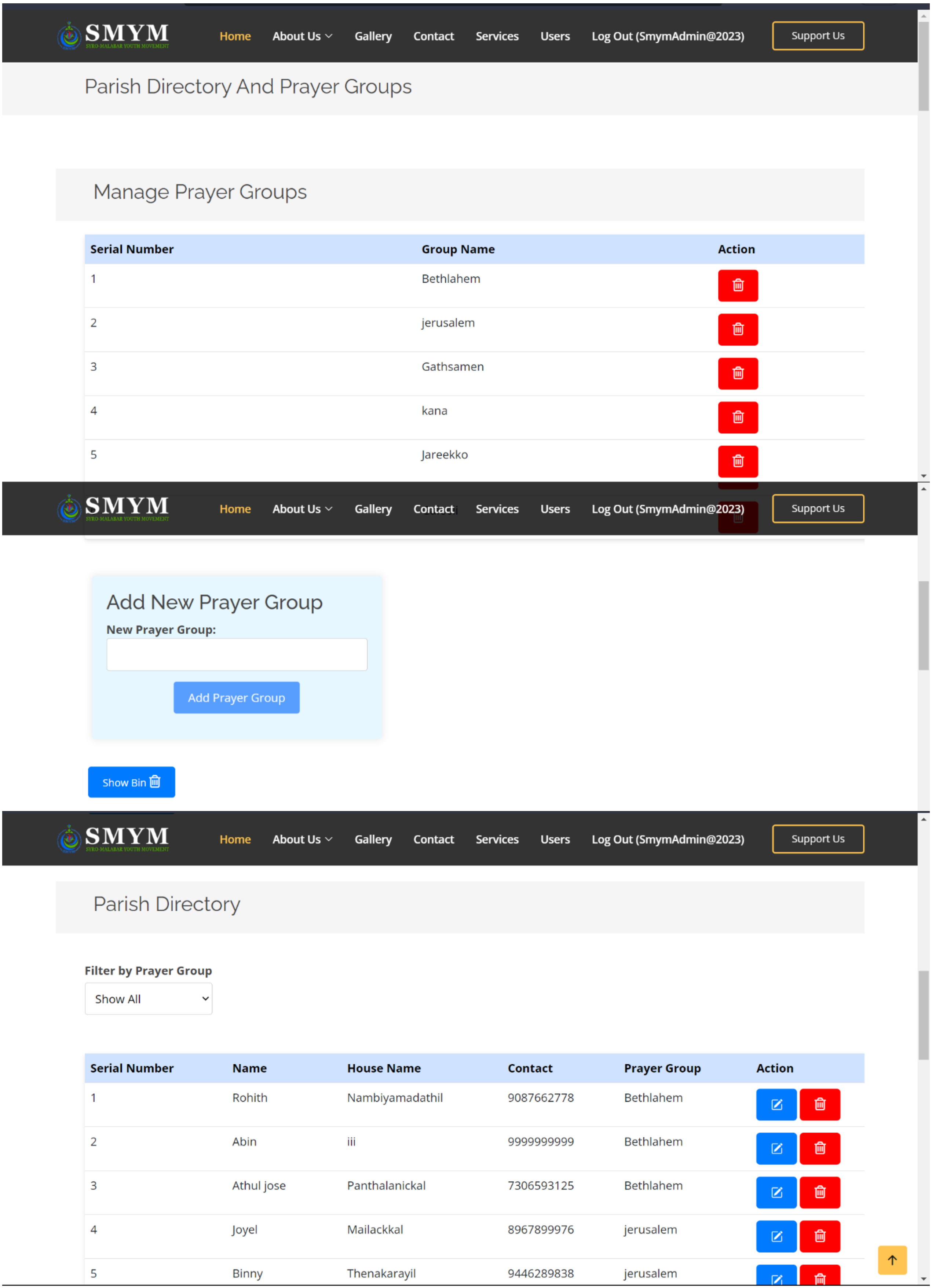
---select gender---

Blood Group:

---select blood group---

Contact:

Parish Directory Page



Parish Directory And Prayer Groups

Manage Prayer Groups

Serial Number	Group Name	Action
1	Bethlahem	🗑
2	jerusalem	🗑
3	Gathsamen	🗑
4	kana	🗑
5	Jareekko	🗑

Add New Prayer Group

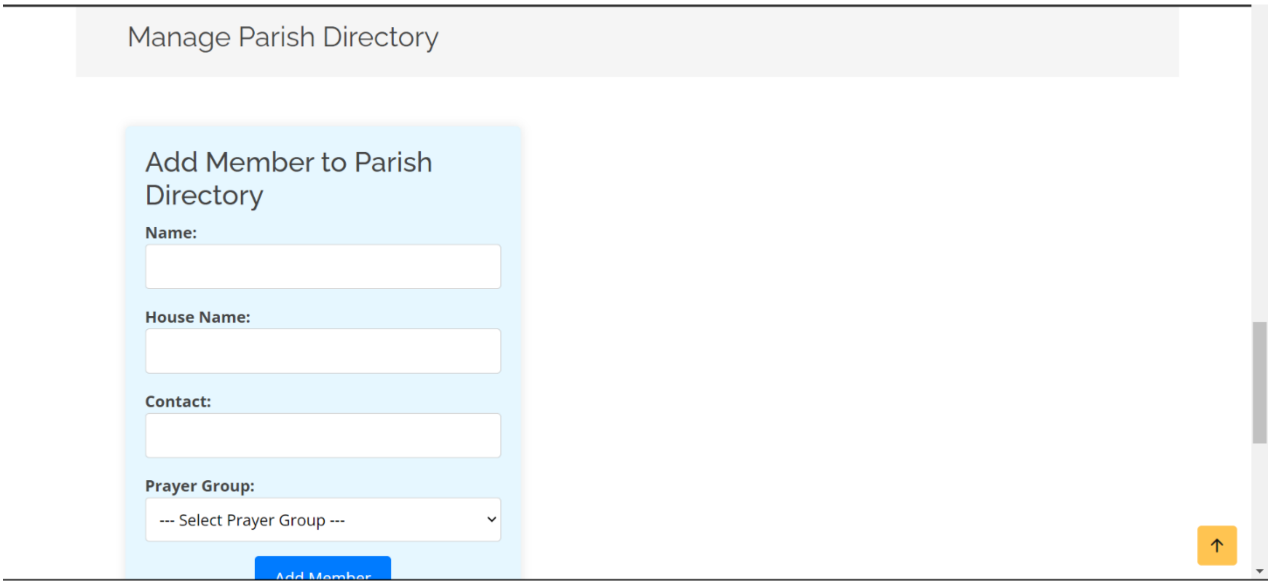
New Prayer Group:

Parish Directory

Filter by Prayer Group

Show All

Serial Number	Name	House Name	Contact	Prayer Group	Action
1	Rohith	Nambiyamadathil	9087662778	Bethlahem	✎ 🗑
2	Abin	iii	9999999999	Bethlahem	✎ 🗑
3	Athul jose	Panthalanickal	7306593125	Bethlahem	✎ 🗑
4	Joyel	Mailackkal	8967899976	jerusalem	✎ 🗑
5	Binny	Thenakarayil	9446289838	jerusalem	✎ 🗑



Manage Parish Directory

Add Member to Parish Directory

Name:

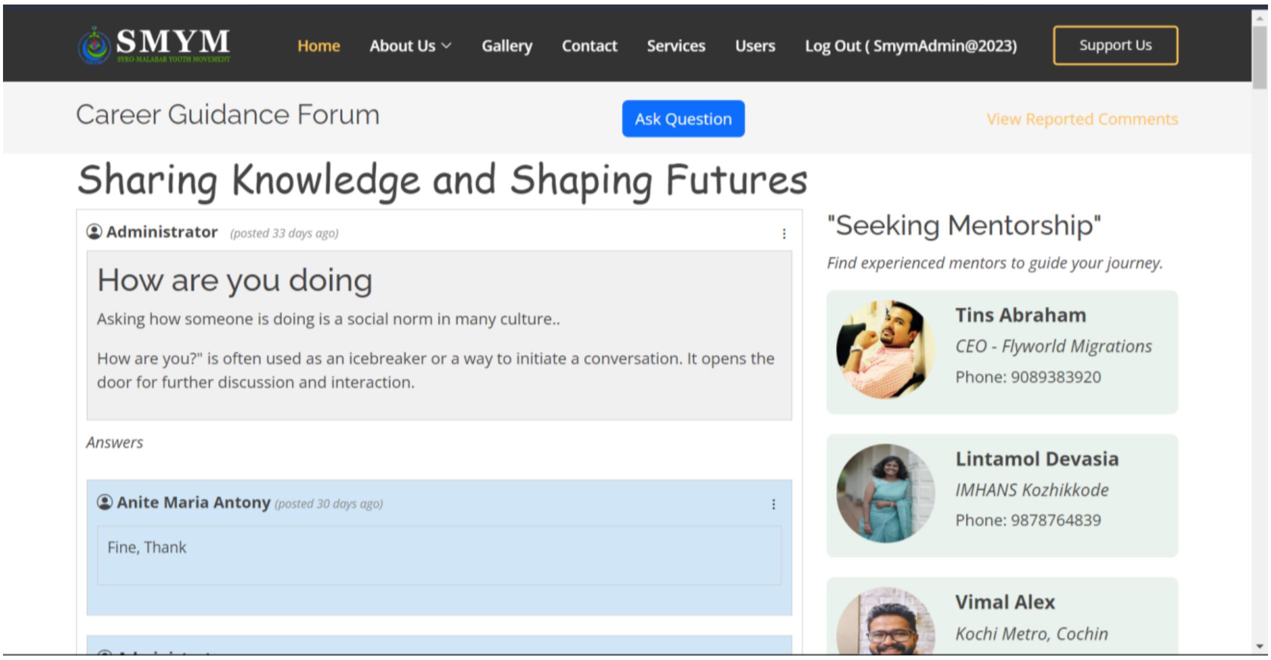
House Name:

Contact:

Prayer Group:

--- Select Prayer Group ---

Career Guidance Page



Career Guidance Forum

Sharing Knowledge and Shaping Futures

Administrator (posted 33 days ago)

How are you doing

Asking how someone is doing is a social norm in many culture..

How are you?" is often used as an icebreaker or a way to initiate a conversation. It opens the door for further discussion and interaction.

Answers

Anite Maria Antony (posted 30 days ago)

Fine, Thank

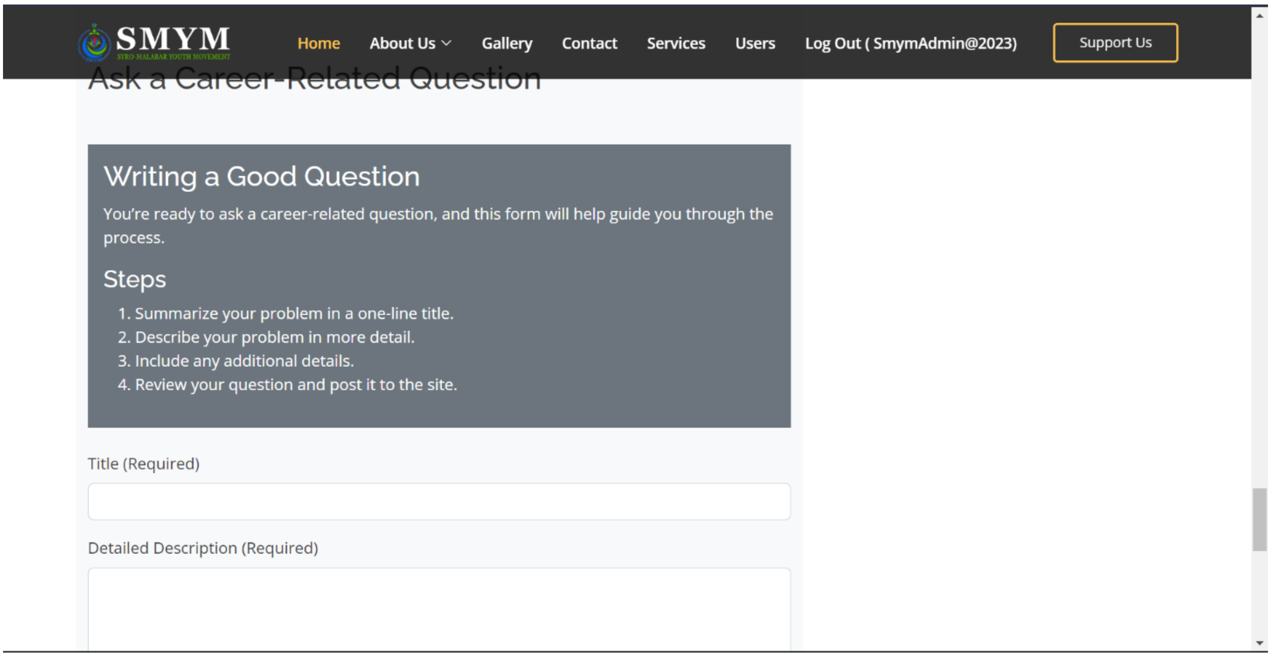
"Seeking Mentorship"

Find experienced mentors to guide your journey.

Tins Abraham
CEO - Flyworld Migrations
Phone: 9089383920

Lintamol Devasia
IMHANS Kozhikkode
Phone: 9878764839

Vimal Alex
Kochi Metro, Cochin



Ask a Career-Related Question

Writing a Good Question

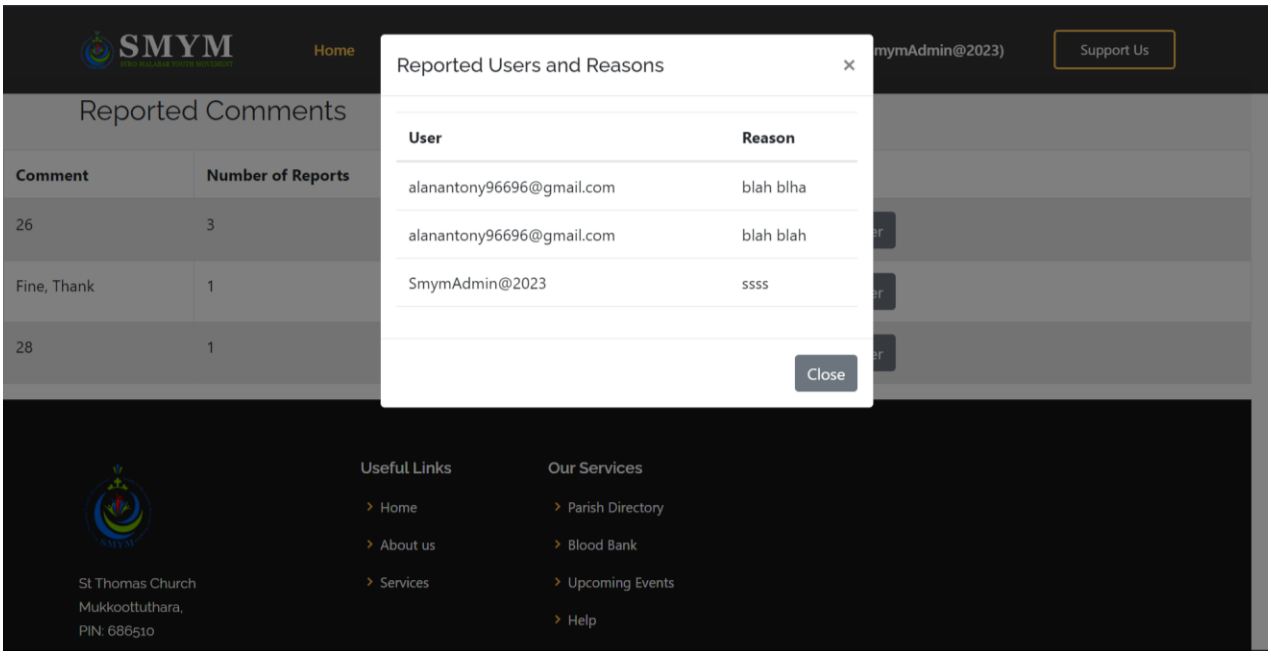
You're ready to ask a career-related question, and this form will help guide you through the process.

Steps

1. Summarize your problem in a one-line title.
2. Describe your problem in more detail.
3. Include any additional details.
4. Review your question and post it to the site.

Title (Required)

Detailed Description (Required)



Reported Comments

Comment	Number of Reports
26	3
Fine, Thank	1
28	1

Reported Users and Reasons

User	Reason
alanantony96696@gmail.com	blah blha
alanantony96696@gmail.com	blah blah
SmymAdmin@2023	ssss

Useful Links
- Home
- About us
- Services

Our Services
- Parish Directory
- Blood Bank
- Upcoming Events
- Help

St Thomas Church
Mukkoottuthara,
PIN: 686510