E-Masjid System

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Final Year Project Report

**Project ID:22-KS-BSIT-15**

by

|  |  |
| --- | --- |
| **Dawood Ahmed** | **089264 (2022-KS-158)** |
| **Haris Ehsan** | **089301 (2022-KS-190)** |
|  |  |

Project Advisor:

**Muhammad Kamran**

Govt. Graduate College, Civil Lines, Sheikhupura

University of the Punjab, Lahore, Pakistan.

(2022-2026)

# **E-Masjid System**

## **Executive Summary**

The E-Masjid System is a web based system and is built to help mosques to manage their daily activities in a digital way. In many places, mosques are still using manual registers for the records of donations and expenses and announcements are made only through loudspeakers. This creates problems like lack of transparency, difficulties in keeping records, inconvenience for people who want to donate or book services like Nikah. This project was developed in order to fix these problems by developing an online system which will be easy to use. The system allow the mosque admin to control the times of prayers, post announcements and also maintain proper records of donations and expenses. Community members are able to donate online via Stripe, view updates from the mosque, register for events and book Nikah services without physically visiting the mosque. The website is designed simply so people of all ages will be able to use it. This document contains the final project for our Final Year Project which includes the project proposal, software requirements, system design and user interface prototype.

# FYDP Overview

**FYDP Title : E-Masjid System**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Roll Numbers** | **Name** | **Signatures** |
| 1. | 089264 | Dawood Ahmed |  |
| 2. | 089301 | Haris Ehsan |  |

Table Project Proposal Summary

|  |
| --- |
| **FYDP Goals** |
| To make digital platform for mosque management that make operations easy, give |
| transparency in donations and make services accessible for community. |
| **FYDP Objectives** |
| Display and manage prayer timings |
| Create transparent donation tracking system |
| Organize events and programs in the community |
| Enable online Nikah service booking |
| Perform secure user role management |
| **FYDP Success Criteria** |
| This system will be considered as successful if, |
| It shows records of donations. |
| It shows records of expenditures |
| Admin can organize events using the system |
| People can book nikah registrar using the system |
| **Assumptions:** |
| Mosque will use system |
| People have internet |
| **Risks & Obstacles** |
| Traditional people may resist new system |
| People face difficulty with technology |
| Internet problems |
| **Organization Address:** Government Graduate College Civil lines Sheikhupura |
| **Target End Users** Mosque administrators, committee members, community members, donors, families needing nikah registrarservices |
| **Suggested Project Supervisor: Muhammad Kamran** |
| **Approved By: Muhammad Kamran** |
| **Date: October 24, 2025** |

Table of Contents

[E-Masjid System 2](#_Toc220336958)

[Executive Summary 2](#_Toc220336959)

[FYDP Overview 3](#_Toc220336960)

[Chapter No 1 10](#_Toc220336961)

[Project Proposal 10](#_Toc220336962)

[1.1. Introduction 11](#_Toc220336963)

[1.2. Background 11](#_Toc220336964)

[1.3. Problem Statement 11](#_Toc220336965)

[1.4. Stakeholders & Interests 11](#_Toc220336966)

[1.5. Objectives: 12](#_Toc220336967)

[1.6. Scope: 12](#_Toc220336968)

[In-scope 12](#_Toc220336969)

[Out-of-scope 12](#_Toc220336970)

[1.7. Assumptions 13](#_Toc220336971)

[1.8. Risks 13](#_Toc220336972)

[1.9. Success Criteria 14](#_Toc220336973)

[1.10. Tools, Libraries & Technologies 14](#_Toc220336974)

[1.11. Work Division 15](#_Toc220336975)

[1.12. Conclusion 15](#_Toc220336976)

[Chapter No 2 16](#_Toc220336977)

[Literature Review 16](#_Toc220336978)

[2.1 Literature Survey 17](#_Toc220336979)

[2.2 Related Work: 17](#_Toc220336980)

[2.3 Gap Analysis: 17](#_Toc220336981)

[2.4 Summary 17](#_Toc220336982)

[Chapter No 3 18](#_Toc220336983)

[Software Requirements Specification 18](#_Toc220336984)

[3.1 Requirements Analysis 19](#_Toc220336985)

[3.2 User classes and characteristics 19](#_Toc220336986)

[3.3 Requirement Identifying Technique 19](#_Toc220336987)

[3.4 Functional Requirements 20](#_Toc220336988)

[**Functional Requirement 1** 20](#_Toc220336989)

[**Functional Requirement 2** 20](#_Toc220336990)

[**Functional Requirement 3** 21](#_Toc220336991)

[**Functional Requirement 4** 22](#_Toc220336992)

[**Functional Requirement 5** 22](#_Toc220336993)

[**Functional Requirement 6** 23](#_Toc220336994)

[**Functional Requirement 7** 23](#_Toc220336995)

[**Functional Requirement 8** 24](#_Toc220336996)

[**Functional Requirement 9** 25](#_Toc220336997)

[**Functional Requirement 10** 25](#_Toc220336998)

[**Functional Requirement 11** 26](#_Toc220336999)

[3.5 Non-Functional Requirements 26](#_Toc220337000)

[1. Reliability 27](#_Toc220337001)

[2. Usability 27](#_Toc220337002)

[3. Performance 27](#_Toc220337003)

[4. Security 27](#_Toc220337004)

[3.6 External Interface Requirements 27](#_Toc220337005)

[1. User Interfaces Requirements 28](#_Toc220337006)

[2. Software Interfaces 28](#_Toc220337007)

[3. Hardware Interfaces 28](#_Toc220337008)

[4. Communications Interfaces 29](#_Toc220337009)

[3.7 Use case Analysis 29](#_Toc220337010)

[Community User Use Cases 29](#_Toc220337011)

[Use Case #1 – Register User 29](#_Toc220337012)

[Use Case #2 – Login User 30](#_Toc220337013)

[Use Case #3 – Forgot Password 30](#_Toc220337014)

[Use Case #4 – View Prayer Times 31](#_Toc220337015)

[Use Case #5 – View Events & Register 31](#_Toc220337016)

[Use Case #6 – View Donation Records 32](#_Toc220337017)

[Use Case #7 – View Announcements 32](#_Toc220337018)

[Use Case #8 – Make Donation 33](#_Toc220337019)

[Use Case #9 – Book Nikah Services 34](#_Toc220337020)

[Use Case #10 – View Booking Status 34](#_Toc220337021)

[Admin Use Cases 35](#_Toc220337022)

[Use Case #11 – Admin Login 35](#_Toc220337023)

[Use Case #12 – Manage Donations & Expenses 35](#_Toc220337024)

[Use Case #13 – Manage Prayer Times 36](#_Toc220337025)

[Use Case #14 – Manage Events 36](#_Toc220337026)

[Use Case #15 – Manage Announcements 37](#_Toc220337027)

[Use Case #16 – Manage Religious Scholar Account 38](#_Toc220337028)

[Religious Scholar Use Case 38](#_Toc220337029)

[Use Case #17 – Check Nikah Requests 38](#_Toc220337030)

[Use Case Diagram 39](#_Toc220337031)

[3.8 Storyboards 41](#_Toc220337032)

[Storyboard 1 – Online Donation System 41](#_Toc220337033)

[Storyboard 2 – Event Management 42](#_Toc220337034)

[Storyboard 3 – Nikah Booking 42](#_Toc220337035)

[3.9 Summary 43](#_Toc220337036)

[Chapter No 4 44](#_Toc220337037)

[Software Design Specification 44](#_Toc220337038)

[4.1 System Design 45](#_Toc220337039)

[4.2 Design Considerations 46](#_Toc220337040)

[**Assumptions** 46](#_Toc220337041)

[**Dependencies** 46](#_Toc220337042)

[**Limitations** 46](#_Toc220337043)

[**Risks** 46](#_Toc220337044)

[4.3 Requirements Traceability Matrix 47](#_Toc220337045)

[4.4 Design Models 48](#_Toc220337046)

[1. Design Class Diagram 48](#_Toc220337047)

[2. Sequence Diagram 49](#_Toc220337048)

[3. State Transition Diagram 52](#_Toc220337049)

[4.5 Architectural Design 55](#_Toc220337050)

[1. UML Component diagram 56](#_Toc220337051)

[4.6 Data Design 57](#_Toc220337052)

[1. Logical Data Model 58](#_Toc220337053)

[2. Data Dictionary 58](#_Toc220337054)

[4.7 User Interface Design 61](#_Toc220337055)

[1. Screen Images 62](#_Toc220337056)

[2. Screen Objects and Actions 66](#_Toc220337057)

[4.8 Design Decisions 67](#_Toc220337058)

[4.9 Summary 67](#_Toc220337059)

[References 68](#_Toc220337060)

[1. Article in a Journal 69](#_Toc220337061)

[2. World Wide Web 69](#_Toc220337062)

**List of Tables**

[Table 1 Project Proposal Summary 3](#_Toc220174875)

[Table 2 Project Success Criteria 14](#_Toc220174876)

[Table 3 Tools Technologies and Libraries 14](#_Toc220174877)

[Table 4 Project Team Members Work Division 15](#_Toc220174878)

[Table 5 User classes 19](#_Toc220174879)

[Table 6 Functional Requirement 1 20](#_Toc220174880)

[Table 7 Functional Requirement 2 20](#_Toc220174881)

[Table 8 Functional Requirement 3 21](#_Toc220174882)

[Table 9 Functional Requirement 4 22](#_Toc220174883)

[Table 10 Functional Requirement 5 22](#_Toc220174884)

[Table 11 Functional Requirement 6 23](#_Toc220174885)

[Table 12 Functional Requirement 7 23](#_Toc220174886)

[Table 13 Functional Requirement 8 24](#_Toc220174887)

[Table 14 Functional Requirement 9 25](#_Toc220174888)

[Table 15 Functional Requirement 10 25](#_Toc220174889)

[Table 16 Functional Requirement 11 26](#_Toc220174890)

[Table 17 Use Case 1 29](#_Toc220174891)

[Table 18 Use Case 2 30](#_Toc220174892)

[Table 19 Use Case 3 30](#_Toc220174893)

[Table 20 Use Case 4 31](#_Toc220174894)

[Table 21 Use Case 5 31](#_Toc220174895)

[Table 22 Use Case 6 32](#_Toc220174896)

[Table 23 Use Case 7 32](#_Toc220174897)

[Table 24 Use Case 8 33](#_Toc220174898)

[Table 25 Use Case 9 34](#_Toc220174899)

[Table 26 Use Case 10 34](#_Toc220174900)

[Table 27 Use Case 11 35](#_Toc220174901)

[Table 28 Use Case 12 35](#_Toc220174902)

[Table 29 Use Case 13 36](#_Toc220174903)

[Table 30 Use Case 14 36](#_Toc220174904)

[Table 31 Use Case 15 37](#_Toc220174905)

[Table 32 Use Case 16 38](#_Toc220174906)

[Table 33 Use Case 17 38](#_Toc220174907)

[Table 34 Requirements Traceability Matrix 47](#_Toc220174908)

[Table 35 Data Dictionary Table 58](#_Toc220174909)

**Tables of Figures**

[Figure 1 Use Case diagram of Community members 39](#_Toc220337063)

[Figure 2 Use Case diagram of Mosque Admin 40](#_Toc220337064)

[Figure 3 Use Case diagram of Religious Scholar 40](#_Toc220337065)

[Figure 4 Online Donation Storyboard 41](#_Toc220337066)

[Figure 5 Event Management Storyboard 42](#_Toc220337067)

[Figure 6 Nikah booking Storyboard 42](#_Toc220337068)

[Figure 7 Class diagram 48](#_Toc220337069)

[Figure 8 Login Sequence Diagram 49](#_Toc220337070)

[Figure 9 Online Donation Sequence Diagram 50](#_Toc220337071)

[Figure 10 Nikah Booking Sequence Diagram 50](#_Toc220337072)

[Figure 11 Admin Update Prayer Times Sequence Diagram 51](#_Toc220337073)

[Figure 12 User Account State Diagram 52](#_Toc220337074)

[Figure 13 Nikah Booking Status State Diagram 53](#_Toc220337075)

[Figure 14 Event State Diagram 54](#_Toc220337076)

[Figure 15 High-Level System Architecture 55](#_Toc220337077)

[Figure 16 Component diagram 56](#_Toc220337078)

[Figure 17 Logical Data Model Diagram 58](#_Toc220337079)

[Figure 18 Home page design 62](#_Toc220337080)

[Figure 19 Login page design 63](#_Toc220337081)

[Figure 20 Donation Transparency page design 64](#_Toc220337082)

[Figure 21 Admin Dashboard design 65](#_Toc220337083)

[Figure 22 Scholar page design 65](#_Toc220337084)

# Chapter No 1

# Project Proposal

# Introduction

This chapter is an introduction about our Final Year Project E-Masjid System. We explain the reason why we chose this project and what problems will be solved. We also list the tools that we will be using and how our team members will be working together on this project. This chapter gives the complete overview of our project idea.

# Background

In our country, mosques are the place where parent send their children to get Islamic teaching related to our religion. Person of any age can come to mosque and ask questions for which they have doubt in their mind so the person get the answers according to the Islam. Every town has one mosque that handles prayer schedule, organize religious events and collect donations. These days mosques manage their records in registers which become messy and can be destroyed by various causes. Also it is very time consuming process to find the records on urgent bases. So there is a need of digital system where the records will be easily accessible and management of operations become easy.

# Problem Statement

People who donate to the mosque do not know where their donations are used and how they are used. When families need nikah services, they face difficulties finding available nikah registrarquickly. People who are away from their area mosque do not get the updates what announcement and events are going to happen. Another big problem is using paper registers for keeping records. This method is slow and if register get lost or damaged, all important information is gone forever.

# Stakeholders & Interests

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **Description** | **Interest** |
| Mosque  Administration | People who manage mosque operations like imam and committee members | Good management of operations and transparent record keeping |
| Donors | Community people who give money to mosque for different causes | Send their donations and see how funds are being used |
| Community  Members | Local people who regularly visit mosque for prayers and activities | View prayer times, announcements, and events, register for events, request services easily. |
| Religious  Scholars | Qualified Islamic scholars who perform nikah ceremonies and religious duties | Manage availability for Nikah, share special prayer timings, avoid doublebooking. |

# Objectives:

1. To make a web based system for mosque management using MERN stack.
2. To manage donations in a clear way and record of where money is spent.
3. To show daily prayer times and special timings for Jummah and Ramadan.
4. To make an event section for Islamic classes, charity and other community programs.
5. To create an online system for booking nikah registrarfor marriage.
6. To give different access to admin and normal users for security and management.

# Scope:

### In-scope

1. Digital management of prayer times and announcements.
2. Transparent tracking and record keeping of donations.
3. Create an event and registration system.
4. Online booking system for Nikah services.
5. User authentication for admin, religious scholar and community members.
6. Single mosque management system.
7. Responsive web design for mobile and desktop.
8. Online payment gateway integration.

### Out-of-scope

* 1. Multi mosque system
  2. Automated SMS/email notification system

# Assumptions

Following are the assumptions of our system:

1. Mosque management know basic computer operations like using a browser and typing.
2. Users have internet connection and email addresses.
3. The mosque has at least one computer or laptop for the admin to use.
4. Religious scholars can use simple websites and click buttons.
5. Community members can use web browsers on their phones or computers.
6. People will use the system honestly and enter correct information.

# Risks

Following are the risks that can occur in the development of our project :

1. **Payment security issues**: Someone may attempt to steal payment information.

**Mitigation:** We will use Stripe payment gateway which is very secure and helps to follow international security standards.

1. **System downtime during prayer times:** If system stops working when people need to check prayer times

**Mitigation:** We will use a reliable hosting service and will maintain backup of data.

1. **Elderly users finding system difficult:** Old people might not understand how to use website.

**Mitigation:** We are going to design simple interface with big buttons and text to be clear.

1. **Data loss due to system crash:** If database gets damaged, all the records can be lost.

**Mitigation:** We will setup automatic weekly backup of database to cloud storage.

1. **Internet connection problems**: Without the internet, system can not work.

**Mitigation:** We will design system to show cached data in case of offline and sync in connection returns.

# Success Criteria

For the success of our system , the following features must work properly:

Table Project Success Criteria

|  |
| --- |
| User Registration and Login of admin , religious scholar and community members  Donation recording and tracking system  Prayer times management and display  Event creation and registration  Nikah booking system  Financial reports depicting income and expenses  Responsive design to work on mobile phones  Safe payment processing using Stripe  Admin Dashboard to manage all the operations  Simple and easy to use interface  Data back up and recovery system  System deployment on live server |

# Tools, Libraries & Technologies

Table Tools Technologies and Libraries

|  |  |  |  |
| --- | --- | --- | --- |
| **Tools, Libraries,**  **And**  **Technologies** | **Tools** | **Version** | **Rationale** |
| VS Code | Latest | Primary code editor with good extensions |
| **Libraries** | **Version** | **Rationale** |
| React.js , Express.js    JWT , Mongoose | Latest | For building responsive design and creating APIs and handling server side logic. |
| **Technology** | **Version** | **Rationale** |
| React , Nodejs , JS ,  Mongo DB, Stripe API | Latest | To build a complete web app |

# Work Division

Table Project Team Members Work Division

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Roll Number** | **Name** | **Role Assignment & Work Division** |
| 1. | 089264 | Dawood Ahmed | Backend |
| 2. | 089301 | Haris Ehsan | Frontend |

# Conclusion

In this chapter, we introduced our project idea of E-Masjid System. We explained the issues with the current management of mosques and how our digital solution will help. We specified clear objectives and scope of our project. We also identified stakeholders and discussed tools we will be using.

# Chapter No 2

# Literature Review

# Literature Survey

We looked at how mosques currently manage their work. We saw that most mosques in Pakistan use paper registers for keeping records. Some mosques use whatsapp groups for announcements. We also checked some mosque websites and found they only show prayer times. We did not find any complete system that does everything a mosque needs.

# Related Work:

There are some systems which are being used for churches that handle donation , events and prayer schedule. In Pakistan most mosque still use old methods like keeping records in register or making announcements on speaker. Some mosques use google forms for event registration or excel sheets for donation tracking but there is no proper system that provide everything in one place. Most existing Islamic applications focus only on prayer times and Qibla direction, they don't help in mosque management.

# Gap Analysis:

After checking existing solutions, we found several important gaps Firstly, we have seen that no system has been made for Pakistani mosque that handle situations like nikah registrarbooking. Secondly, the current system do not give the financial transparency that people want. Our project fills these gaps with a cheap and easy to use platform.

# Summary

In this chapter, we have observed how mosques work today. We found that most use old methods that have problems. Our system will solve these problems by giving one platform for all mosque work. This chapter gave us an idea about the features that we need to add in our project.

# Chapter No 3

# Software Requirements Specification

# Requirements Analysis

This section explains the detailed requirements for the E-Masjid System. The main purpose of this SRS is to explain what our system will do and who will use it and what functions it will perform. We checked what the system needs to do and identify all the users who will interact with it. This analysis helps us to understand that what features to build and how they should work for different types of users.

## **User classes and characteristics**

Table User classes

|  |  |
| --- | --- |
| **User Class** | **User Characteristics** |
| Mosque Administration | People who manage mosque operations like imam and committee members. They need full control over system and ability to manage all activities. |
| Donor | People who give donations to the mosque. They can view donation records, reports and transparency details. They may donate online or in person. |
| Community Members | Local people who visit mosque regularly. They need to see prayer times, announcements, events and request services. They have limited access. |
| Religious Scholars | Islamic scholars who perform nikah ceremonies. They need to manage their availability and see their booking schedule. |

# **Requirement Identifying Technique**

To find out what our E-Masjid System should do, we used different methods to understand what users really need. First, we talked to mosque committee members and community people to hear about their problems. They told us about manual record keeping, lack of transparency in donations, and how difficult it is to manage events. Then we used the Use Case technique because our system is an interactive website where different users do different things. This helped us understand how community members, mosque admins, and religious scholars will use the system. We made use case diagrams to show all the main features clearly. We also created storyboards to imagine how real users would complete tasks like making donations or booking nikah services. By combining these methods, we made sure we didn't miss any important features and that our system solves real problems people face in mosque management.

# Functional Requirements

We identified 11 main functional requirements for our system. Each feature has specific functional requirements that explain what the system should do. These requirements are written from user’s view to clearly explain the expected behavior.

### **Functional Requirement** **1**

Table Functional Requirement 1

|  |  |
| --- | --- |
| **Identifier** | FR-1 |
| **Title** | User Registration and Login |
| **Requirement** | The system will allow users to register and login with email and password, with different access levels for admin , Religious Scholar and community members. |
| **Source** | System security needs |
| **Inputs** | Email, password, name, phone number, user role |
| **Destination** | User data is stored in database and login result is shown on screen |
| **Outputs** | Successful login message or error message |
| **Rationale** | To protect sensitive information and manage permissions |
| **Business Rule** | Admin users have full access, community users have limited access |
| **Dependencies** | None |
| **Priority** | High |

### **Functional Requirement 2**

Table Functional Requirement 2

|  |  |
| --- | --- |
| **Identifier** | FR-2 |
| **Title** | Show Donation Reports |
| **Requirement** | The system will show donation records, expense details, and financial reports so people can see both income and spending. |
| **Source** | Community trust needs |
| **Inputs** | Selection of report date or month |
| **Destination** | Displayed on admin and user dashboard |
| **Outputs** | List of donations, list of expenses, total income, total expenses |
| **Rationale** | People want to see where their money is spent. |
| **Business Rule** | Reports should show income vs expenses clearly. |
| **Dependencies** | FR-3, FR-10 |
| **Priority** | High |

### **Functional Requirement 3**

Table Functional Requirement 3

|  |  |
| --- | --- |
| **Identifier** | FR-3 |
| **Title** | Record Donations |
| **Requirement** | The mosque admin will be able to record cash donations with donor name, amount, date, and donation type. |
| **Source** | Mosque committee discussion |
| **Inputs** | Donor name, amount, date, donation type |
| **Destination** | Saved in donations collection in database |
| **Outputs** | Donation record created and success message shown |
| **Rationale** | To maintain proper records and show transparency |
| **Business Rule** | Each donation must have at least donor name and amount |
| **Dependencies** | FR-1 |
| **Priority** | High |

### **Functional Requirement 4**

Table Functional Requirement 4

|  |  |
| --- | --- |
| **Identifier** | FR-4 |
| **Title** | Event & Announcement Management |
| **Requirement** | The admin will be able to add, update, or remove events and announcements such as islamic classes, community programs, and eid prayers. Users can view them on the main page. |
| **Source** | Community engagement needs |
| **Inputs** | Event title, date, time, description, announcement details |
| **Destination** | Stored in database and shown on website homepage |
| **Outputs** | Event or announcement published successfully |
| **Rationale** | Helps mosque communicate better with community |
| **Business Rule** | Events should show date, time and location clearly |
| **Dependencies** | FR-1 |
| **Priority** | Medium |

### **Functional Requirement 5**

Table Functional Requirement 5

|  |  |
| --- | --- |
| **Identifier** | FR-5 |
| **Title** | Book Nikah Services |
| **Requirement** | Community members will be able to book nikah registrar for nikah ceremonies by selecting date and providing contact details |
| **Source** | Community service needs |
| **Inputs** | Preferred date, time, contact information, ceremony details |
| **Destination** | Saved in nikah bookings collection |
| **Outputs** | Booking request submitted and confirmation message |
| **Rationale** | People need easy way to arrange marriage ceremonies |
| **Business Rule** | Booking requests must include confirm date and contact information |
| **Dependencies** | FR-1 |
| **Priority** | Medium |

### **Functional Requirement 6**

Table Functional Requirement 6

|  |  |
| --- | --- |
| **Identifier** | FR-6 |
| **Title** | Manage Prayer Times |
| **Requirement** | The admin will be able to set and update daily prayer times including special timings for Jummah and Ramadan. |
| **Source** | Community feedback |
| **Inputs** | Fajr, Zuhr, Asr, Maghrib, Isha times, special timings |
| **Destination** | Saved in prayer times collection |
| **Outputs** | Updated prayer times visible to everyone |
| **Rationale** | People need accurate prayer schedules. |
| **Business Rule** | Prayer times must be visible without login. |
| **Dependencies** | FR-1 |
| **Priority** | High |

### **Functional Requirement 7**

Table Functional Requirement 7

|  |  |
| --- | --- |
| **Identifier** | FR-7 |
| **Title** | Online Donation System |
| **Requirement** | Community members will be able to make donations online through the website. They will enter the amount and personal details, process payments securely using Stripe, and see a success message on the screen after the payment is processed. |
| **Source** | Community needs an easier way to donate |
| **Inputs** | Donation amount, card details, donor information |
| **Destination** | Saved in donations collection and sent to Stripe for processing |
| **Outputs** | Payment success or failure message and donation recorded |
| **Rationale** | People want to donate easily without visiting mosque |
| **Business Rule** | Each online donation must record donor information and amount |
| **Dependencies** | FR-1 |
| **Priority** | Medium |

### **Functional Requirement 8**

Table Functional Requirement 8

|  |  |
| --- | --- |
| **Identifier** | FR-8 |
| **Title** | Password Reset via Email |
| **Requirement** | If users forget their password, they can reset it using their email address. |
| **Source** | Community feedback |
| **Inputs** | User email address |
| **Destination** | Email sent to user with reset link |
| **Outputs** | Password reset email sent and password change successful |
| **Rationale** | People often forget passwords and need an easy way to get back into their account |
| **Business Rule** | The reset link should work only for 24 hours and can be used one time |
| **Dependencies** | FR-1 |
| **Priority** | High |

### **Functional Requirement 9**

Table Functional Requirement 9

|  |  |
| --- | --- |
| **Identifier** | FR-9 |
| **Title** | Check Nikah Booking Status |
| **Requirement** | Users can see if their Nikah service request is pending, accepted, or rejected |
| **Source** | Community feedback |
| **Inputs** | User clicks on "My Nikah Bookings" |
| **Destination** | Displayed on user my booking page |
| **Outputs** | List of bookings with current status |
| **Rationale** | People want to know what's happening with their booking request |
| **Business Rule** | The status should update automatically when scholar changes it |
| **Dependencies** | FR-5 |
| **Priority** | Medium |

### **Functional Requirement 10**

Table Functional Requirement 10

|  |  |
| --- | --- |
| **Identifier** | FR-10 |
| **Title** | Record Mosque Expenses |
| **Requirement** | Admin can add where mosque money is spend like for repairs, electricity etc. |
| **Source** | Community trust issues |
| **Inputs** | Expense description, amount, date, category |
| **Destination** | Saved in expenses collection |
| **Outputs** | Expense record added successfully |
| **Rationale** | Community wants to see how their donations are being used |
| **Business Rule** | Every expense must have description, amount, and date |
| **Dependencies** | FR-1 |
| **Priority** | High |

### **Functional Requirement 11**

Table Functional Requirement 11

|  |  |
| --- | --- |
| **Identifier** | FR-11 |
| **Title** | Create Scholar Accounts |
| **Requirement** | Admin can create special accounts for religious scholars who perform Nikah |
| **Source** | Use Case Analysis |
| **Inputs** | Scholar name, email, phone, specialization |
| **Destination** | Saved in users collection with "scholar" role |
| **Outputs** | Scholar account created and scholar can login |
| **Rationale** | Scholars need their own accounts to manage Nikah requests |
| **Business Rule** | Only admin can create and manage these special accounts |
| **Dependencies** | FR-1 |
| **Priority** | Medium |

# Non-Functional Requirements

This section describes the quality requirements of our system that how it should perform and how easy it should be to use and how secure it should be.

## Reliability

The system must be reliable in the day to day running of the mosque. It should not crash frequently and should recover quickly if any problems occur.

1. The system should be available 95% of the time during peak hours
2. If system goes down, it should recover within 30 minutes
3. Donation data should not be lost even if system has problems
4. Important data should be automatically backed up after every week

## Usability

The system must be easy to use and less complex to the mosque admins and ordinary users. All buttons and forms will be clear and labeled properly.

1. Prayer times should be available to new users in 2 clicks
2. Donation recording process should take less than 3 minutes for admin
3. Nikah booking form should be completable within 5 minutes
4. Large fonts and clear buttons should be used by the elderly users in interface
5. Every key feature must be available at home page

## Performance

When there are several people using the system, it should be fast and easily functioning.

1. Prayer times page should load within 3 seconds
2. Donation reports should generate within 5 seconds
3. System should handle up to 100 users at the same during Friday prayers
4. The registration of the event must take a maximum of 5 seconds

## Security

The system should protect sensitive information like donor details and maintain privacy.

1. Database should store user passwords in an encrypted format
2. The system must not allow unauthorized access to administrator functions
3. Session should timeout after 1 hour of inactivity
4. All payment transactions must be secured using SSL encryption.

# External Interface Requirements

This section describes how our E-Masjid system will interact with users and other systems. It covers the user interface design, software connections, and communication methods.

## User Interfaces Requirements

We will have a clean and simple interface on our system which will be effective with the mosque users who are aged, administrators and community members.

**Design Guidelines:**

1. Use simple colors that are common in Islamic design
2. Large buttons and text for easy reading, especially for older users
3. Consistent navigation menu on all pages
4. Prayer times always visible on the header
5. Mobile friendly design that works on smartphones and tablets
6. Use common icons that people can easily understand
7. Error messages in simple language, not technical terms

**Layout Standards:**

1. Homepage shows prayer times, announcements, and quick access to main features
2. Admin dashboard with clear sections for donations, events, and services
3. Forms should be simple with clear labels and instructions
4. Use responsive design that adjusts to different screen sizes

## Software Interfaces

The system will use the following software tools.

**Frontend:**

1. React.js web application running in modern browsers
2. Compatible with mobile browsers on iOS and Android

**Backend:**

1. Node.js server with Express.js framework
2. MongoDB database for storing all data
3. JWT tokens for user authentication

**External Services:**

1. Stripe payment gateway will be used for real online donations
2. Email service for sending password reset links.

## Hardware Interfaces

The system will run on any normal computer or smartphone that has an internet connection and a browser. No special hardware is required, but a basic server will host the system.

## Communications Interfaces

Our system will use web communication:

**Network Requirements:**

1. Web access via Standard HTTP/HTTPS
2. Internet connection needed to operate the system
3. No special network configuration needed

**Communication Features:**

1. Basic website notifications for new announcements
2. No SMS integration initially
3. No email marketing system
4. Simple contact forms for communication

# Use case Analysis

## **Community User Use Cases**

## **Use Case #1 – Register User**

Table Use Case 1

|  |  |
| --- | --- |
| **UC Identifier** | UC1 |
| **Use Case Name** | Register User |
| **Requirements Traceability** | FR-1 |
| **Purpose** | To allow new users to register in the system using email and password. |
| **Priority** | High |
| **Preconditions** | User is not already registered. |
| **Post conditions** | New user account created. |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User opens registration page.  2. Enters name, email, password.  3. System validates input.  4. Account created successfully.  5. User can now login to the system |
| **Alternate Flows** | User already exists then system shows “Email already registered.” |
| **Exceptions** | Invalid input fields or server error. |
| **Includes** | None |

## **Use Case #2 – Login User**

Table Use Case 2

|  |  |
| --- | --- |
| **UC Identifier** | UC2 |
| **Use Case Name** | Login User |
| **Requirements Traceability** | FR-1 |
| **Purpose** | To let registered users log in using email and password. |
| **Priority** | High |
| **Preconditions** | User must be registered. |
| **Post conditions** | User successfully logged in |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User enters email and password.  2. System checks credentials.  3. If correct, login successful.  4. If correct, user is logged in and taken to homepage. |
| **Alternate Flows** | If wrong password then system show “Invalid email or password.” |
| **Exceptions** | Server not responding |
| **Includes** | User Authentication |

## **Use Case #3 – Forgot Password**

Table Use Case 3

|  |  |
| --- | --- |
| **UC Identifier** | UC3 |
| **Use Case Name** | Forgot Password |
| **Requirements Traceability** | FR-8 |
| **Purpose** | To help users who forgot their password |
| **Priority** | High |
| **Preconditions** | User must be registered with valid email |
| **Post conditions** | User can set new password and login |
| **Actors** | |  | | --- | | Community Member | |
| **Extends** | Login User |
| **Main Success Scenario** | |  | | --- | |  |  |  | | --- | | 1. User clicks "Forgot Password" on login page  2. Enters email address  3. System sends password reset link to email  4. User clicks link in email  5. Enters new password  6. System updates password  7. User can now login with new password | |
| **Alternate Flows** | If email not found, system still shows "If email exists, reset link sent" for security |
| **Exceptions** | |  | | --- | |  |  |  | | --- | | If email service is not working | |
| **Includes** | User authentication |

## **Use Case #4 – View Prayer Times**

Table Use Case 4

|  |  |
| --- | --- |
| **UC Identifier** | UC4 |
| **Use Case Name** | View Prayer Times |
| **Requirements Traceability** | FR-6 |
| **Purpose** | To let users see daily prayer timings |
| **Priority** | High |
| **Preconditions** | None |
| **Post conditions** | User can see all prayer times |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | 1. User visits website homepage  2. Prayer times are displayed on top of page  3. User can see Fajr, Zuhr, Asr, Maghrib, Isha times  4. Special timings for Jummah are also shown |
| **Alternate Flows** | User can view weekly prayer schedule |
| **Exceptions** | |  | | --- | |  |   If admin hasn't updated times, show default times |
| **Includes** | None |

## **Use Case #5 – View Events & Register**

Table Use Case 5

|  |  |
| --- | --- |
| **UC Identifier** | UC5 |
| **Use Case Name** | View Events & Register |
| **Requirements Traceability** | FR-4 |
| **Purpose** | To see mosque events and register for them |
| **Priority** | Medium |
| **Preconditions** | User must be logged in to register |
| **Post conditions** | User registered for event |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |  |  | | --- | | 1. User goes to Events page  2. Views list of upcoming events  3. Clicks on event to see details  4. Clicks "Register for Event"  5. System confirms registration  6. User gets confirmation message | |
| **Alternate Flows** | User can view events without login, but needs login to register |
| **Exceptions** | If system cannot register user due to server issueMissing date or invalid contact info. |
| **Includes** | Login User |

## **Use Case #6 – View Donation Records**

Table Use Case 6

|  |  |
| --- | --- |
| **UC Identifier** | UC6 |
| **Use Case Name** | View Donation Records |
| **Requirements Traceability** | FR-2 |
| **Purpose** | To see donation reports and where money is spent |
| **Priority** | High |
| **Preconditions** | User must be logged in to view reports |
| **Post conditions** | User can see financial reports |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User goes to donation report page  2. Views donation records and amounts  3. Sees expense details like "5000 for new fans"  4. Builds trust in mosque management |
| **Alternate Flows** | User can filter reports by date filter |
| **Exceptions** | If no data available, show "No records yet" Missing date or invalid contact info. |
| **Includes** | Login User |

## **Use Case #7 – View Announcements**

Table Use Case 7

|  |  |
| --- | --- |
| **UC Identifier** | UC7 |
| **Use Case Name** | View Announcements |
| **Requirements Traceability** | FR-4 |
| **Purpose** | To read important mosque announcements |
| **Priority** | Medium |
| **Preconditions** | None |
| **Post conditions** | User reads announcements |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User visits website homepage  2. Announcements are shown in main section  3. User reads important updates  4. Urgent announcements are highlighted in red |
| **Alternate Flows** | User can see old announcements |
| **Exceptions** | If no announcements, show "No current announcements" |
| **Includes** | None |

## **Use Case #8 – Make Donation**

Table Use Case 8

|  |  |
| --- | --- |
| **UC Identifier** | UC8 |
| **Use Case Name** | Make Donation |
| **Requirements Traceability** | FR-7 |
| **Purpose** | To donate money to mosque online using real payments |
| **Priority** | Medium |
| **Preconditions** | User must be logged in |
| **Post conditions** | Donation processed and recorded |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member, Donor | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User clicks "Donate Online"  2. Selects donation type  3. Enters amount in rupees  4. Enters card details for Stripe payment  5. Clicks "Donate Now"  6. Stripe processes payment  7. System shows "Donation Successful" |
| **Alternate Flows** | If user is not logged in, system redirects user to Login page before allowing donation. |
| **Exceptions** | If payment fails then show "Payment failed try again" |
| **Includes** | Login User |

## **Use Case #9 – Book Nikah Services**

Table Use Case 9

|  |  |
| --- | --- |
| **UC Identifier** | UC9 |
| **Use Case Name** | Book Nikah Services |
| **Requirements Traceability** | FR-5 |
| **Purpose** | To book religious scholar for marriage ceremony |
| **Priority** | Medium |
| **Preconditions** | User must be logged in |
| **Post conditions** | Nikah booking request submitted |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. User goes to Nikah Services page  2. Selects preferred date and time  3. Fills contact details and ceremony information  4. Submits the request  5. System sends confirmation "Request Submitted" |
| **Alternate Flows** | User can view their previous bookings |
| **Exceptions** | If date not available, show "Please select another date" |
| **Includes** | Login User |

## **Use Case #10 – View Booking Status**

Table Use Case 10

|  |  |
| --- | --- |
| **UC Identifier** | UC10 |
| **Use Case Name** | View Booking Status |
| **Requirements Traceability** | FR-9 |
| **Purpose** | To check if Nikah booking is accepted or pending |
| **Priority** | Medium |
| **Preconditions** | User must be logged in and have booking request |
| **Post conditions** | User sees current status of their request |
| **Actors** | |  | | --- | |  |  |  | | --- | | Community Member | |
| **Extends** | Book Nikah Services |
| **Main Success Scenario** | |  | | --- | |  |   1. User goes to My Bookings page  2. Views list of their Nikah requests  3. See status like Pending, Accepted, or Rejected  4. If accepted, sees confirmed date and time  5. If rejected, sees reason if provided |
| **Alternate Flows** | User can cancel pending requests |
| **Exceptions** | If no bookings, show "No booking requests yet" |
| **Includes** | Login User |

## **Admin Use Cases**

## **Use Case #11 – Admin Login**

Table Use Case 11

|  |  |
| --- | --- |
| **UC Identifier** | UC11 |
| **Use Case Name** | Admin Login |
| **Requirements Traceability** | FR-1 |
| **Purpose** | To let mosque admin access the admin dashboard |
| **Priority** | High |
| **Preconditions** | Admin must have admin account credentials |
| **Post conditions** | Admin gains access to admin dashboard |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to special /admin login URL  2. Enters admin username and password  3. Clicks "Admin Login"  4. System verifies admin details  5. Admin dashboard opens with all management options |
| **Alternate Flows** | If wrong details, show "Invalid admin login" |
| **Exceptions** | If admin account not set up yet |
| **Includes** | User Authentication |

## **Use Case #12 – Manage Donations & Expenses**

Table Use Case 12

|  |  |
| --- | --- |
| **UC Identifier** | UC12 |
| **Use Case Name** | Manage Donations & Expenses |
| **Requirements Traceability** | FR-3, FR-10 |
| **Purpose** | To handle all money records like donations and expenses |
| **Priority** | High |
| **Preconditions** | Admin must be logged in |
| **Post conditions** | Financial records updated |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to Manage Donations & Expenses section  2. Can add new cash donations with donor details  3. Can record expenses like "5000 for mosque repairs"  4. Can edit or delete existing records  5. System updates financial reports automatically |
| **Alternate Flows** | Admin can generate monthly financial reports |
| **Exceptions** | If invalid data entered, show appropriate error messages |
| **Includes** | Admin Login |

## **Use Case #13 – Manage Prayer Times**

Table Use Case 13

|  |  |
| --- | --- |
| **UC Identifier** | UC13 |
| **Use Case Name** | Manage Prayer Times |
| **Requirements Traceability** | FR-6 |
| **Purpose** | To set and update daily prayer timings for community |
| **Priority** | High |
| **Preconditions** | Admin must be logged in |
| **Post conditions** | New prayer times displayed on website |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to Prayer Times section  2. Updates all five daily prayer times  3. Sets special Jummah prayer timings  4. Clicks "Save Times"  5. New times immediately show on website for everyone |
| **Alternate Flows** | Admin can set weekly schedule instead of daily updates |
| **Exceptions** | If invalid time format, show "Please use correct time format" |
| **Includes** | Admin Login |

## **Use Case #14 – Manage Events**

Table Use Case 14

|  |  |
| --- | --- |
| **UC Identifier** | UC14 |
| **Use Case Name** | Manage Events |
| **Requirements Traceability** | FR-4 |
| **Purpose** | To create, update, and manage mosque events |
| **Priority** | Medium |
| **Preconditions** | Admin must be logged in |
| **Post conditions** | Events published and visible to community |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to Events Management section  2. Creates new events with details and registration options  3. Updates existing event information  4. Deletes or cancels events when needed  5. Events appear on website for community registration |
| **Alternate Flows** | Admin can save events as drafts before publishing |
| **Exceptions** | If past date entered, show "Event date cannot be in past" |
| **Includes** | Admin Login |

## **Use Case #15 – Manage Announcements**

Table Use Case 15

|  |  |
| --- | --- |
| **UC Identifier** | UC15 |
| **Use Case Name** | Manage Announcements |
| **Requirements Traceability** | FR-4 |
| **Purpose** | To post and manage important mosque announcements |
| **Priority** | Medium |
| **Preconditions** | Admin must be logged in |
| **Post conditions** | Announcements visible on website |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to Announcements section  2. Creates new announcements with titles and details  3. Updates existing announcements if information changes  4. Deletes old or incorrect announcements  5. Marks announcements as urgent if important  6. Announcements show on website immediately |
| **Alternate Flows** | Admin can schedule announcements for future dates |
| **Exceptions** | If announcement too long, show "Please shorten announcement" |
| **Includes** | Admin Login |

## **Use Case #16 – Manage Religious Scholar Account**

Table Use Case 16

|  |  |
| --- | --- |
| **UC Identifier** | UC16 |
| **Use Case Name** | Manage Religious Scholar Account |
| **Requirements Traceability** | FR-11 |
| **Purpose** | To create and manage account for Nikah service scholars |
| **Priority** | Medium |
| **Preconditions** | Admin must be logged in |
| **Post conditions** | Scholar accounts created and activated for Nikah management |
| **Actors** | |  | | --- | |  |  |  | | --- | | Mosque Admin | |
| **Extends** | None |
| **Main Success Scenario** | |  | | --- | |  |   1. Admin goes to Scholar Management section  2. Creates new scholar accounts with login credentials  3. Updates scholar account details if needed  4. Deletes or deactivates scholar accounts  5. Scholars receive login details and can manage Nikah requests |
| **Alternate Flows** | Admin can reset scholar passwords if forgotten |
| **Exceptions** | If email already used, show "Email already registered" |
| **Includes** | Admin Login |

## **Religious Scholar Use Case**

## **Use Case #17 – Check Nikah Requests**

Table Use Case 17

|  |  |
| --- | --- |
| **UC Identifier** | UC17 |
| **Use Case Name** | Check Nikah Requests |
| **Requirements Traceability** | FR-5 |
| **Purpose** | To allow religious scholars to check new nikah booking requests and update their status as accepted or rejected. |
| **Priority** | Medium |
| **Preconditions** | The religious scholar account must already be created by the mosque admin, and the scholar must be logged into the system. |
| **Post conditions** | The nikah request status is updated to either “Accepted” or “Declined,” and the request status is updated in the admin dashboard. |
| **Actors** | Religious Scholar, Mosque Admin |
| **Extends** | None |
| **Main Success Scenario** | 1.Religious scholar logs into system  2.Goes to "Nikah Requests" page  3.Views list of pending booking requests  4.Clicks on a request to see details  5.Clicks "Accept" or "Decline" button  6.System updates request status  7.Admin sees updated status in dashboard |
| **Alternate Flows** | If no requests available, system shows "No pending requests"  If scholar tries to accept conflicting time, system shows "Time not available" |
| **Exceptions** | If system error occurs, shows "Unable to update request" |
| **Includes** | User authentication |

## Use Case Diagram

These diagrams shows all the users and features of our E-Masjid system. It helps show how different users interact with different parts of the system.

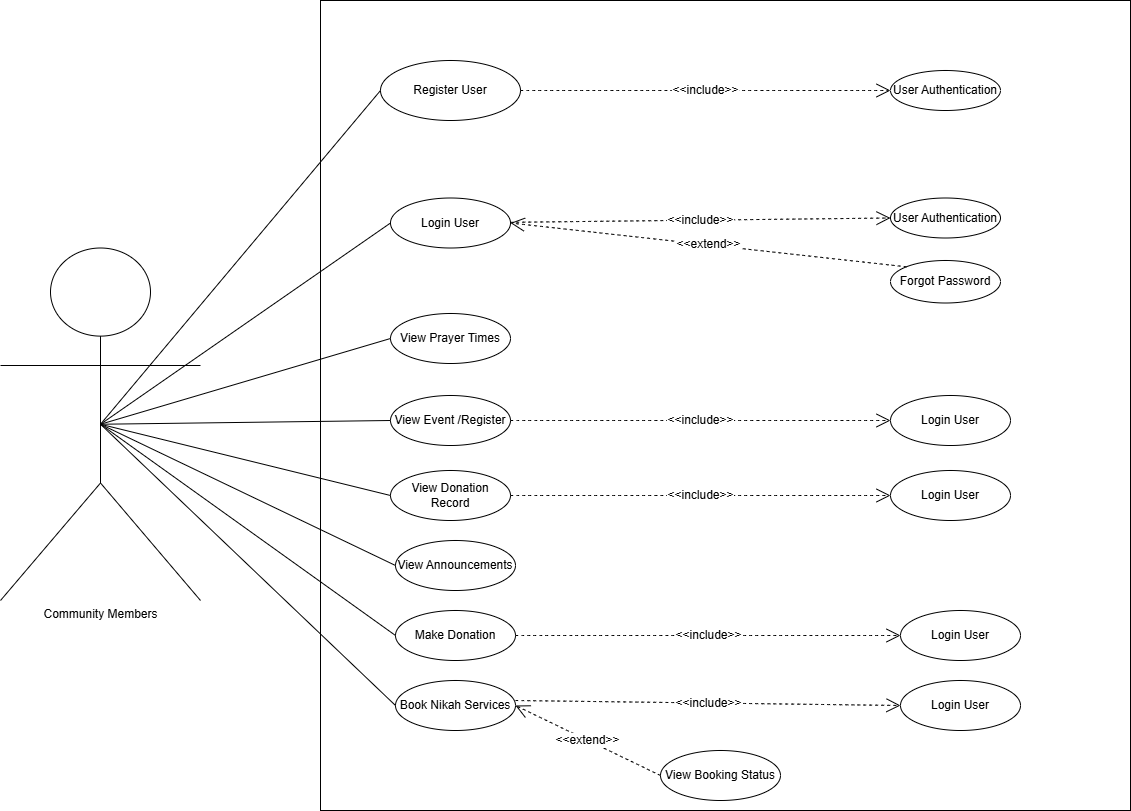


Figure Use Case diagram of Community members

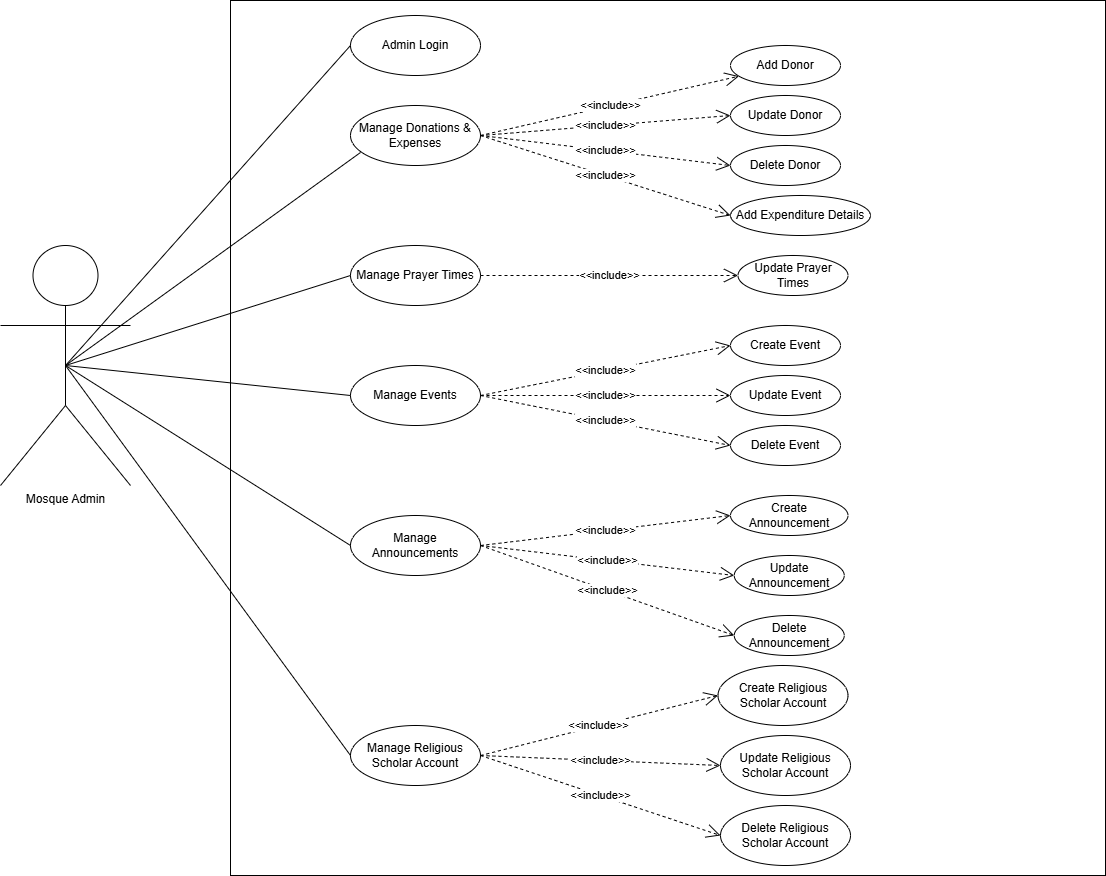
****

Figure Use Case diagram of Mosque Admin

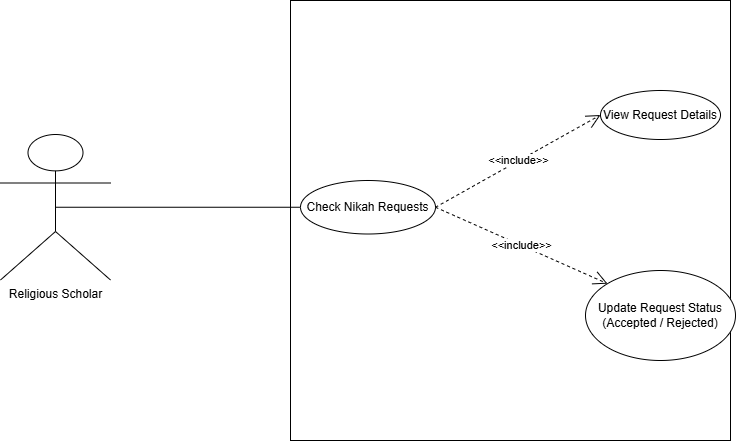
****

Figure Use Case diagram of Religious Scholar

# Storyboards

This section shows how users will use our system in real life. Each storyboard explains one main feature that happen on screen.

## **Storyboard 1 – Online Donation System**

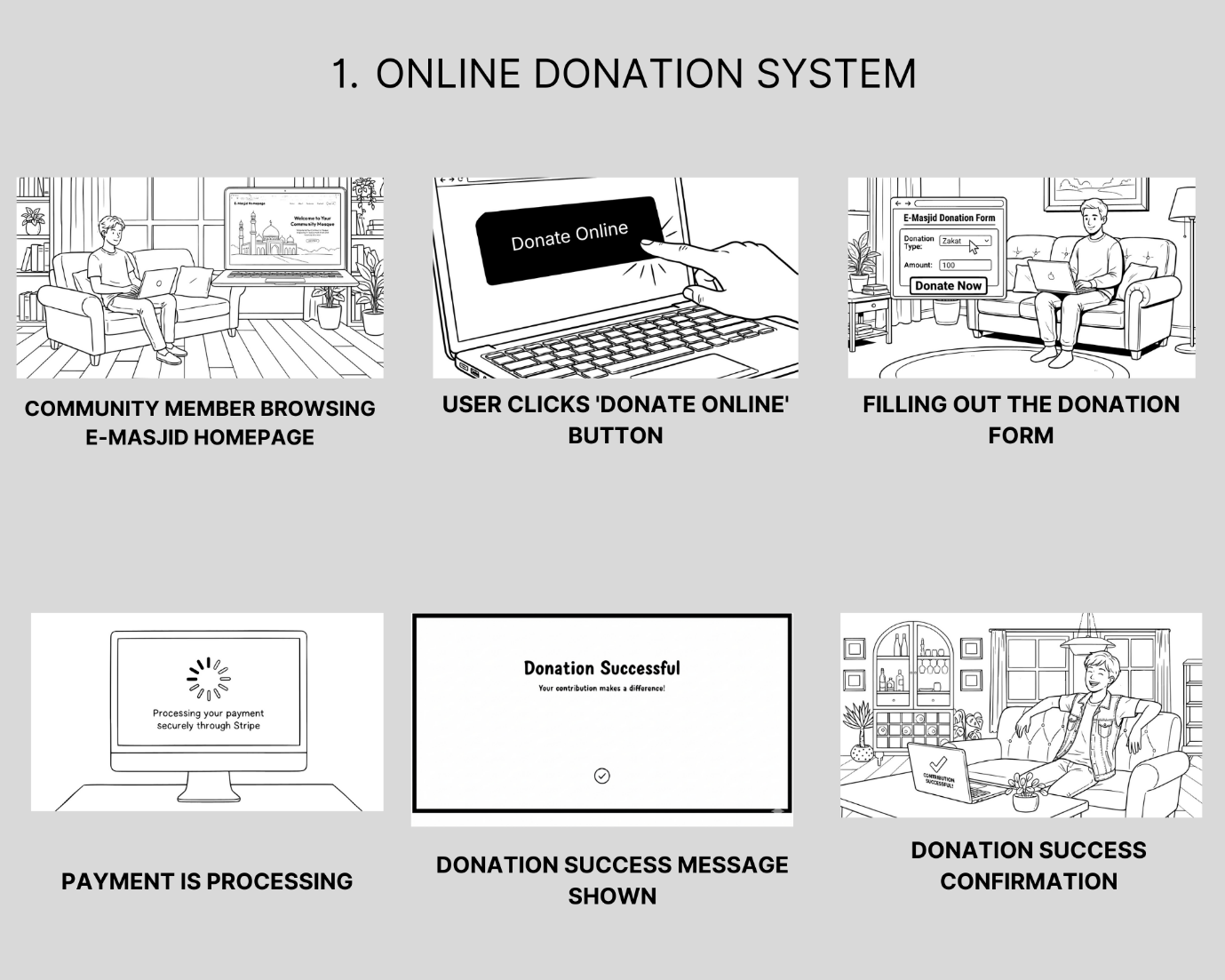


Figure Online Donation Storyboard

## **Storyboard 2 – Event Management**

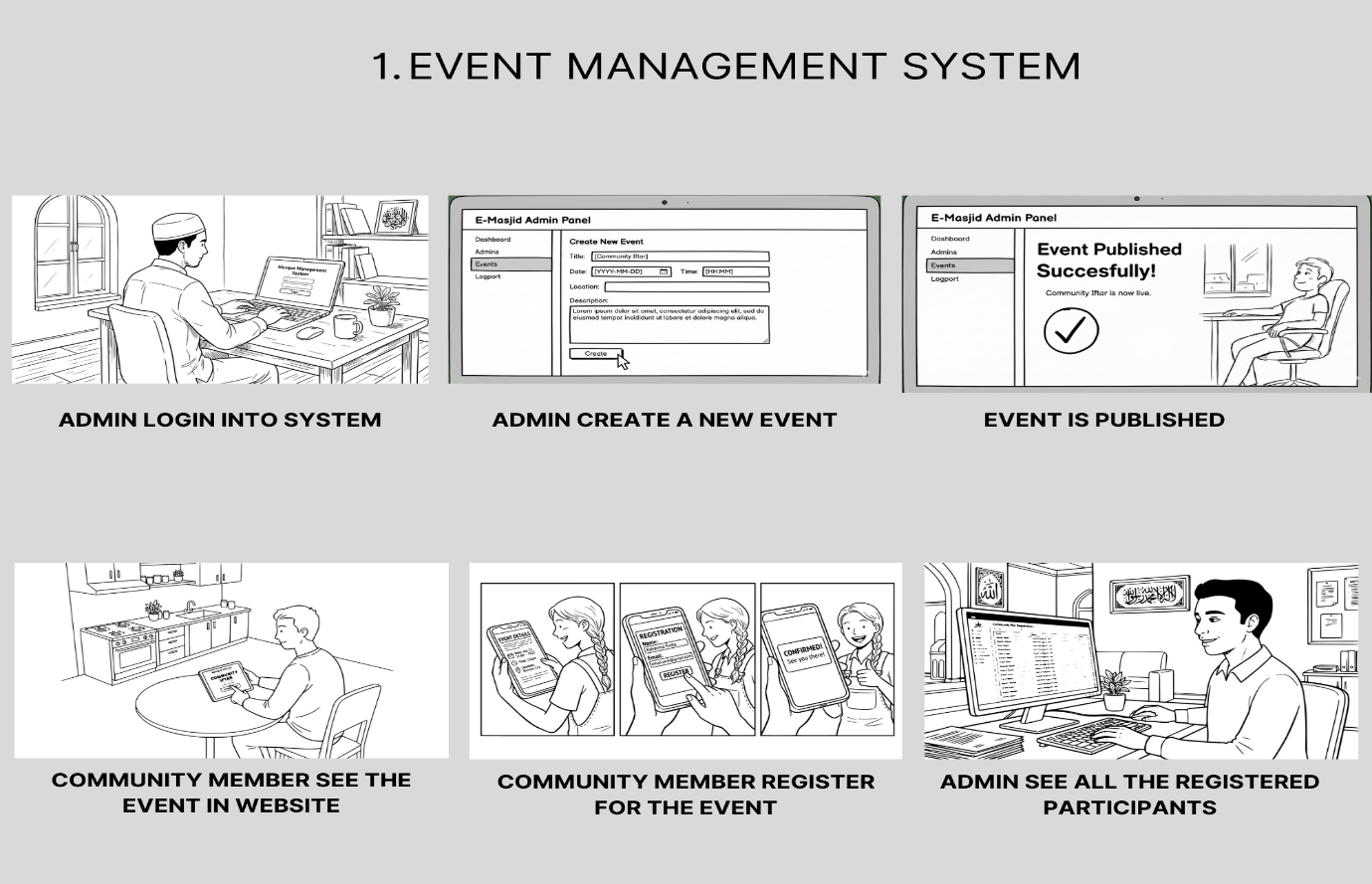


Figure Event Management Storyboard

## **Storyboard 3 – Nikah Booking**

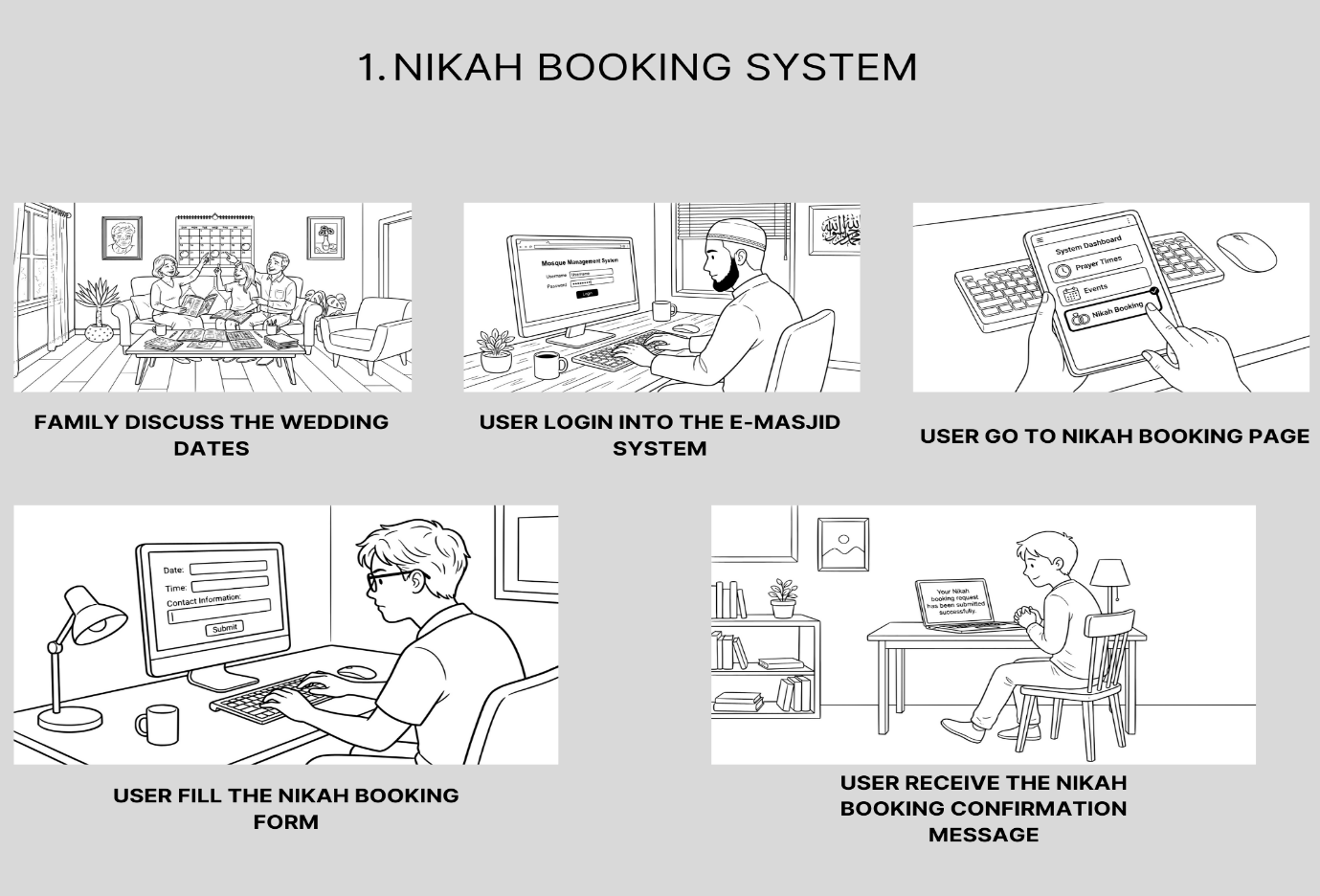


Figure Nikah booking Storyboard

# Summary

In this SRS document, we explained all the main requirements and features of our E-Masjid System. We started by understanding the problems faced by mosque committees and community members, then used use cases and storyboards to identify the real needs of the system. We listed both functional and non-functional requirements like donation management, prayer time setup, service requests, and security. This document helped us to understand what our system will do, who will use it and how each feature will work. It will also help us in the next steps like design and development because all features and requirements are already clear. Overall this SRS gives a complete picture of the system before we start coding.

# Chapter No 4

# Software Design Specification

# System Design

Our E-Masjid System is a complete web based platform that will be accessible through any modern web browser. The system is designed to serve mosque administration for managing operations and community members can use these services.

**Dependencies**

1. Stable internet connection for all users.
2. Stripe service availability for payment processing.
3. Modern web browsers supporting React.js features.
4. MongoDB database server for data storage.

**Interaction with Other Systems**

1. **Stripe Payment Gateway:** It is used for making online donations securely.
2. **Email Service:** It is used for sending password reset links.
3. **Internet Connection:** It is required for all users to access the system.

**Design Constraints**

1. **Performance Requirements:** Prayer times page loads within 3 seconds, handles 100+ users during Friday prayers.
2. **Usability Requirements:** Simple interface with large buttons, works on mobile and computer, elderly friendly design.
3. **Security Requirements:** Encrypted passwords, secure payments through Stripe, admin access protection.
4. **Technical Constraints:** MERN stack technology, responsive design, automatic weekly backups.

# Design Considerations

### **Assumptions**

**Following are the assumptions:**

1. Mosque administrators have basic computer knowledge.
2. Users have internet access and email accounts.
3. The mosque has at least one computer for admin use.
4. Religious scholars can use basic web applications.
5. Community members can use web browsers on phones or computers.

### **Dependencies**

**Following are the dependencies:**

1. Stable internet connection for all users.
2. Stripe payment service available at any time.
3. Web browsers supporting modern JavaScript.
4. MongoDB database running properly.

### **Limitations**

**Following are the limitations:**

1. Cannot work without internet connection.
2. No SMS notifications for announcements.
3. No mobile app version.
4. Payment system requires card payments only.
5. Cannot handle offline data entry.

### **Risks**

**Following are the risks:**

1. Payment security issues.
2. System downtime during prayer times.
3. Elderly users finding the system difficult.
4. Data loss from system crashes

## Requirements Traceability Matrix

Table 34 Requirements Traceability Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Scope** | **Requirement Description** | **Design Specification** |
| FR-1 | User Management | The system will allow users to register and login with email and password, with different access levels for admin, religious scholar and community members | Component "User Authentication Module" |
| FR-2 | Financial Reports | The system will show donation records, expense details, and financial reports so people can see both income and spending | Component "Financial Reporting Module" |
| FR-3 | Donation Management | The mosque admin will be able to record cash donations with donor name, amount, date, and donation type | Component "Donation Management Module" |
| FR-4 | Event Management | The admin will be able to add, update, or remove events and announcements such as islamic classes, community programs, and eid prayers. Users can view them on the main page | Component "Event & Announcement Manager" |
| FR-5 | Nikah Services | Community members will be able to book nikah registrar for nikah ceremonies by selecting date and providing contact details | Component "Nikah Booking Module" |
| FR-6 | Prayer Times | The admin will be able to set and update daily prayer times including special timings for Jummah and Ramadan | Component "Prayer Times Manager" |
| FR-7 | Online Payments | Community members will be able to make donations online through the website by entering amount and personal details and process real payments using Stripe and see payment success message | Component "Online Payment Processor" |
| FR-8 | Account Security | If users forget their password, they can reset it using their email address | Component "Password Recovery Module" |
| FR-9 | Booking Status | Users can see if their Nikah service request is pending, accepted, or rejected | Component "Booking Status Tracker" |
| FR-10 | Expense Tracking | Admin can add where mosque money is spend like for repairs, electricity etc | Component "Expense Management Module" |
| FR-11 | User Role Management | Admin can create special accounts for religious scholars who perform Nikah | Component "User Account Manager" |

## Design Models

In this section, we show how we designed the E-Masjid System. We made different diagrams to explain the system structure, how data is organized, and how different parts of the system work together. These pictures make the design easy to understand for everyone.

## Design Class Diagram

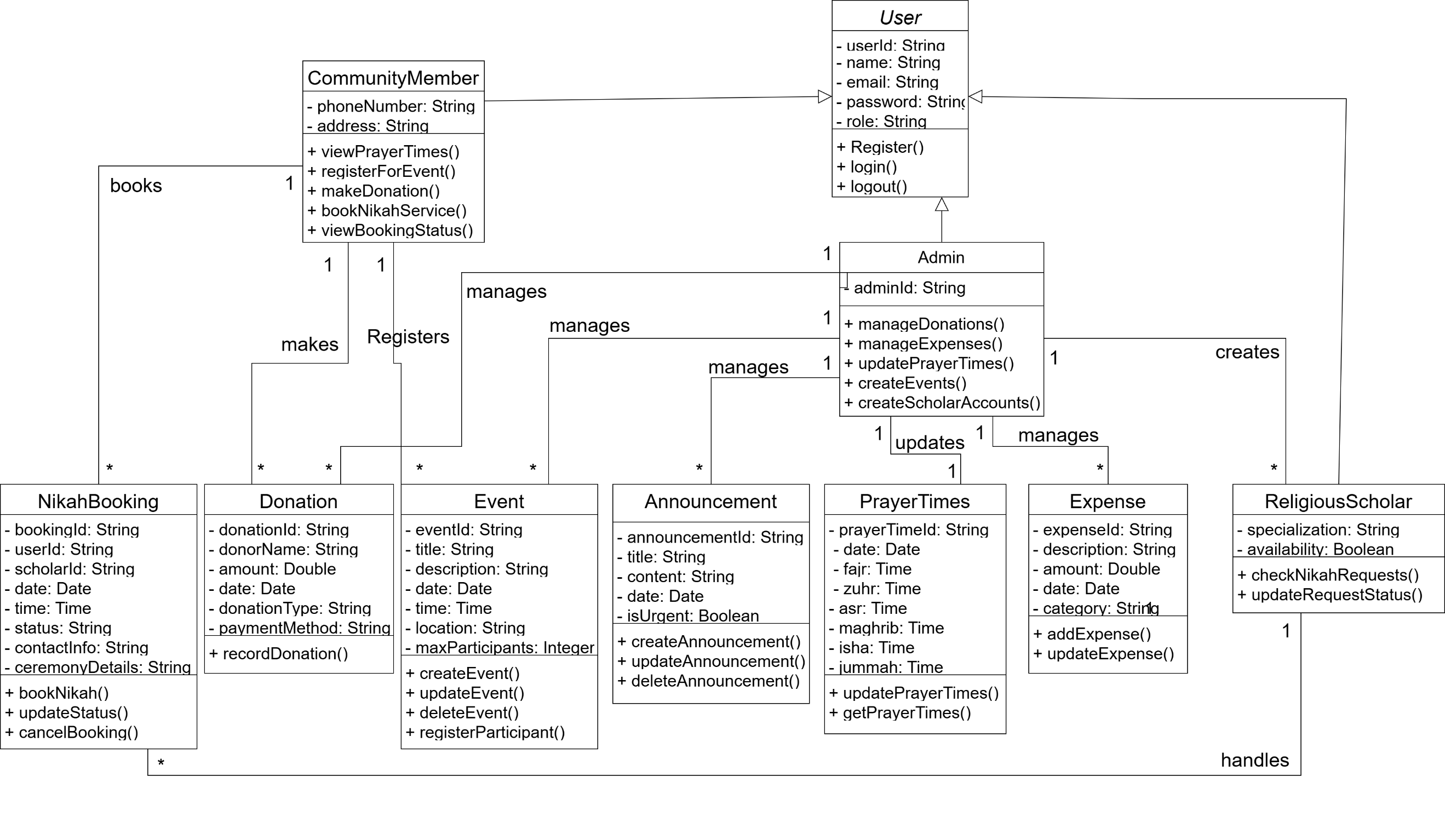


Figure Class diagram

## Sequence Diagram

**User Login Sequence Diagram**

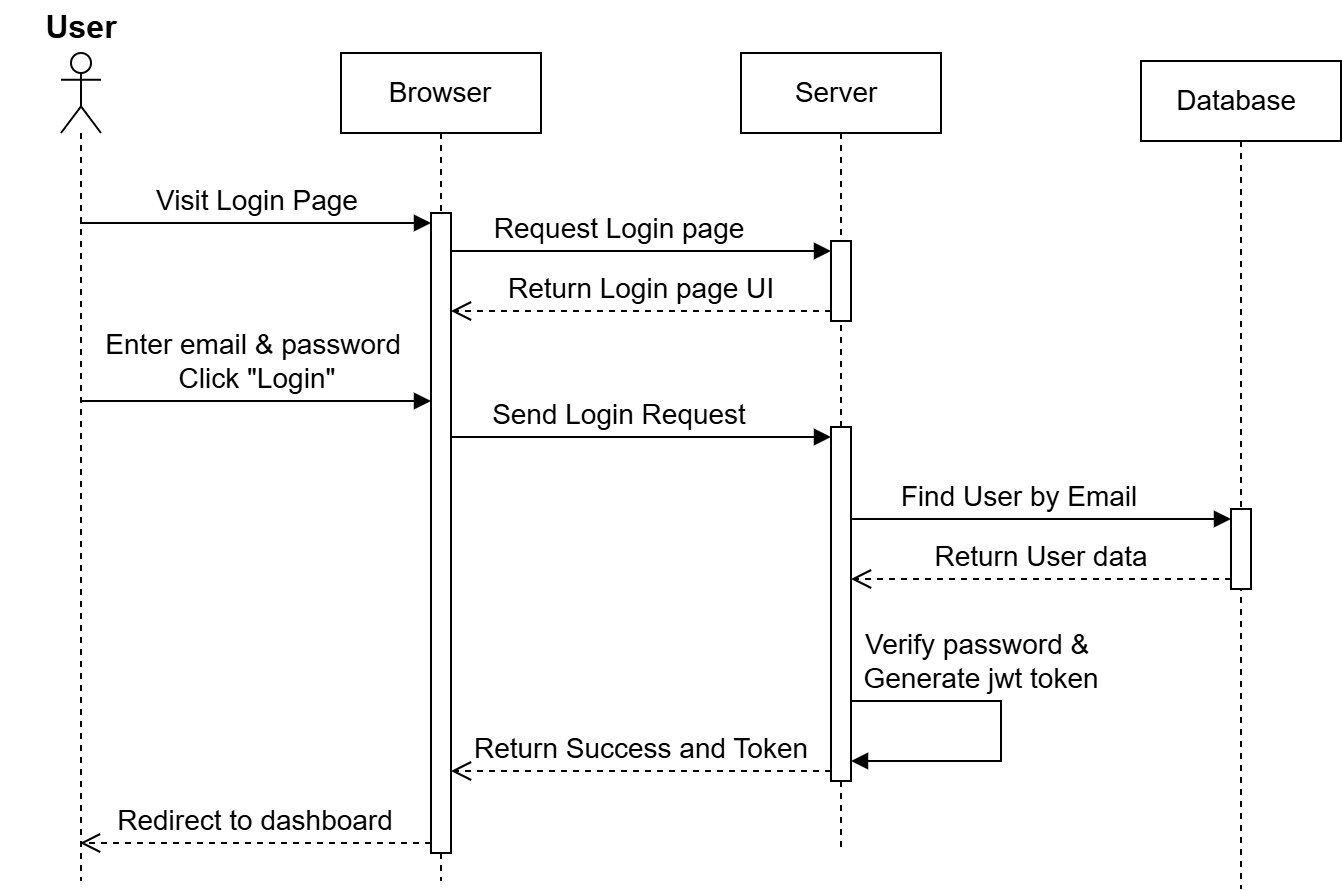


Figure Login Sequence Diagram

This diagram shows the steps that happen when a user logs into the E-Masjid system. It explains how the system checks the user's details to ensure security.  
**Key Steps:**

1. User enters email and password, clicks "Login".
2. Browser sends details to server.
3. Server checks database for the email.
4. Database finds user and sends info back.
5. Server checks if password is correct.
6. If correct, server creates security token.
7. Token sent to browser, user taken to dashboard.
8. If wrong, error message shown to use.

**Online Donation Sequence Diagram**

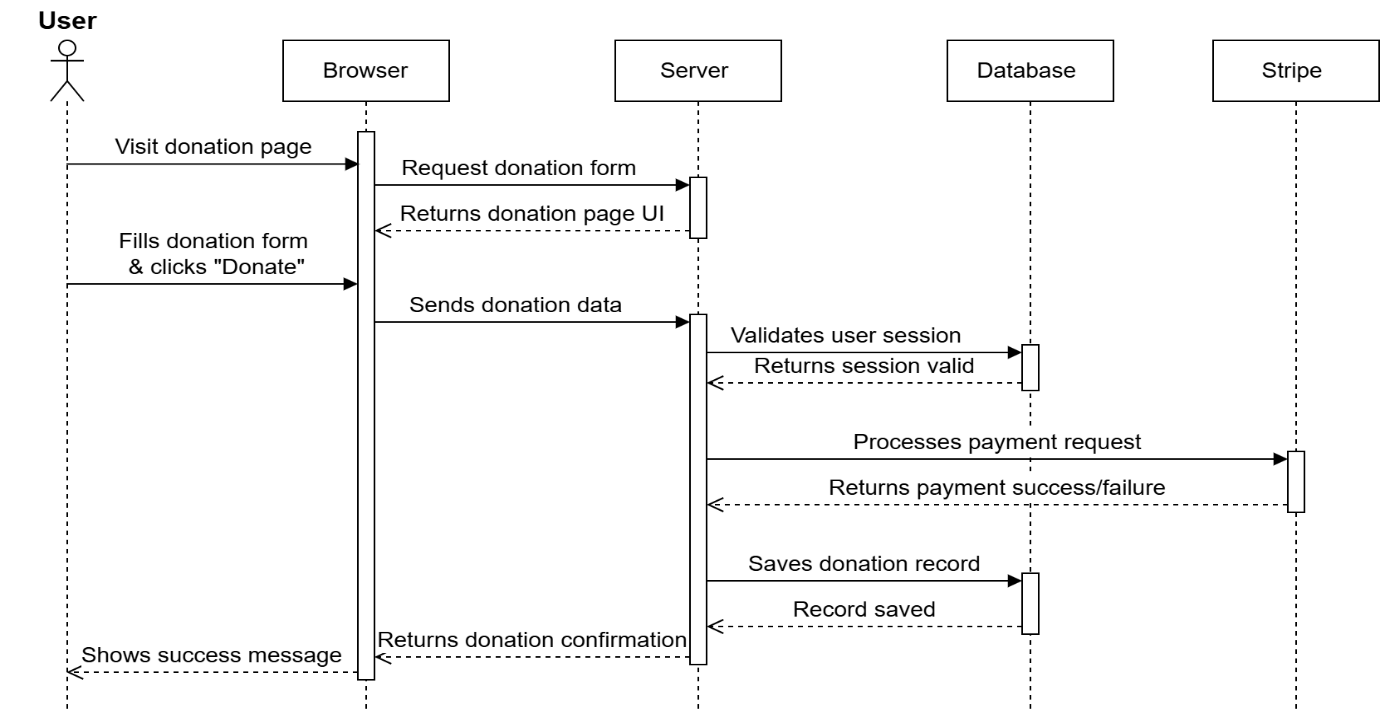


Figure Online Donation Sequence Diagram

This diagram shows the complete flow when a community member makes an online donation. The process involves the user interface, backend server, database, and Stripe payment gateway working together to process the payment and record the transaction.

**Key Steps:**

1. User submits donation form.
2. Backend validates and processes payment via Stripe.
3. Payment result recorded in database and confirmation displayed to user.

**Nikah Booking Sequence Diagram**

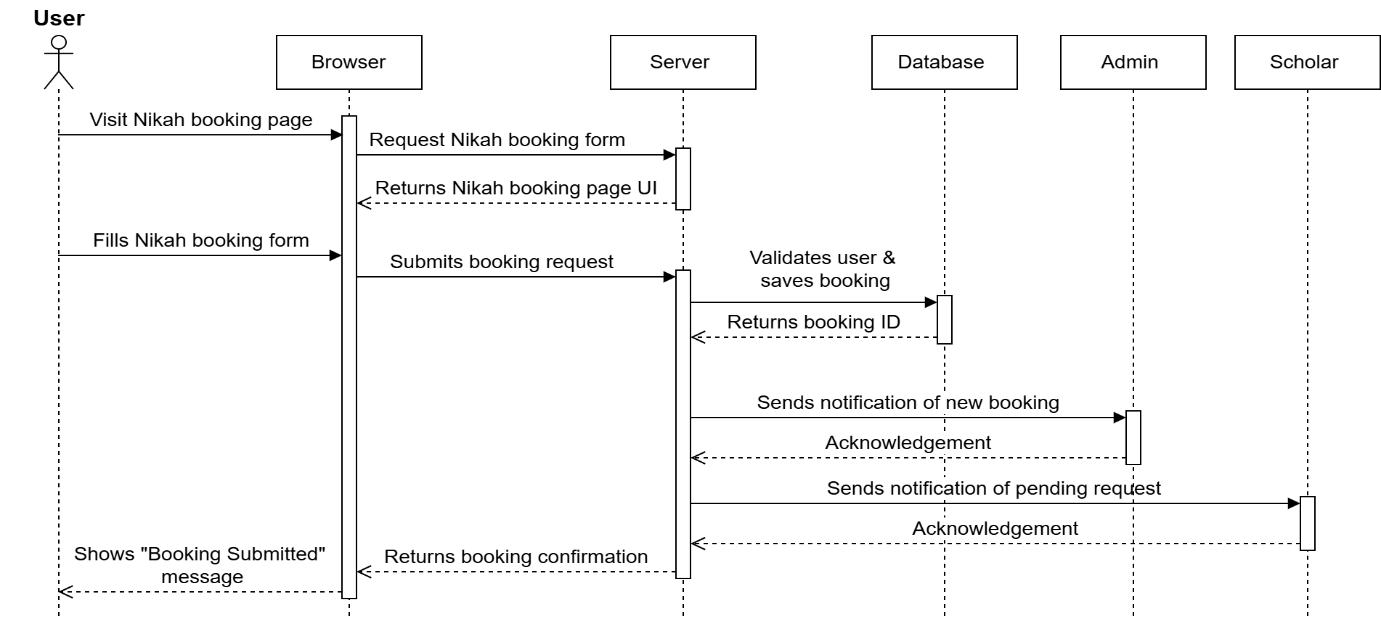


Figure Nikah Booking Sequence Diagram

This diagram display all the steps of the process when a user books Nikah services. The booking request goes through validation, storage, and notification to both admin and religious scholar.

**Key Steps:**

1. User submits booking request.
2. System validates and stores the request.
3. Notifications sent to admin and scholar.
4. User receives booking confirmation.
5. Scholar can later update booking status.

**Prayer Times Update Sequence Diagram**

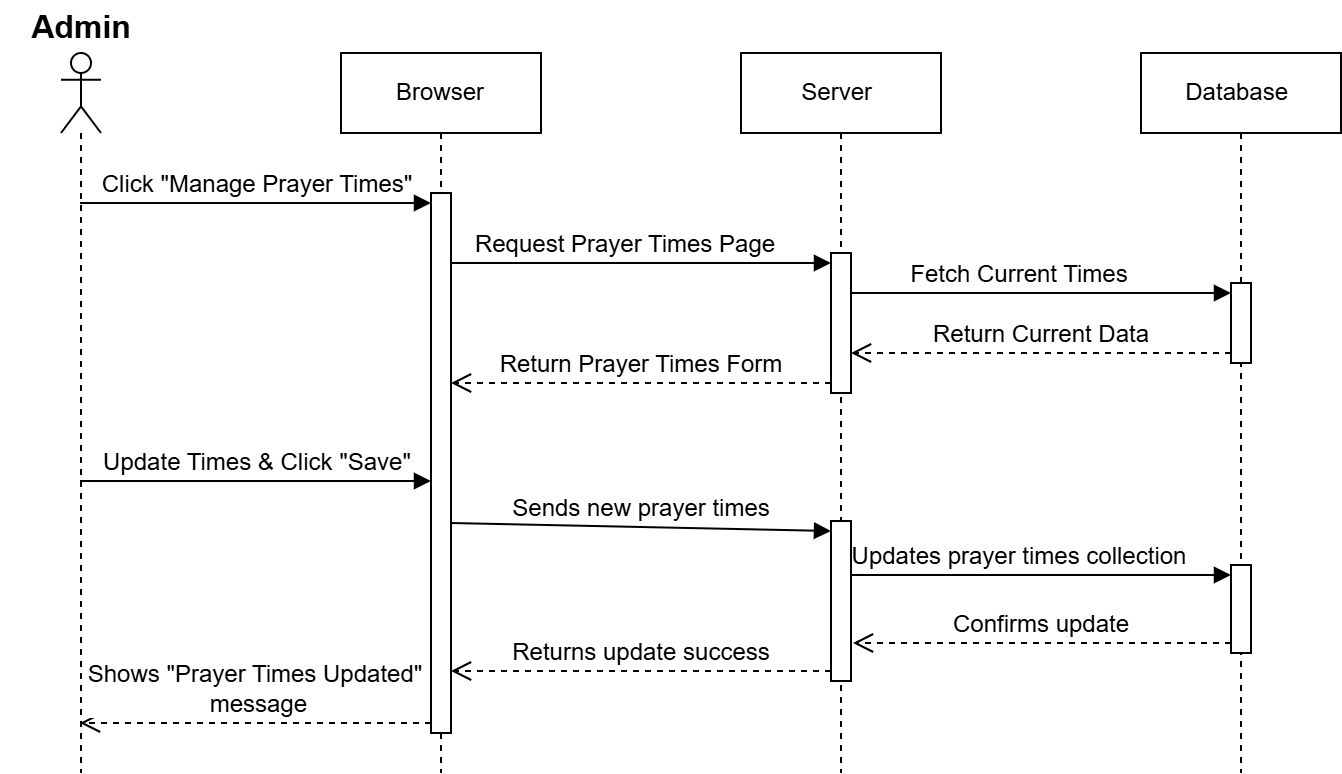


Figure Admin Update Prayer Times Sequence Diagram

This diagram shows how mosque admin updates prayer times.

**Key Steps:**

1. Admin updates times.
2. Changes saved to database.
3. Admin receives confirmation.

## State Transition Diagram

These diagrams show how the system status changes when a user performs different actions.

They show how entities transition between states in response to user actions and system events.

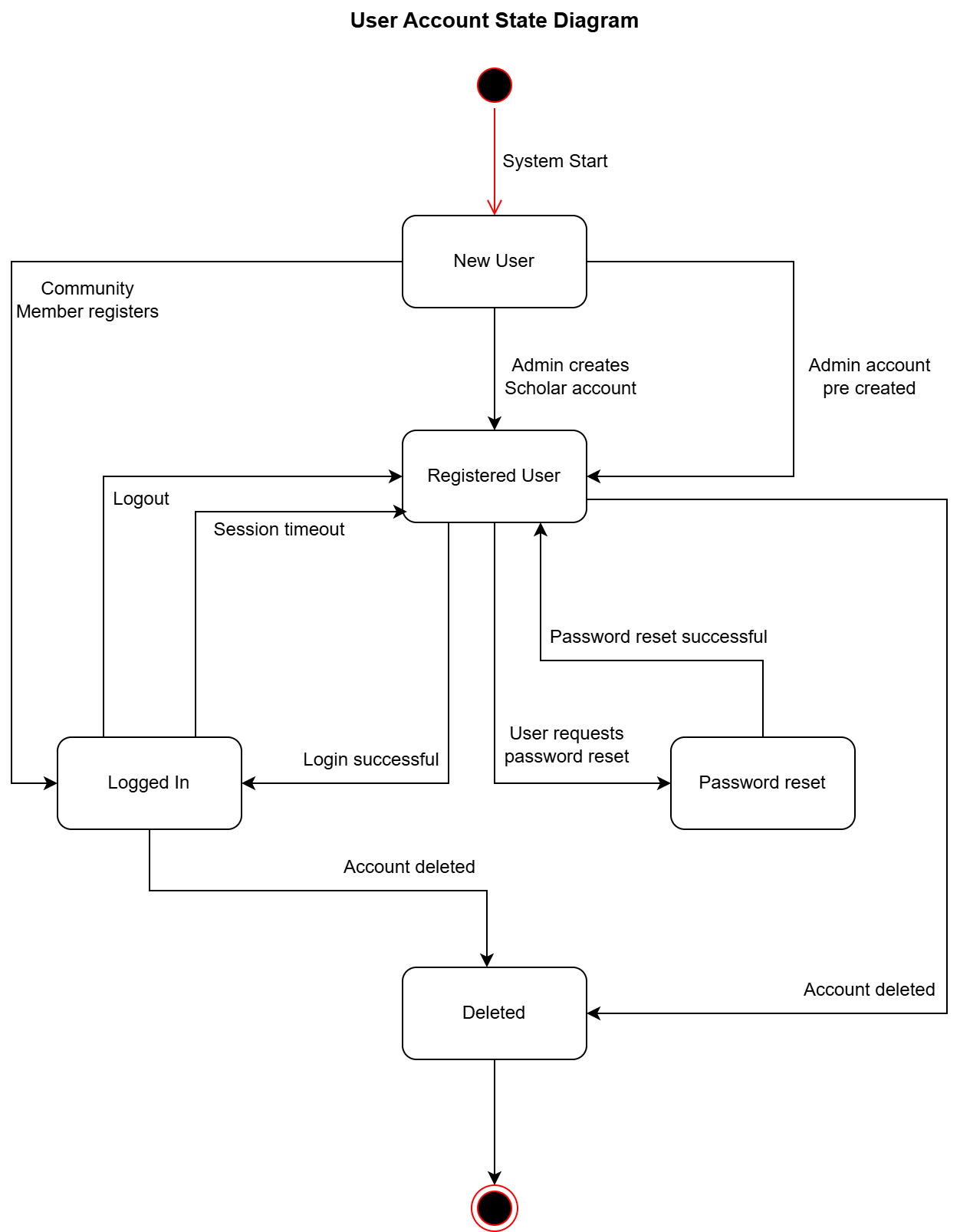


Figure User Account State Diagram

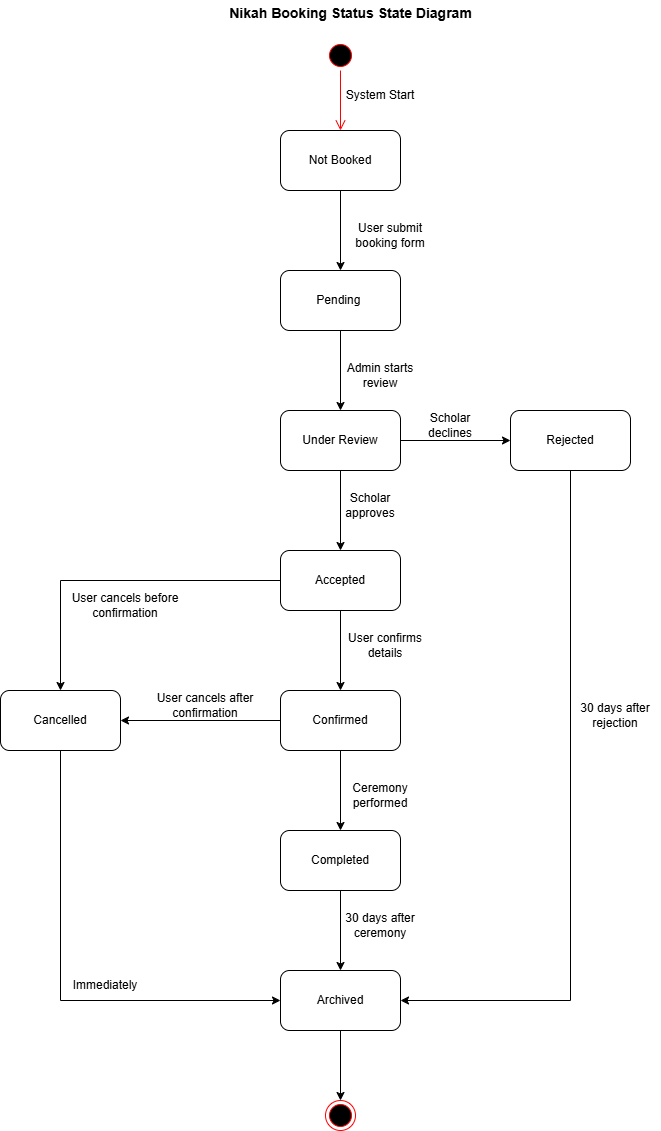


Figure Nikah Booking Status State Diagram



Figure Event State Diagram

## Architectural Design

Our E-Masjid System follows the MVC architecture pattern. This separates the system into three main parts.

1. **Model:** Handles data and business logic.
2. **View:** User interface that users see.
3. **Controller:** Processes user requests and connect Model and View.

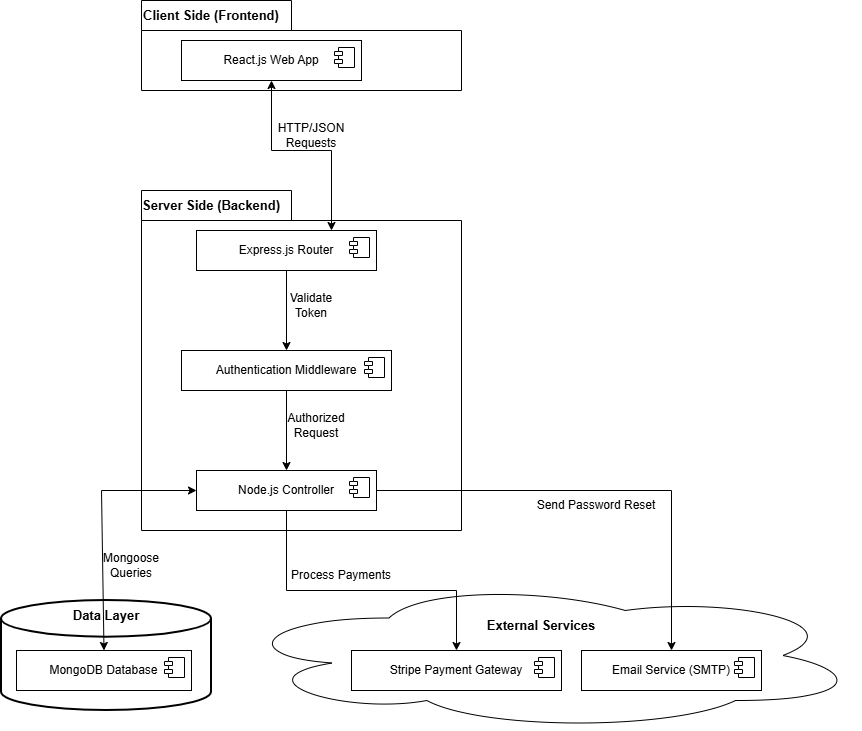


Figure High-Level System Architecture

**System Architecture Components**

**Frontend Layer:**

1. User Interface Components.
2. Mobile-responsive design.
3. Client side validation.
4. Real-time updates.

**Backend Layer:**

1. API routes and controllers.
2. Business logic processing.
3. Authentication and authorization.
4. Payment processing.

**Data Layer:**

1. User data and profiles.
2. Donation and expense records.
3. Prayer time schedules.
4. Event and announcement data.
5. Nikah booking requests.

**External Services:**

1. Stripe Payment Gateway.
2. Email Service for password resets.

## UML Component diagram

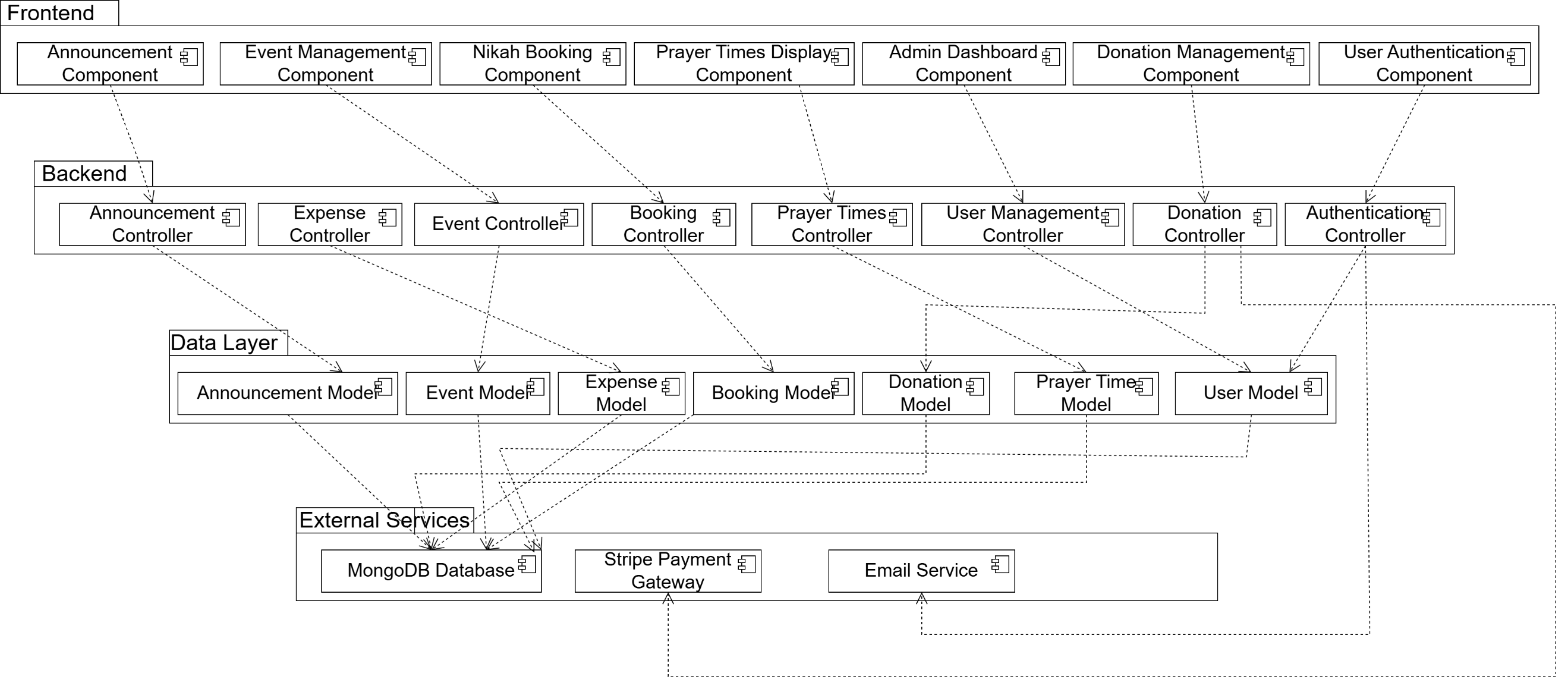


Figure Component diagram

# Data Design

Our system uses MongoDB database to store information. We have separate collections for different things like users, donations, and events. Each collection keeps related information together. Admin and religious scholars are both in the users collection, with different role values.

**Database Collections Structure:**

1. **Users Collection**
2. Stores all user information including admin, community members, and religious scholars.
3. Uses role based access control.
4. Fields: userId, name, email, password, role, phone, address, specialization.
5. **Donations Collection**
6. Records all donation transactions.
7. Links to donor information for transparency.
8. Fields: donationId, donorId, amount, date, type , paymentMethod.
9. **Expenses Collection**
10. Tracks mosque expenditure for financial transparency.
11. Categorized expenses for better reporting.
12. Fields: expenseId, description, amount, date, category.
13. **Events Collection**
14. Manages mosque events and programs.
15. Supports online registration.
16. Fields: eventId, title, description, date, time, location, maxParticipants, registeredUsers[ ].
17. **Announcements Collection**
18. Stores important mosque announcements.
19. Supports urgent flag for important updates.
20. Fields: announcementId, title, content, date, isUrgent, publishedBy.
21. **Prayer Times Collection**
22. Stores daily prayer schedules.
23. Special entries for Jummah and Ramadan.
24. Fields: prayerTimeId, date, fajr, zuhr, asr, maghrib, isha, jummah, isSpecial.
25. **Nikah Bookings Collection**
26. Manages marriage service requests.
27. Tracks booking status.
28. Fields: bookingId, userId, scholarId, date, time, status, contactInfo, ceremonyDetails.

**Data Relationships:**

1. One-to-many: One user can make multiple donations.
2. One-to-many: One admin can create multiple events.
3. Many-to-many: Many users can register for many events.
4. One-to-one: Each day has one prayer time schedule.

## Logical Data Model

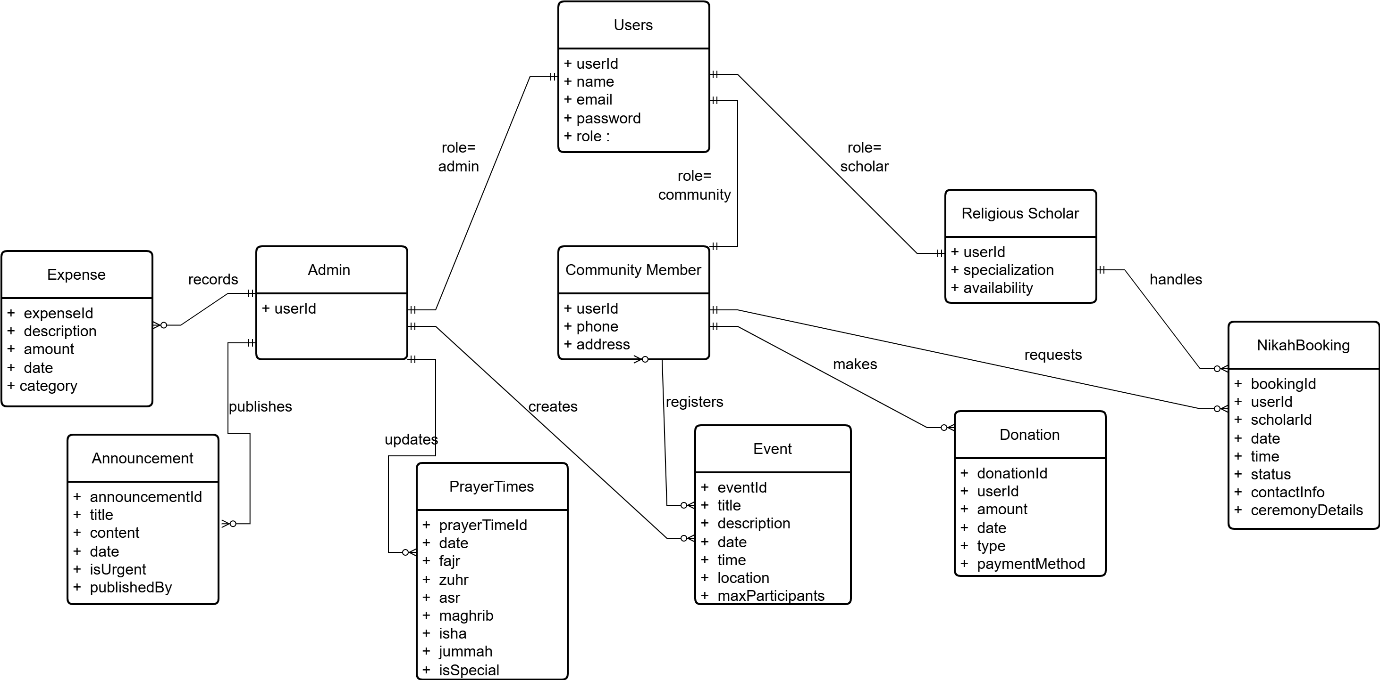


Figure Logical Data Model Diagram

## Data Dictionary

Table Data Dictionary Table

|  |  |
| --- | --- |
| **Terminology** | **Description** |
| Users Collection | Stores all system user accounts |
| userId | String, Primary key for user identification |
| name | String, Full name of the user |
| email | String, User email address |
| password | String, Encrypted password for security |
| role | String, User role (admin/community/scholar) |
| phone | String, Contact phone number. |
| address | String, User residential address |
| specialization | String, For scholars area of expertise |
| Donations Collection | Records all financial donations |
| donationId | String, Unique donation identifier |
| donorId | String, Reference to user who donated |
| Amount | Number, Donation amount in rupees |
| Date | Date, When donation was made |
| Type | String, Donation type (Zakat/Sadaqah/Mosque Fund) |
| paymentMethod | String, Cash or Card payment |
| Expenses Collection | Tracks mosque spending |
| expenseId | String, Unique expense identifier |
| Description | String, What the money was spent on |
| Amount | Number, Expense amount in rupees |
| Date | Date, When expense occurred |
| Category | String, Expense category |
| Events Collection | Manages mosque events |
| eventId | String, Unique event identifier |
| Title | String, Event name/title |
| Description | String, Detailed event information |
| Date | Date, Event date |
| Time | Time, Event time |
| Location | String, Where event will be held |
| maxParticipants | Number, Maximum allowed attendees |
| registeredUsers | Array, List of user IDs who registered |
| Announcements Collection | Stores mosque announcements |
| announcementId | String, Unique announcement identifier |
| Title | String, Announcement headline |
| Content | String, Full announcement text |
| Date | Date, When announcement was posted |
| isUrgent | Boolean, Marks urgent announcements |
| publishedBy | String, Admin who posted the announcement |
| PrayerTimes Collection | Stores prayer schedules |
| prayerTimeId | String, Unique prayer time identifier |
| Date | Date, Date for prayer times |
| Fajr | Time, Fajr prayer time |
| Zuhr | Time, Zuhr prayer time |
| Asr | Time, Asr prayer time |
| Maghrib | Time, Maghrib prayer time |
| Isha | Time, Isha prayer time |
| Jummah | Time, Jummah prayer time |
| NikahBookingsCollection | Manages marriage service requests |
| bookingId | String, Unique booking identifier |
| userId | String, Reference to user who booked |
| scholarId | String, Reference to assigned scholar |
| Date | Date, Requested ceremony date |
| Time | Time, Requested ceremony time |
| Status | String, Current status (Pending/Accepted/Rejected) |
| contactInfo | String, User contact details for ceremony |
| ceremonyDetails | String, Additional ceremony information |

# User Interface Design

The E-Masjid System will have a clean, simple, and easy to use interface designed for all

types of users, including elderly people who may not be comfortable with complex technology.

**User Experience**

1. **Homepage:** Shows current prayer times, recent announcements, and quick access buttons for main features.
2. **Navigation:** Simple menu at the top with clear labels.
3. **Mobile Friendly:** All screens work perfectly on mobile phones and tablets.
4. **Elderly Friendly:** Large buttons, clear text, and simple forms.

**How Different Users Will Use the System:**

For Community Members:

1. Prayer times always visible on top of home page.
2. Simple donation form with card payment option.
3. One click registration for events.
4. Easy booking form for nikah.
5. View booking status in mybooking page.

For Mosque Admin:

1. Special admin panel accessible through /admin URL.
2. Simple forms to add/update donations, expenses, events, announcements.
3. Financial reports showing income and expenses.
4. Create and manage religious scholar accounts.

For Religious Scholars:

1. View pending requests in simple list format.
2. One click buttons to accept or reject bookings.
3. See their booked ceremonies in calendar view.

Feedback and Messages:

1. Green popup messages for successful actions.
2. Red popup messages with simple explanations.
3. "Are you sure?" prompts for important actions.

## Screen Images

We have created basic screen designs to show how the interface will look. These designs follow our guidelines of simplicity and ease of use.

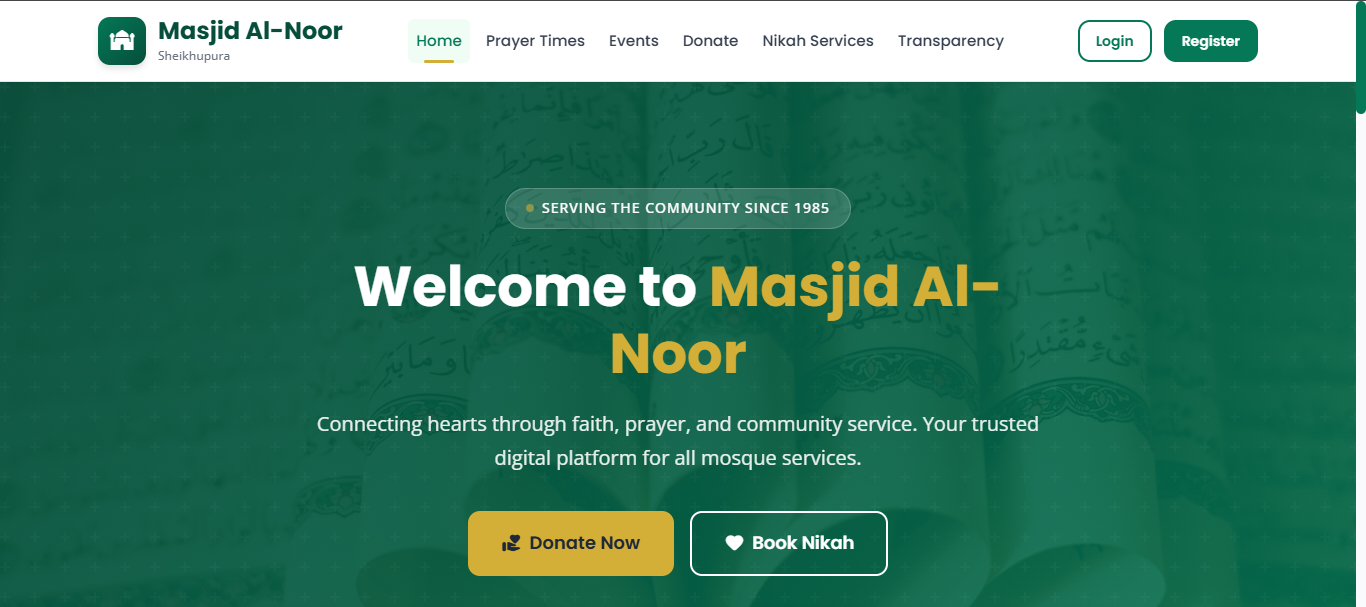


Figure Home page design

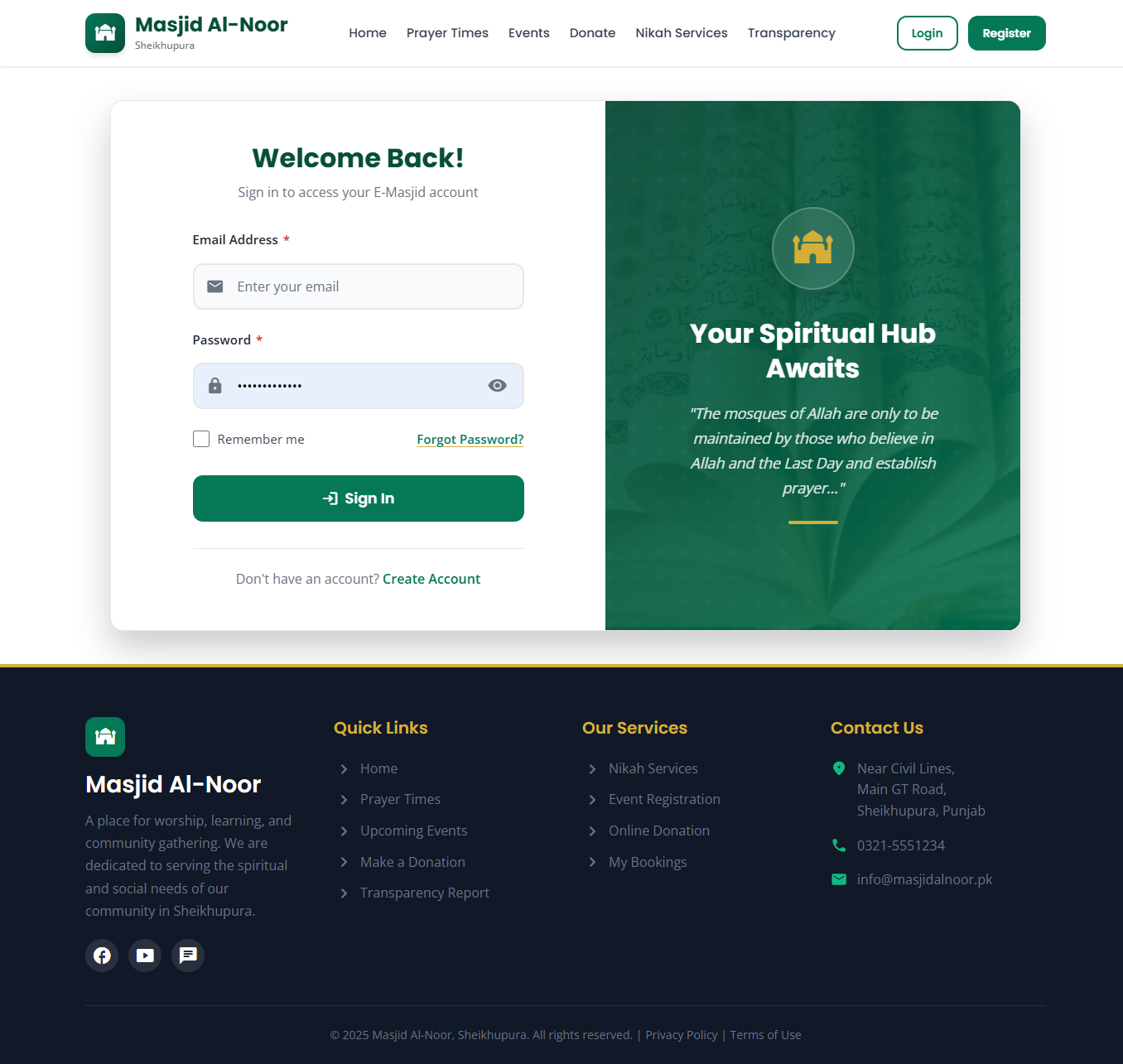


Figure Login page design

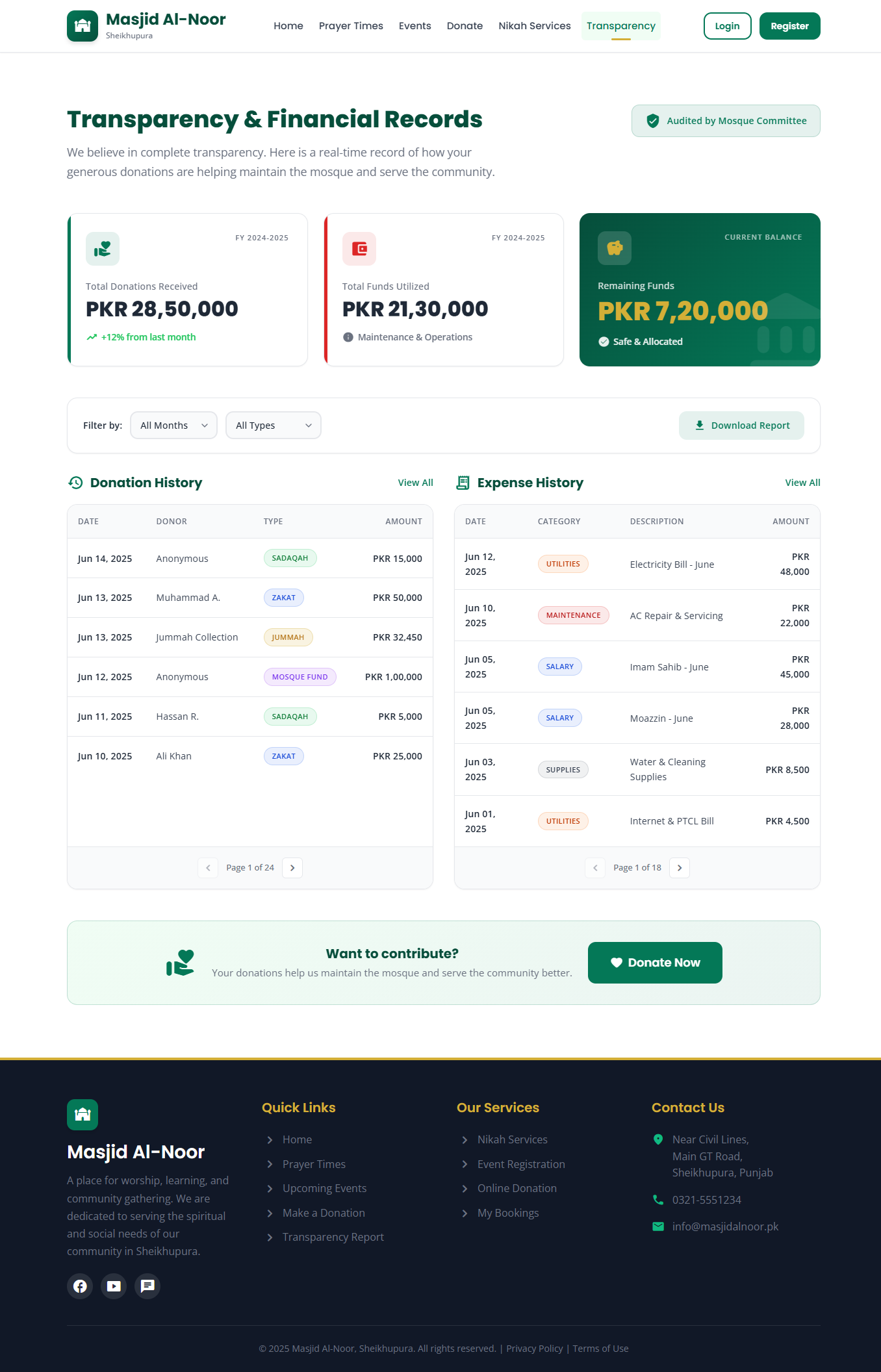


Figure Donation Transparency page design

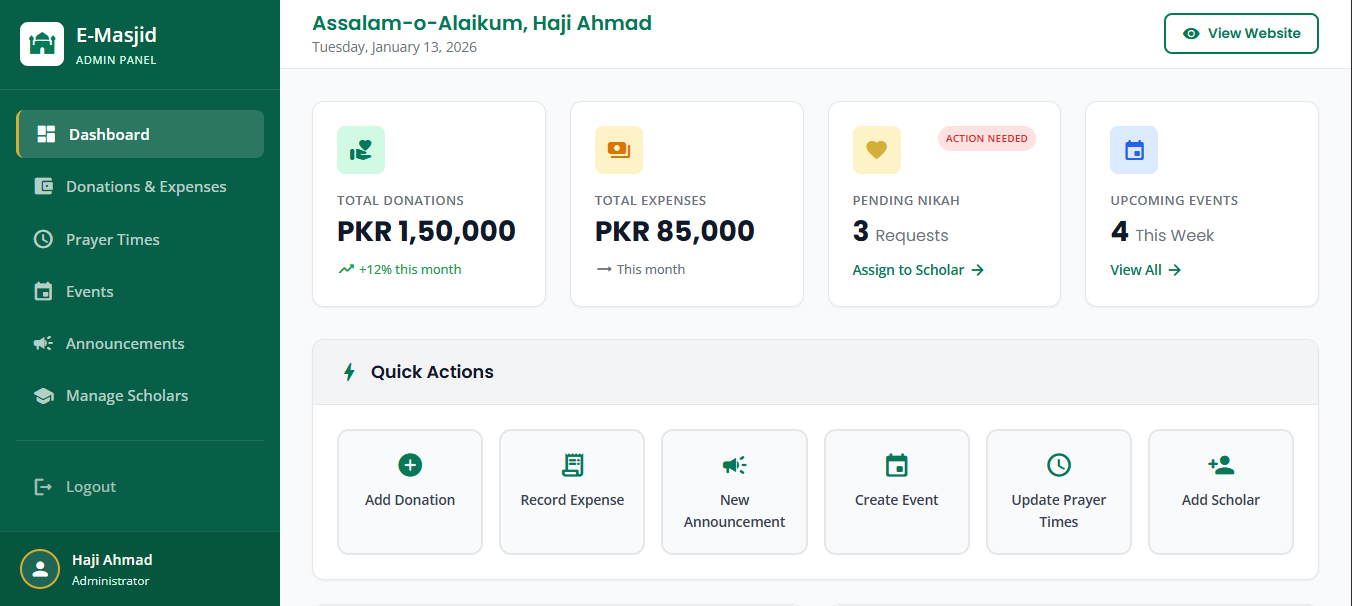


Figure Admin Dashboard design

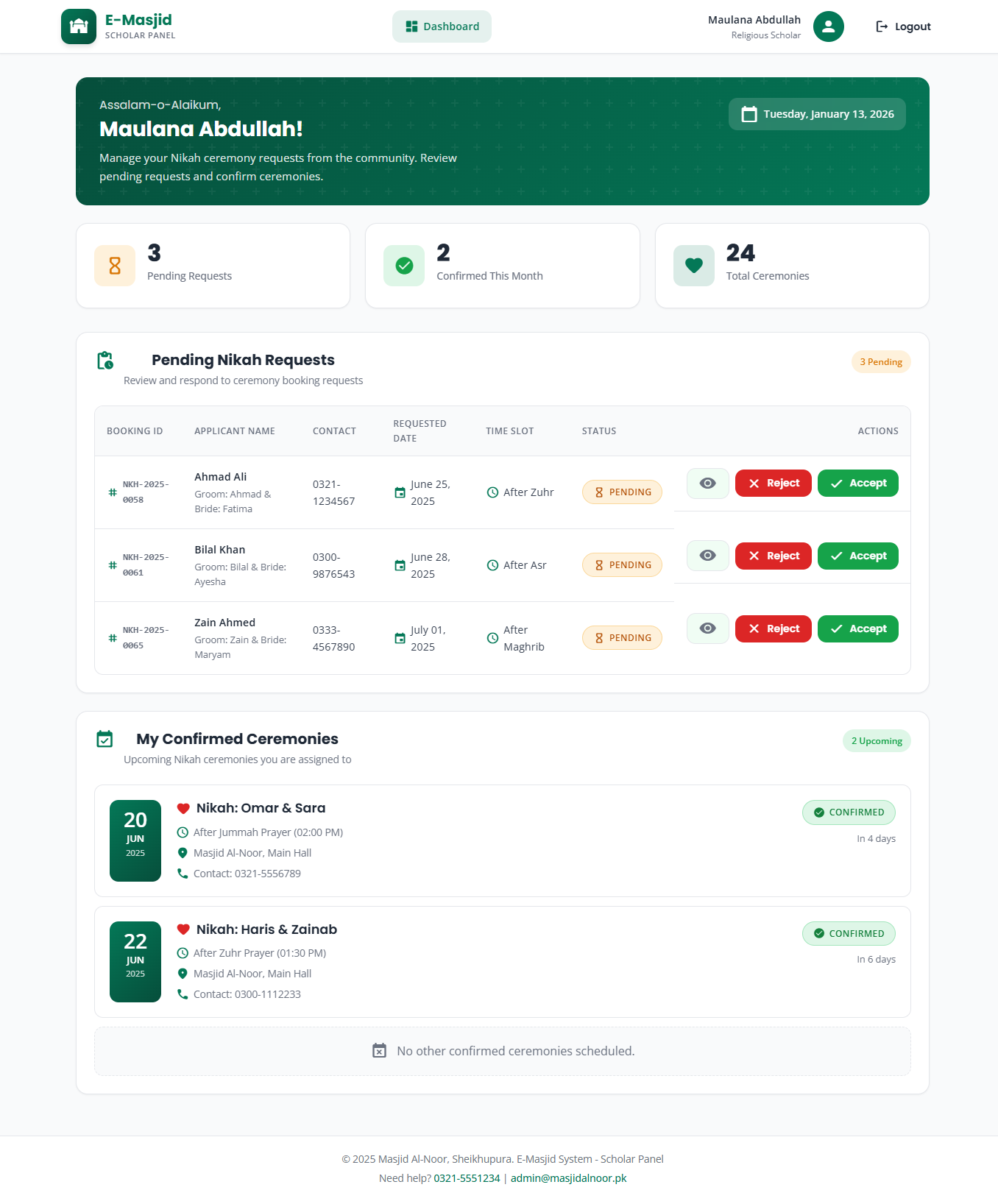


Figure Scholar page design

## Screen Objects and Actions

**Use Case 1: Making an Online Donation**

**Screen Objects:**

1. **Donation Amount Field**: Text box to enter donation amount.
2. **Donation Type Dropdown**: Select Zakat, Sadaqah, or Mosque Fund.
3. **Card Details Form**: Fields for card number, expiry, CVC.
4. **Donate Now Button**: Green button to submit donation.

**Actions:**

1. **User enters amount**: System validates it's a positive number.
2. **User selects donation type**: System shows description of that type.
3. **User enters card details**: System validates card format.
4. **User clicks Donate Now**: System processes payment via Stripe.
5. **Payment successful**: Shows green "Donation Successful" message on the dashboard.
6. **Payment failed**: Shows red "Payment Failed" message with retry option.

**Use Case 2: Admin Creating an Event**

**Screen Objects:**

1. **Event Title Field**: Text box for event name.
2. **Date and Time Pickers**: Calendar and time selectors.
3. **Description Box**: Large text area for event details.
4. **Location Field**: Text box for event location.
5. **Max Participants Field**: Number field for attendance limit.
6. **Publish Button**: Blue button to publish event.
7. **Save Draft Button**: Gray button to save for later.

**Actions:**

1. **Admin enters event details**: System validates all required fields.
2. **Admin sets date/time**: System checks date is not in past.
3. **Admin sets max participants**: System validates positive number.
4. **Admin clicks Publish**: System creates event and shows on website.
5. **Event published**: Shows green "Event Published Successfully".
6. **Validation error**: Shows red message next to incorrect field.

# Design Decisions

This section explains the main design choices we made for the E-Masjid System and why we chose them.

**Architecture Pattern Selection**

We decided to use the MVC pattern for our system. This means we separate our code into three main parts which is model, view and controller. This makes our code cleaner and easier to fix if something goes wrong.

**Technology Stack MERN**

We are using the MERN stack for our project. We chose this because all parts use JavaScript, which makes development faster. React helps us create reusable components, and Node.js can handle many users at once.

**Database System**

We picked MongoDB instead of traditional SQL databases because it works naturally with JavaScript and Node.js. It can handle lots of data when many people use the system and it is free to use for our project size. We tried MySQL first but found MongoDB easier for our type of data.

**Pay Gateway Integration**

We chose Stripe for online donations because it is very secure and handles card details safely and many other projects use it successfully. Stripe has good documentation that helped us learn and it is reliable and processes payments quickly.

**Deployment Strategy**

We plan to deploy the system on cloud services because it is more reliable than our own computers and can handle more users during busy times. It automatically backs up data and it is affordable for a mosque budget.

# Summary

This Software Design Specification show the complete technical plan for our E-Masjid System. We have explained the MVC architecture, MERN technology, database design, and user interfaces. The design describes all functional requirements and provides clear guidance for development. Following this document we will ensure to build a reliable system that meets the mosque needs.

# References

References

## Article in a Journal

[1] M. Ahmad, A. Hassan, and S. Khan, "Trust and Transparency in Religious Charitable Organizations in Pakistan," *Journal of Islamic Management Studies*, vol. 12, no. 3, pp. 45-62, 2019.

[2] F. Khan, "Digital Transformation of Religious Institutions: A Case Study of Mosques in Urban Pakistan," *Pakistan Journal of Information Technology*, vol. 8, no. 2, pp. 112-128, 2020.

## World Wide Web

[3] Meta Platforms Inc., "React Documentation," *React*. [Online]. Available: <https://react.dev/>. [Accessed: Sep 18, 2025].

[4] OpenJS Foundation, "Node.js Documentation," *Node.js*. [Online]. Available: <https://nodejs.org/en/docs/>. [Accessed: Oct. 15, 2025].

[5] MongoDB Inc., "MongoDB Manual," *MongoDB*. [Online]. Available: <https://www.mongodb.com/docs/manual/>. [Accessed: Nov 1, 2025].

[6] Automattic Inc., "Mongoose Documentation," *Mongoose*. [Online]. Available: <https://mongoosejs.com/docs/>. [Accessed: Nov 10, 2025].

[7] Stripe Inc., "Stripe API Reference," *Stripe*. [Online]. Available: <https://stripe.com/docs/api>. [Accessed: Dec 17, 2025].

[8] Auth0, "Introduction to JSON Web Tokens," *JWT.io*. [Online]. Available: <https://jwt.io/introduction>. [Accessed: Jan. 02, 2026].

[9] [Draw.io](https://draw.io/) User Manual for UML Diagrams. [Online]. Available: <https://www.drawio.com/>. [Accessed: Dec 2025].