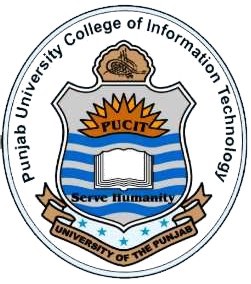
# Donation Centre and Orphanage Location Website

# (Safe Haven)

**Final Year Project Report**



by

|  |  |
| --- | --- |
|  |  |
| Sofia Fazal | 069606 |
| Noor-ul-Huda | 069565 |
| Ume Umaira | 069605 |

Session: BSIT 2021 – 2025

Project Advisor: Prof Abu Zar Tamimi

Bachelor of Science in Information Technology

(2021-2025)

Faculty of Computing & Information Technology

University of the Punjab, Lahore, Pakistan

**Donation Centre and Orphanage Location Website**

**(Safe Haven)**

A project presented to

**University of the Punjab, Lahore**

In partial fulfilment of the requirement for the degree of

**Bachelors of Science in Information Technology(2021-2025)**

by

|  |  |
| --- | --- |
|  |  |
| Sofia Fazal | 069606 |
| Noor-ul-Huda | 069565 |
| Ume Umaira | 069605 |

Faculty of Computing & Information Technology

University of the Punjab, Lahore, Pakistan

# Declaration

We hereby declare that this software, neither whole nor as a part, has been copied from any source. It is further declared that we have developed this software and accompanying report entirely based on our efforts. If any part of this project is proven to be copied from any source or found to be a reproduction of some other work, we will stand by the consequences. No portion of the work presented has been submitted to any application for any other degree or qualification of this or any other university or institute of learning.

Signature: --------------------------- Signature: ------------------------

Sofia Fazal Noor-ul-Huda

069606 069565

Signature: ---------------------------

Ume Umaira

069605

# Certificate Of Approval

It is to certify that the final year design project (FYDP) of BSIT “Safe Haven” was developed by Sofia Fazal**(BXXF21Mxxx),** Noor-ul-Huda**(BXXF21Mxxx),** andUme Umaira**(BXXF21Mxxx)** under the supervision of Prof Abu Zar in my opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Information Technology.

Signature: ---------------------------------------

**FYDP Supervisor:**

**Signatures (Faculty Advisory Committee (FAC)**

|  |  |  |  |
| --- | --- | --- | --- |
| Signatures |  |  |  |
| Name |  |  |  |
|  | FAC1 | FAC2 | FAC3 |

Signature: ---------------------------------------

**Head of FYDP Coordination Office:**

Signature:--------------------------------------- Dated: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Chairperson, Department of Information Technology**

# Donation Centre and Orphanage Location Website

# (Safe Haven)

# Executive Summary

The Safe Haven project is a web-based platform designed to connect orphanages with donors, facilitating transparent donations, efficient orphanage management, and seamless communication between stakeholders. This project proposal outlines the system's objectives, including an intuitive interface, real-time orphanage search and secure donation tracking.

The Software Requirements Specification (SRS) defines the system’s functional and non-functional requirements, covering key features such as user authentication, orphanage listing, donation management, gamification elements, and an admin dashboard for monitoring and moderation.

The Software Design Specification (SDS) also provides a detailed technical structure, including system architecture, database design, and third-party API integrations like Google Maps for location tracking and EasyPaisa for payment processing.

The Safe Haven prototype serves as a preliminary visualization of the system, showcasing interactive user interfaces, workflows, and essential features to validate usability and functionality. The requirements and use case analysis ensure a user-centric approach, aligning system capabilities with real-world needs.

By integrating accessibility features and security measures, the platform aims to create an inclusive and trustworthy environment. This document collectively establishes a foundation for the structured development and deployment of Safe Haven, ensuring scalability, reliability, and efficiency in connecting orphanages with potential supporters.

**Keywords**: Centralized platform, Donation management, Orphanage visibility, Web-based solution, User-friendly interface, Real-time updates, Secure transactions, Transparency and trust, Location-based search, Donor engagement, Community support, Impact tracking, Digital transformation, Accessible design, Verified orphanages, Automated notifications, Interactive dashboards, Social responsibility

# Acknowledgement

I would like to express my sincere gratitude to everyone who supported and guided me throughout the development of the *Safe Haven* project. I am especially thankful to my supervisor for their continuous encouragement, valuable insights, and constructive feedback that shaped the direction of this work. I also extend my appreciation to my team members for their collaboration and dedication. Lastly, I am grateful to my family and friends for their constant support and motivation during the entire process.

Signature: --------------------------- Signature: -----------------------

Sofia Fazal Noor-ul-Huda

069606 069565

Signature: ---------------------------

Ume Umaira

069605

FYDP OVERVIEW

**FYDP Title: SAFE HAVEN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Roll Numbers** | **Name** | **Signatures** |
| 1. | 069606 | Sofia Fazal |  |
| 2. | 069565 | Noor-ul-Huda |  |
| 3. | 069605 | Ume Umaira |  |

Table 1 Project Proposal Summary

|  |  |
| --- | --- |
| **FYDP Goals** | * Create a Centralized Platform * Enhance Transparency * Simplify the Donation Process * Improve Resource Allocation * Increase Donor Engagement * Implement Secure Access * Support Scalability * Encourage Social Responsibility * Support Multiple Orphanage Needs |
| **FYDP Objectives** | * Develop a centralized web platform for orphanage support. * Simplify the donation process. * Ensure transparency in fund utilization and orphanage needs. * Increase the visibility of orphanages through detailed profiles. * Enhance community engagement through awareness campaigns. * Facilitate long-term sponsorship opportunities for organizations. * Introduce gamification features to motivate user participation. * Offer a subscription-based donation model for sustained support. * Integrate educational resources on adoption and child welfare * Support diverse donation types (money, food, clothing, medical aid). * Strengthen data security and privacy for all users * Expand the platform’s reach to connect orphanages globally. |
| **FYDP**  **Success**  **Criteria** | * Fully Functional Web Platform * User-friendly Interface * Accurate & UpToDate Information * Secure User Authentication * Seamless Donation Process * Increased Donor Participation * Successful Resource Allocation * Positive User Feedback * Scalability & Future Expansion |
| **Assumptions:** | * Availability of Reliable Data * Donors Are Willing to Use the Platform * Stable Internet Connectivity * Orphanages Have Basic Digital Literacy * Users Trust the Platform * Scalability Is Feasible * Secure Transactions * Legal and Ethical Compliance * Minimal Maintenance Effort * Collaboration with Orphanages |
| **Risks**  **& Obstacles** | * Fraudulent Activities and Misuse of Funds * Difficulty in Verifying Orphanage Legitimacy * Legal and Compliance Challenges * Scalability and Performance Challenges * Resistance from Orphanages * Complexity in Donation and Resource Allocation * Internet Connectivity Issues in Remote Areas * Technical Issues and System Downtime * Security and Privacy Concerns * Low User Engagement |
| **Organisation Address:** | Government Jinnah Islamia Graduate College Sialkot |
| **Target End Users:** | Orphanages, donors, donation centers |
| **Suggested Project Supervisor** | Prof. Muhammad Abu Zar Tamimi |
| **Approved By:** |  |
| **Date:** |  |

# Abbreviations

Table 2 abbreviations

|  |  |
| --- | --- |
| **API**: | Application Programming Interface |
| **BSIT**: | Bachelor of Science in Information Technology |
| **CSS**: | Cascading Style Sheets |
| **DFD**: | Data Flow Diagram |
| **DCD**: | Design Class Diagram |
| **ERD**: | Entity-Relationship Diagram |
| **FAQ**: | Frequently Asked Questions |
| **FCM**: | Firebase Cloud Messaging |
| **FK**: | Foreign Key |
| **FR**: | Functional Requirement |
| **FYDP**: | Final Year Design Project |
| **GDPR**: | General Data Protection Regulation |
| **GPS**: | Global Positioning System |
| **HTML**: | HyperText Markup Language |
| **HTTP**: | Hypertext Transfer Protocol |
| **IEEE**: | Institute of Electrical and Electronics Engineers |
| **JWT**: | JSON Web Tokens |
| **MBTF**: | Mean Time Between Failures |
| //**MERN Stack**: | MongoDB, Express.js, React, Node.js |
| **MTTR**: | Mean Time to Recover |
| **MVC**: | Model-View-Controller |
| **NGO**: | Non-Governmental Organization |
| **NoSQL**: | Not Only SQL |
| **OO**: | Object-Oriented |
| **ORM**: | Object-Relational Mapping |
| **PK**: | Primary Key |
| **RBAC**: | Role-Based Access Control |
| **RESTful APIs**: | Representational State Transfer Application Programming Interfaces |
| **SDS**: | Software Design Specification |
| **SMS**: | Short Message Service |
| **SRS**: | Software Requirements Specification |
| **SSL/TLS**: | Secure Sockets Layer/Transport Layer Security |
| **UI**: | User Interface |
| **UML**: | Unified Modeling Language |
| **UC**: | Use Case |
| **UX**: | User Experience |
| **WCAG**: | Web Content Accessibility Guidelines |

**Table of Contents**

[Donation Centre and Orphanage Location Website 1](#_Toc201665837)

[(Safe Haven) 1](#_Toc201665838)

[Declaration 3](#_Toc201665839)

[Certificate Of Approval 4](#_Toc201665840)

[Donation Centre and Orphanage Location Website 5](#_Toc201665841)

[(Safe Haven) 5](#_Toc201665842)

[Executive Summary 5](#_Toc201665843)

[Acknowledgement 6](#_Toc201665844)

[Abbreviations 9](#_Toc201665845)

[CHAPTER 1 15](#_Toc201665846)

[INTRODUCTION 15](#_Toc201665847)

[1.1 Introduction 16](#_Toc201665848)

[1.2 Problem Statement 16](#_Toc201665849)

[1.3 Problem Solution 16](#_Toc201665850)

[1.4 Objectives 17](#_Toc201665851)

[1.5 Scope of study 17](#_Toc201665852)

[1.6 System Components 18](#_Toc201665853)

[1.7 Related System Analysis/Literature Review 19](#_Toc201665854)

[1.8 System Limitations and Constraints 20](#_Toc201665855)

[1.9 Tools, Libraries and Technologies with Reasoning 21](#_Toc201665856)

[1.10 Project Planning 22](#_Toc201665857)

[1.11 Project Deliverables 22](#_Toc201665858)

[1.12 Summary 23](#_Toc201665859)

[CHAPTER 2 24](#_Toc201665860)

[DESIGN AND ARCHITECTURE 24](#_Toc201665861)

[2.1 Requirements Analysis 25](#_Toc201665862)

[2.2 User classes and characteristics 25](#_Toc201665863)

[2.3 Requirement Identifying Techniques 25](#_Toc201665864)

[2.4 Functional Requirements 26](#_Toc201665865)

[2.5 Non-Functional Requirements 27](#_Toc201665866)

[2.6 External Interface Requirements 28](#_Toc201665867)

[2.6.1 User Interface Requirements 28](#_Toc201665868)

[2.6.2 Software Requirements 28](#_Toc201665869)

[2.6.3 Hardware Interfaces 29](#_Toc201665870)

[2.6.4 Communication Interfaces 29](#_Toc201665871)

[2.7 SUMMARY 29](#_Toc201665872)

[CHAPTER 3 30](#_Toc201665873)

[Design and Architecture 30](#_Toc201665874)

[3.1 System Design 31](#_Toc201665875)

[3.1.1 Dependencies and Interactions with Other Systems 31](#_Toc201665876)

[3.1.2 Design Constraints 31](#_Toc201665877)

[3.2 Design Considerations 32](#_Toc201665878)

[3.2.1 Assumptions 32](#_Toc201665879)

[3.2.2 Limitations 32](#_Toc201665880)

[3.3 Requirement Traceability Matrix 32](#_Toc201665881)

[3.4 Design Models 33](#_Toc201665882)

[3.4.1 Design Class Diagram (DCD) 33](#_Toc201665883)

[3.4.1.1 User Management Class Diagram 33](#_Toc201665884)

[3.4.1.2 Donation Management Class Diagram 34](#_Toc201665885)

[3.4.1.3 Notification System Class Diagram 35](#_Toc201665886)

[3.4.1.4 Orphanage & Request Management Class Diagram 36](#_Toc201665887)

[3.4.1.5 Payment & Transaction Processing Class Diagram 37](#_Toc201665888)

[3.4.2 Interaction Diagram (Sequence diagram) 38](#_Toc201665889)

[3.4.2.1 Sequence Diagram for Each Actor 38](#_Toc201665890)

[3.4.2.1.1 Admin Dashboard Management: 38](#_Toc201665891)

[3.4.2.1.2 Donation Management: 39](#_Toc201665892)

[3.4.2.1.3 User Management: 40](#_Toc201665893)

[3.4.2.1.4 Orphanage Management: 41](#_Toc201665894)

[3.4.3 State Transition Diagram 42](#_Toc201665895)

[3.4.3.1 User authentication 42](#_Toc201665896)

[3.4.3.2 Donation process 43](#_Toc201665897)

[3.4.3.3 Orphanage registration 44](#_Toc201665898)

[3.4.3.4 Donation center Inventory management 45](#_Toc201665899)

[3.4.3.5 Search and filter 46](#_Toc201665900)

[3.5 System Architecture 47](#_Toc201665901)

[3.5.1 Components: 47](#_Toc201665902)

[3.5.2 Interactions: 47](#_Toc201665903)

[3.5.3 System Architecture Diagram 48](#_Toc201665904)

[3.6 Data Design 49](#_Toc201665905)

[3.6.1 Database Design 49](#_Toc201665906)

[3.6.1.1 Data Requirements 49](#_Toc201665907)

[3.6.1.2 Entity and their Attributes 49](#_Toc201665908)

[3.6.1.3 Primary Keys (PK) and Foreign Keys (FK) 50](#_Toc201665909)

[3.6.1.4 Relationships 50](#_Toc201665910)

[3.6.1.5 ER Diagram 51](#_Toc201665911)

[3.6.2 Data Flow Diagram (DFD) 51](#_Toc201665912)

[3.6.3 Data Dictionary 52](#_Toc201665913)

[3.7 User Interface Design 54](#_Toc201665914)

[3.7.1 Components 54](#_Toc201665915)

[3.7.2 Wireframe Of Each Component 56](#_Toc201665916)

[3.7.3 Screen Objects and Actions 62](#_Toc201665917)

[3.8 Design Decisions 63](#_Toc201665918)

[3.9 Summary 63](#_Toc201665919)

[References 64](#_Toc201665920)

[Appendix A 66](#_Toc201665921)

[Use case Description 66](#_Toc201665922)

[Use Case Analysis 66](#_Toc201665923)

[Use Case Diagram: 68](#_Toc201665924)

[Use Case Diagram for Each Actor 69](#_Toc201665925)

**LIST OF TABLES**

[**Table 1 Project Proposal Summary 7**](#_Toc201665985)

[**Table 2 abbreviations 9**](#_Toc201665986)

[**Table 3 related work 19**](#_Toc201665987)

[**Table 4 Tools, Technologies, and Libraries 21**](#_Toc201665988)

[**Table 5 User Classes and Their Characteristics 25**](#_Toc201665989)

[**Table 6 Functional Requirements Specification for User Registration 26**](#_Toc201665990)

[**Table 7 Functional Requirements Specification for User Profile 26**](#_Toc201665991)

[**Table 8 Functional Requirements Specification for Orphanage Registration 26**](#_Toc201665992)

[**Table 9 Functional Requirements Specification for User Orphanage Listing 27**](#_Toc201665993)

[**Table 10 Functional Requirements Specification for Donation Management 27**](#_Toc201665994)

[**Table 11 Requirement Traceability Matrix 32**](#_Toc201665995)

[**Table 12 User Management Class Diagram 33**](#_Toc201665996)

[**Table 13 Donation Management Class Diagram 34**](#_Toc201665997)

[**Table 14 Notification System Class Diagram 35**](#_Toc201665998)

[**Table 15 Orphanage & Request Management Class Diagram 36**](#_Toc201665999)

[**Table 16 Payment & Transaction Processing Class Diagram 37**](#_Toc201666000)

[**Table 17 User authentication 42**](#_Toc201666001)

[**Table 18 Donation process 43**](#_Toc201666002)

[**Table 19 Orphanage registration 44**](#_Toc201666003)

[**Table 20 Donation center Inventory management 45**](#_Toc201666004)

[**Table 21 Search and filter 46**](#_Toc201666005)

[**Table 22 Alphabetical List of System Entities 52**](#_Toc201666006)

[**Table 23 Structured Approach: Functions and Function Parameters 53**](#_Toc201666007)

[**Table 24 ObjectOriented (OO) Approach 54**](#_Toc201666008)

[**Table 25 Screen: Registration & Login Page 62**](#_Toc201666009)

[**Table 26 Screen: Donation Page 62**](#_Toc201666010)

[**Table 27 User Registration & Authentication 66**](#_Toc201666011)

[**Table 28 Organization Registration 66**](#_Toc201666012)

[**Table 29 Donation Management 67**](#_Toc201666013)

[**Table 30 Gamification Features 67**](#_Toc201666014)

[**Table 31 Location Based Orphanage Finder 68**](#_Toc201666015)

**TABLE OF FIGURES**

[Figure 1 Gantt Chart 22](#_Toc201665926)

[Figure 2 User Management Class Diagram 33](#_Toc201665927)

[Figure 3 Donation Management Class Diagram 34](#_Toc201665928)

[Figure 4 Notification System Class Diagram 35](#_Toc201665929)

[Figure 5 Orphanage & Request Management Class Diagram 36](#_Toc201665930)

[Figure 6 Payment & Transaction Processing Class Diagram 37](#_Toc201665931)

[Figure 7 Admin managing orphanages. 38](#_Toc201665932)

[Figure 8 Donor selecting an orphanage 39](#_Toc201665933)

[Figure 9 Steps for a new user (donor) signing up. 40](#_Toc201665934)

[Figure 10 Steps for orphanages registering and updating their profiles. 41](#_Toc201665935)

[Figure 11 User authentication 42](#_Toc201665936)

[Figure 12 Donation process 43](#_Toc201665937)

[Figure 13 Orphanage registration 44](#_Toc201665938)

[Figure 14 Donation center Inventory management 45](#_Toc201665939)

[Figure 15 Search and filter 46](#_Toc201665940)

[Figure 16 System Architecture Diagram 48](#_Toc201665941)

[Figure 17 ER Diagram 51](#_Toc201665942)

[Figure 18 Data Flow Diagram 51](#_Toc201665943)

[Figure 19 home page 56](#_Toc201665944)

[Figure 20 Profile Page: 57](#_Toc201665945)

[Figure 21 About Us 58](#_Toc201665946)

[Figure 22 Subscription Page 59](#_Toc201665947)

[Figure 23 FAQ Page 60](#_Toc201665948)

[Figure 24 Donation Page: 61](#_Toc201665949)

[Figure 25 Donor Use Cases 69](#_Toc201665950)

[Figure 26 Orphanage Admin Use Cases 70](#_Toc201665951)

[Figure 27 System Admin Use Cases 70](#_Toc201665952)

[Figure 28 Guest User Use Cases 71](#_Toc201665953)

A blue heart with a hand holding a heart

AI-generated content may be incorrect.

# CHAPTER 1

# INTRODUCTION

# Introduction

Safe Haven is an online platform that acts as a central hub and is created to bring orphanages and donation centers to people who wish to give a helping hand by donating. It serves as a sanctuary where orphanages can share their needs, stories and contact data, and donors can find out what the orphanages need.

Now, think for a moment that you have the means to help those in need, but you do not have a reliable way to do so. You want to donate to a local orphanage, but you are not sure if the orphanage near you is authentic or not, what to give to these organizations or if your donation will even reach the right hands. This is the reality for many potential donors in a city like Sialkot.

Many people want to contribute but face obstacles:

**Where to donate**: no unique and reliable source that lists all donation centers and orphanages in Sialkot. This forces donors to trust mouth-to-mouth or incomplete online information. **What specific needs exist**: donors often do not know what is required: money, clothing, school supplies or food. This lack of clarity discourages them from taking the next step. **Effective use of donations**: even after donating, people care that their contributions were used correctly or had a real impact.

Safe Haven was born with a clear mission: connecting the donors and orphanages with security, ease and impact. By providing a centralized platform, our goal is to simplify the donation process, create awareness of donation platforms, promote trust and create an effort promoted by the community to address social needs in Sialkot.

# Problem Statement

Potential donors often face significant challenges in the identification of qualified organizations, including their specific needs and guarantee that their contributions reach the planned beneficiaries. Without a simplified platform, donors can have difficulties:

* Access to updated and verified information about donation centers and orphanages.
* Know what articles or funds are needed more urgently.
* Manage and track your donations effectively.

On the other hand, donation centers and orphanages experience difficulties such as:

* Communicate your immediate and continuous needs to a broad audience.
* Maintain transparency and responsibility, which are critical to build donor confidence.
* Reach possible donors beyond their immediate geographical or social networks.

This lack of a transparent and optimised donation process results in:

* Wasting resources due to an excess of offers of less necessary items or contributions insufficient for high-priority needs.
* Reduction of trust among donors due to concerns about fraud or poor management.
* Lost opportunities to support vulnerable populations effectively.

# Problem Solution

Safe Haven proposes a centralized online platform to address these issues. It will offer detailed and verified profiles of orphanages, highlighting their specific requirements in real-time. A secure online donation system will allow users to contribute directly, with transparent tracking of funds and confirmation receipts. The platform will incorporate location-based search, helping users find nearby orphanages and donation centers easily. By simplifying the donation process and fostering awareness, Safe Haven intends to build trust and increase effective resource utilization. This comprehensive web-based solution aims to minimize wasted resources, combat distrust due to fraud concerns, and ensure vulnerable populations receive adequate support.

# Objectives

* **Donation and fundraising:**

Safe Haven has simplified donations and fundraising with options such as unique, recurring or specific contributions to specific programs such as education or medical care.

* **Facilitate the visibility and consciousness of the orphanage:**

The detailed orphanage profiles highlight the needs, stories and achievements, helping donors make informed decisions.

* **Creation of a scalable and sustainable platform:**

With sponsorship programs and organisational associations, Safe Haven guarantees long-term sustainability and space for growth.

* **Incorporation of localized solutions for greater accessibility:**

The platform offers local payment options, language support and culturally relevant characteristics to guarantee accessibility for various users.

* **Location-based orphanage search:**

A manual or enabled search for GPS helps users find orphanages nearby, improving accessibility and convenience.

* **Donation portal for financial support:**

A safe online donation system allows users to contribute directly to orphanages, with transparent follow-up of funds.

* **Orphanage reviews and grades**:

Users can leave reviews and grades, promote transparency and help others evaluate the quality of care provided by orphanages.

# Scope of study

The scope study for this project is to address the problem of locating orphanages, donation and well-being centers through an easy web platform. Development focuses on providing an easy experience to use for individuals, families, organizations and companies interested in donating.

The system will simplify the orphanage location process and make donations currently taken inefficiently for users. All orphanages that appear on the platform will be required to maintain their profiles with precise and updated information, ensuring that users have access to reliable details about each orphanage.

The project will focus on improving the usability of the web-based system in terms of user experience and navigation interface, which facilitates users to interact with the platform. The system will be developed using modern web technologies, including a responsive design to guarantee accessibility on several devices, and will present functionalities such as a safe donation portal.

# System Components

The Safe Haven online platform is designed to provide a secure, efficient, and user-friendly experience for connecting donors with orphanages and donation centers. Its integrated modules cater to the distinct needs of administrators, organizations, and donors.

**1. Core Management & Verification Module**

* **User and Organization Registration & Authentication:** Facilitates secure account creation and login for all users (administrators, organizations, donors) and manages their profiles.
* **Trust and Vetting Process:** Implements a robust process for authenticating and verifying new orphanages and donation centers joining the platform, building donor confidence. This also includes potential donor verification.
* **Content Moderation:** Oversees and approves all content shared by organizations, ensuring accuracy and appropriateness.
* **System Configuration & Maintenance:** Allows administrators to manage platform settings, categorize needs, and perform essential system upkeep.

**2. Needs & Donation Facilitation Module**

* **Organization Profile & Story Sharing:** Enables orphanages and donation centers to create comprehensive profiles, share their stories, and provide essential contact data.
* **Needs Posting & Management:** Allows organizations to clearly articulate and post their specific requirements (e.g., money, clothing, school supplies, food).
* **Donation Search & Discovery:** Provides donors with tools to search for and easily discover registered organizations and their specific needs.
* **Secure Donation Processing:** Offers streamlined and secure mechanisms for donors to initiate and track their contributions.

**3. Communication, Transparency & Analytics Module**

* **In-Platform Communication:** Facilitates secure messaging between donors and organizations, as well as announcements from administrators to all users.
* **Automated Notifications & Reminders:** Sends timely alerts to donors about new needs and updates on their donations, and to organizations regarding received contributions.
* **Donation & Impact Tracking:** Provides tools for both donors to view their past contributions and for organizations to track incoming donations and manage their utilization.
* **Reporting & Analytics:** Generates detailed reports for organizations on fulfilled needs and for administrators on platform usage, donation trends, and overall system performance. This includes an audit trail for accountability.
* **Public Transparency Features:** Potentially offers anonymized public dashboards showcasing overall donation impact to further build trust.

# Related System Analysis/Literature Review

Various digital platforms have been developed globally and in Pakistan to support orphanages and orphaned children. These systems offer services ranging from crowdfunding and sponsorship programs to complete shelter and education facilities.

While many have made significant contributions, most platforms lack localized integration, real-time donation tracking, orphanage geolocation, and direct communication tools between donors and institutions.

Our proposed project aims to fill these gaps by providing a centralized, interactive, and transparent web-based system for orphanage discovery and donation management.

Table 3 related work

| **Application Name** | **Weakness** | **Proposed Project Solution** |
| --- | --- | --- |
| Africa’s Orphans Support Platform | Focuses on crowdfunding but lacks a dedicated orphanage mapping system or region-specific support for countries like Pakistan. | Provides orphanage geolocation in Pakistan with user-friendly filters and search tools. |
| Miracle Foundation | Focuses on family-based care transition rather than platform-based orphanage support or donor interactions. | Designed specifically to support orphanage facilities and connect them directly with potential donors. |
| World Orphan Fund | Highly transparent but lacks a digital platform with interactive features like maps, donation dashboards, or mobile accessibility. | Offers an interactive web platform with maps, donor dashboards, and mobile-friendly interface. |
| Alkhidmat Foundation's Orphan Care Program | Offers sponsorship but no online platform for orphanage discovery or user interaction. | Integrates local orphanage listings and allows online donor selection and tracking. |
| Pakistan Sweet Home | Strong nationwide presence but lacks an open-access digital platform with location-based search and public donation system. | Adds public accessibility via a web platform that allows users to find and support orphanages based on location. |
| AlMustafa Welfare Society | Provides emotional and financial support but no user platform for direct public engagement or donation process. | Builds a donor-facing interface for transparent donation and communication with orphanage representatives. |
| Edhi Foundation | Offers broad social services but does not have a focused online solution for orphanage discovery and individual support tracking. | Focuses solely on orphanages and enables individualized support via a centralized and easy-to-use online portal. |

# System Limitations and Constraints

**Limitations (External Factors Beyond Control):**

* **Limited International Donations**: The project's reliance on local payment gateways, such as Easypaisa and JazzCash, restricts the ability of international donors to contribute unless additional global payment options are integrated in the future.
* **API Rate Limits & Service Downtime**: The system is dependent on third-party APIs like Google Maps and Firebase, which may impose limits on the number of requests or experience occasional downtime, potentially affecting the platform's performance and functionality.
* **Performance on Low-end Devices**: Although designed to be mobile-friendly, some features of "Safe Haven" may not operate optimally on very old or low-performance devices.
* **Regulatory Compliance Challenges**: Adhering to both local and international financial regulations for charitable contributions requires continuous updates and legal validation, which is an ongoing external challenge.
* **Internet Connectivity Issues**: Users, including donors and orphanages, in remote areas or those with poor internet connectivity may face challenges accessing and utilizing the platform effectively.

**Constraints (Self-Imposed Controllable Factors):**

* **Security & Privacy**: All user data must be encrypted using SSL/TLS to ensure secure communication. The platform must comply with GDPR and data privacy laws to protect donor and orphanage information.
* **Scalability**: The platform is constrained to be scalable, meaning it must accommodate an increasing number of users and orphanages without experiencing performance degradation.
* **Compatibility**: The system must be responsive and compatible with all modern web browsers (Chrome, Firefox, Edge, Safari) and function across desktop, tablet, and mobile devices without loss of functionality.
* **User Digital Literacy**: The assumption is that users will possess basic digital literacy skills to navigate the web platform, register, and perform actions like donations and searching for orphanages.
* **Genuine Orphanage Registration**: A self-imposed constraint is that all orphanages registering on the platform are expected to provide valid and truthful information regarding their organization, needs, and services.
* **Secure Transactions**: All financial transactions within the system will be processed through secure and trusted payment gateways to ensure donor safety and prevent fraud.
* **Orphanage Participation**: Orphanages are constrained to actively maintain their profiles, update donation needs, and respond to donors in a timely manner.

# Tools, Libraries and Technologies with Reasoning

Table 4 Tools, Technologies, and Libraries

|  |  |  |
| --- | --- | --- |
| **Tools** | **Version** | **Rationale** |
| Visual Studio Code | Latest | Lightweight and feature-rich code editor |
| Git & GitHub | Latest | Version control and collaboration among team members. |
| Figma | Latest | Prototype and design |
| **Technology** | **Version** | **Rationale** |
| Web Development (HTML, CSS, JavaScript) | Latest | Core technologies for building the web application. |
| MERN Stack (MongoDB, React, Node) | Latest | Ensures scalability, flexibility, and speed. |
| RESTful APIs | Latest | Facilitates communication between the front end and back end. |

# Project Planning



Figure 1 Gantt Chart

# Project Deliverables

* **Project Proposal Document**: Initial concept, objectives, scope, and feasibility analysis.
* **Software Requirements Specification (SRS)**: Detailed functional and non-functional requirements, use cases, and user stories.
* **System Design Documentation:** UML diagrams (class, use case, sequence, state), database ERD, architecture diagrams.
* **Frontend Development**: Responsive web interface using HTML, CSS, JavaScript.
* **Backend Development**: Server-side logic using Node.js with API endpoints.
* **Database Schema and Integration**: MongoDB-based database design with data models and secure CRUD operations.
* **Authentication & Authorization Module**: JWT-based login system for donors, orphanages, and admin users.
* **Donation Management System**: Secure and trackable donation workflows for monetary and in-kind donations.
* **Location-Based Orphanage Finder**:Google Maps API integration for geolocation and search functionality.
* **Notification & Subscription System**: Real-time alerts and email updates for users based on activity and preferences.
* **Review & Rating System**: Feedback mechanism for donors to rate and review orphanages.
* **Admin Dashboard**: Admin panel to manage users, verify orphanages, monitor activity, and generate reports.
* **Documentation & Reports**: Final project report, user manuals, test reports, and deployment guide.
* **Prototype / Wireframes**: Visual representation of core features created using Figma.

# Summary

The first chapter provides a comprehensive foundation for understanding the Safe Haven project by outlining its purpose, objectives, and key considerations. It begins with an introduction to the platform, explaining its significance in connecting donors and orphanages.

The background research highlights the existing challenges in the orphanage support system, emphasising the need for a centralized and transparent platform. The problem statement clearly defines the gaps that Safe Haven aims to address, while the objectives outline the specific goals of the project.

Lastly, the work division details how responsibilities are distributed among team members to facilitate smooth development. This chapter serves as the foundation for the rest of the document, guiding the project toward successful implementation.

**A blue heart with a hand holding a heart

AI-generated content may be incorrect.**

# CHAPTER 2

# DESIGN AND ARCHITECTURE

# Requirements Analysis

The Requirements Analysis of Safe Haven involves identifying key functional and nonfunctional needs to ensure an efficient and secure donation platform. Functional requirements include user authentication, orphanage verification, donation tracking, and a needs listing system to facilitate targeted contributions.

The platform must support real-time updates, secure payment processing, and communication features for donors and orphanages. Nonfunctional requirements focus on security, scalability, usability, and performance, ensuring a trustworthy, user-friendly, and efficient system. The goal is to create a seamless, transparent, and impactful platform that simplifies the donation process while fostering community engagement.

# User classes and characteristics

Table 5 User Classes and Their Characteristics

|  |  |
| --- | --- |
| **User Class** | **User Characteristics** |
| **Donor** | Individuals or organizations willing to donate money, food, clothes, or other essentials. They require a transparent and secure system to ensure their contributions reach the right orphanage. |
| **Orphanage Admin** | Representatives managing orphanages who list needs, receive donations and verify transactions. They require a user-friendly interface for updating orphanage profiles and responding to donors. |
| **General User** | Individuals exploring the platform for awareness or future contributions. They may not register immediately but require access to orphanage details and donation impact reports. |
| **Admin** | Platform administrators are responsible for user verification, orphanage approvals, monitoring donations, and ensuring system security and smooth operation. |

# Requirement Identifying Techniques

1. **Stakeholder Interviews**

* Conduct one-on-one or group interviews with donors, orphanage representatives, and administrators to understand their needs and expectations.
* Example: Asking orphanages about the challenges they face in receiving donations.

1. **Surveys & Questionnaires**

* Collect data from potential users through online surveys to understand donation habits, security concerns, and preferred features.
* Example: Asking donors if they prefer monetary donations or physical goods.

1. **Document Analysis**

* Review existing orphanage records, donation receipts, and financial reports to understand donation trends and regulatory requirements.
* Example: Analyzing reports from NGOs on donation fraud prevention.

# Functional Requirements

**Functional Requirements Specification for User Registration & Authentication**

Table 6 Functional Requirements Specification for User Registration

|  |  |
| --- | --- |
| **Identifier** | **FR1** |
| **Title** | User Registration & Authentication |
| **Requirement** | The system shall allow users (donors, orphanages, and admins) to register and log in securely using email and password authentication. Users shall also be able to recover passwords if forgotten. |
| **Source** | Project Stakeholder |
| **Rationale** | Ensures secure user access and prevents unauthorized use of the platform. |
| **Business Rule (if required)** | Users must verify their email before accessing full platform functionalities. |
| **Dependencies** | FR2 (User Profiles), FR8 (Admin Dashboard) |
| **Priority** | High |

**Functional Requirements Specification for User Profile**

Table 7 Functional Requirements Specification for User Profile

|  |  |
| --- | --- |
| **Identifier** | **FR2** |
| **Title** | User Profiles |
| **Requirement** | The system shall allow users to create, update, and manage their profiles, including personal information and donation history. |
| **Source** | User Needs Analysis |
| **Rationale** | Enables personalization and helps track user activities on the platform. |
| **Business Rule (if required)** | Users must provide valid information; admins can suspend incomplete or fraudulent profiles. |
| **Dependencies** | FR1 (User Registration), FR5 (Donation Management) |
| **Priority** | High |

**Functional Requirements Specification for Orphanage Registration**

Table 8 Functional Requirements Specification for Orphanage Registration

|  |  |
| --- | --- |
| **Identifier** | **FR3** |
| **Title** | Orphanage Registration |
| **Requirement** | The system shall allow orphanages to register and create profiles, listing their needs, location, and contact information. |
| **Source** | Orphanage Administrators, Platform Guidelines |
| **Rationale** | Enables donors to find and support orphanages efficiently. |
| **Business Rule (if required)** | Orphanages must submit verification documents before approval. |
| **Dependencies** | FR8 (Admin Dashboard), FR4 (Orphanage Listing) |
| **Priority** | High |

**Functional Requirements Specification for User Orphanage Listing & Search**

Table 9 Functional Requirements Specification for User Orphanage Listing

|  |  |
| --- | --- |
| **Identifier** | **FR4** |
| **Title** | Orphanage Listing & Search |
| **Requirement** | The system shall display a list of verified orphanages and allow users to search and filter by location, needs, and ratings. |
| **Source** | User Experience Research |
| **Rationale** | Simplifies the process of finding and supporting orphanages. |
| **Business Rule (if required)** | Only verified orphanages will appear in search results. |
| **Dependencies** | FR3 (Orphanage Registration) |
| **Priority** | High |

**Functional Requirements Specification for Donation Management**

Table 10 Functional Requirements Specification for Donation Management

|  |  |
| --- | --- |
| **Identifier** | **FR5** |
| **Title** | Donation Management |
| **Requirement** | The system shall allow users to donate money or in-kind items to orphanages and receive confirmation receipts. |
| **Source** | Donor Feedback, Legal Compliance |
| **Rationale** | Ensures a structured donation process with transparency. |
| **Business Rule (if required)** | Donors must receive confirmation from the orphanage after a successful donation. |
| **Dependencies** | FR3 (Orphanage Registration), FR4 (Orphanage Listing) |
| **Priority** | High |

# Non-Functional Requirements

1. **Reliability** 
   * The Mean Time Between Failures (MTBF) should be at least 500 hours to minimise unexpected downtime.
   * In case of failure, the system shall ensure a Mean Time to Recover (MTTR) of fewer than 30 minutes for critical services.
2. **Performance Efficiency**

* The system shall handle at least 500 concurrent users without performance degradation.
* Page loading time should not exceed 3 seconds under normal traffic conditions.
* Donation transactions shall be processed in less than 5 seconds.

1. **Security**

* The system shall use JWT-based authentication to secure user sessions.
* User passwords shall be stored hashed to prevent unauthorized access.
* Access to sensitive data shall be restricted based on role-based access control (RBAC).

1. **Usability**

* The platform shall maintain a user-friendly interface, ensuring that at least 85% of first-time users can navigate without guidance.
* A search and filter feature shall allow users to find orphanages and donation opportunities efficiently.

1. **Maintainability & Scalability**

* The platform should be modular, allowing easy updates and feature additions without impacting existing functionality.
* The database shall support efficient indexing and query optimization to handle growing data.

# External Interface Requirements

## User Interface Requirements

The Safe Haven platform is designed to provide an intuitive, accessible, and responsive user experience for donors, orphanages, and administrators. It follows industry best practices to ensure usability, accessibility, and consistency across different devices.

1. **Color Scheme:** Calming **blue & green** shades
2. **Layout & Navigation**

* **Responsive Design:** Supports screens from **360px (mobile) to 1920px (desktop)**.
* **Navigation:** Fixed **top navigation bar** with a **hamburger menu**.
* **UI Elements:** **Header (logo, navigation, hamburger )** & **Footer (contact info, social media links)** on every page.

1. **Standard Buttons & Functions**

* **Primary Actions:** Donate Now, Subscribe for Updates.
* **Secondary Actions:** Learn More, Contact Us, View Details.
* **Keyboard Shortcuts:** Tab (move fields), Enter (submit), Esc (close modals).

1. **Message Display & Feedback**

* **Success (Green):** Donation successfully processed!
* **Error (Red):** Invalid email address. Please try again.

## Software Requirements

1. **Frontend Requirements**
   * **Programming Languages:** JavaScript
   * **Frameworks & Libraries:** React.js (or Next.js}
   * **Styling & UI Components:** CSS
   * **Responsive Design:** Flexbox, Grid
2. **Backend Requirements**
   * **Programming Language:** Node.js
3. **Authentication & Authorization:**
   * JWT (JSON Web Tokens) for secure user authentication
4. **Database & Storage Requirements**
   * **Database:** MongoDB
5. **External APIs & Services**
   * **Payment Processing:** local payment gateway
   * **Mapping & Location Services:** Google Maps API or OpenStreetMap for orphanage locations
   * **Email & Notifications:** Nodemailer (for sending emails)
   * **SMS Services:** a local SMS provider (for alerts and confirmations)
6. **Security & Data Protection**
   * **Encryption:** SSL/TLS for secure communication
   * **Backup & Recovery:** Automatic database backups using AWS RDS or Firebase Firestore Backup
7. **Development & Deployment Tools**
   * **Version Control:** Git & GitHub

## Hardware Interfaces

**Description:** Defines the interaction between Safe Haven and external hardware components.

* **Server Requirements:**
  + The backend shall run on Node.js with MongoDB for data storage.
* **Security Devices:**
  + Supports integration with biometric authentication devices if required for orphanage verification in the future.

## Communication Interfaces

**Description:** Defines how the system interacts with users through messaging and alerts.

* **RealTime Chat:**
  + The system shall provide a chat feature for communication between donors and orphanages.
* **Email & SMS Notifications:**
  + Users shall receive notifications for donation requests and transaction confirmations.
* **Community Forum:**
  + A discussion board for donors and orphanages to share experiences

# SUMMARY

The requirements identification and use case analysis for Safe Haven were conducted to ensure a well-structured and efficient system for connecting donors and orphanages. The process began with defining functional requirements, such as orphanage registration, donation tracking, and communication features.

Nonfunctional requirements focused on security, reliability, system performance, and accessibility to provide a seamless user experience. Each requirement was analysed to align with project objectives, ensuring that the platform remains transparent, trustworthy, and easy to navigate.

Through use case analysis, user interactions were mapped, detailing system responses, preconditions, postconditions, and alternate flows. Key features like subscription management, donation processing, and communication systems were documented for clarity.

Additionally, storyboards were created to visualize user journeys, refining the design and usability. This structured approach ensures that Safe Haven effectively meets its mission of simplifying the donation process, increasing engagement, and fostering a sense of community-driven support for orphanages.

**A blue heart with a hand holding a heart

AI-generated content may be incorrect.**

# CHAPTER 3

# Design and Architecture

# System Design

The Safe Haven platform is a web-based application that facilitates secure donations and orphanage support through a centralized system. It operates as an interactive online platform, providing a structured way for users (donors and orphanages) to connect, manage donations, and track contributions effectively. The system functions as an intermediary between donors and orphanages, ensuring transparency and security in transactions.

## Dependencies and Interactions with Other Systems

* **Authentication Services**: Integration with JWT authentication ensures secure user login and account management.
* **Payment Gateways**: The platform supports Easypaisa, JazzCash and local payment providers to facilitate monetary donations securely.
* **Mapping & Location Services**: Uses Google Maps API to help users locate orphanages.
* **Email & Notification System**: Implements Nodemailer for email notifications and On**e** Signal for real-time alerts.
* **SMS Integration**: Uses a local SMS provider to send donation confirmations and urgent alerts.
* **Database Management**: Uses MongoDB (NoSQ**L)**  for efficient storage and retrieval of user data, donation history, and orphanage details.

## Design Constraints

* + - **Performance Constraints**
  + The system must support high concurrency, handling multiple users accessing and performing transactions simultaneously.
  + Page load times should be under 3 seconds for optimal user experience.
  + Efficient database indexing and caching mechanisms must be implemented to enhance performance.
    - **Security & Privacy Constraints**
  + All user data must be encrypted using SSL/TLS to ensure secure communication.
  + Compliance with GDPR and data privacy laws to protect donor and orphanage information.
  + Secure role-based access control (RBAC) must be implemented to restrict user permissions.
    - **Scalability Constraints**
  + The platform should be scalable to accommodate an increasing number of users and orphanages without performance degradation.
    - **Compatibility Constraints**
  + The system must be responsive and compatible with all modern browsers (Chrome, Firefox, Edge, Safari).
  + It should work across desktop, tablet, and mobile devices without functionality loss.

# Design Considerations

## Assumptions

* Users have stable internet access and use modern web browsers (Chrome).
* Local payment gateways (Easypaisa, JazzCash) will be available for processing transactions.
* External services like Google Maps API, Nodemailer (for emails), and Firebase(for notifications) will function reliably.
* The platform will be mobile-friendly, supporting various screen sizes and devices.
* Users will adhere to secure authentication protocols using JWT tokens.

## Limitations

* **Limited International Donations:** The reliance on local payment gateways restricts international donors unless additional global payment options are integrated.
* **API Rate Limits & Service Downtime:** The system depends on third-party APIs (Google Maps, Firebase, etc.), which may limit the number of requests or experience downtime.
* **Data Privacy & Customization:** Initial versions of the system may not support advanced data privacy features or extensive customization.
* **Performance on Low-end Devices:** While mobile friendly, some features may notwork optimally on very old or low performance devices.
* **Regulatory Compliance Challenges:** Adhering to local and international financial regulations requires continuous updates and legal validation.

# Requirement Traceability Matrix

Table 11 Requirement Traceability Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Scope** | **Requirement Description** | **Design Specification** |
| **FR1** | User Registration & Authentication | The system shall allow users to register and log in securely using email and password authentication. | User Authentication Module |
| **FR2** | User Profiles | The system shall allow users to create, update, and manage their profiles. | User Profile Management |
| **FR3** | Orphanage Registration | The system shall allow orphanages to register and create profiles. | Orphanage Registration Module |
| **FR4** | Orphanage Listing & Search | The system shall display a list of verified orphanages and allow users to search and filter. | Orphanage Listing & Search Engine |
| **FR5** | Donation Management | The system shall allow users to donate money or in-kind items to orphanages. | Donation Processing System |

# **Design Models**

## Design Class Diagram (DCD)

### ****User Management Class Diagram****

**Description:**

This class diagram represents the user management system, where users (donors, orphanages, and admins) can register, log in, and manage their profiles. The system enforces authentication and role-based access control.

**Table Representation:**

Table 12 User Management Class Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attributes** | **Methods** | **Relationships** |
| **User** | id:String  name:String  email:String  password: String  role: String | register():void  login():void  logout(): void | Generalization (Donor, Orphanage, Admin) |
| **Donor** | donation History: List Donation | donate(): void | Inherits User |
| **Orphanage** | location: String  needs: List String | update Needs(): void  receiveDonation(): void | Inherits User |
| **Admin** | privileges: ListString | manageUsers(): void  approveOrphanage(): void | Inherits User |

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2 User Management Class Diagram

### Donation Management Class Diagram

**Description:**

This class diagram models the donation process, where donors can contribute money or goods to orphanages. The system ensures secure transactions and donation tracking.

**Table Representation:**

Table 13 Donation Management Class Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attributes** | **Methods** | **Relationships** |
| **Donation** | id: String  amount: Double  type: String  status: String | confirmDonation(): void  trackDonation(): void | Association with Donor and Orphanage |
| **Donor** | donationHistory: ListDonation | donate(): void | Association with Donation |
| **Orphanage** | needs: ListString | receiveDonation(): void | Association with Donation |
| **PaymentGateway** | provider: String | processPayment(): void | Aggregation with Donation |

A close-up of a white background

AI-generated content may be incorrect.

Figure 3 Donation Management Class Diagram

### Notification System Class Diagram

**Description:**

This diagram represents the real-time notification system, ensuring that users receive updates about donations, new orphanages, and urgent needs.

**Table Representation:**

Table 14 Notification System Class Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attributes** | **Methods** | **Relationships** |
| **Notification** | id: String  message: String  type: String  timestamp: DateTime | sendNotification(): void | Association with User |
| **User** | name: String  email: String | receiveNotification(): void | Association with Notification |
| **Notifier** | notificationList: ListNotification | triggerNotification(): void | Aggregation with Notification |

A white rectangular object with black text

AI-generated content may be incorrect.

Figure 4 Notification System Class Diagram

### Orphanage & Request Management Class Diagram

**Description:**

This class diagram represents the orphanage management and request system, where orphanages can register, update their details, and request specific donations. Donors can view and fulfill these requests.

**Table Representation:**

Table 15 Orphanage & Request Management Class Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attributes** | **Methods** | **Relationships** |
| **Orphanage** | id: String  name: String  location: String  needs: ListString | updateProfile(): void  listNeeds(): void | Aggregation with Request |
| **Request** | id: String  description: String  status: String  requestDate: DateTime | approveRequest(): void  fulfillRequest(): void | Association with Donor and Orphanage |
| **Donor** | id: String  name: String  donationHistory: ListRequest | viewRequests(): void  fulfillRequest(): void | Association with Request |
| **Admin** | id: String  privileges: ListString | approveOrphanage(): void  manageRequests(): void | Association with Orphanage and Request |

A screenshot of a computer

AI-generated content may be incorrect.

Figure 5 Orphanage & Request Management Class Diagram

### Payment & Transaction Processing Class Diagram

**Description:**

This diagram represents the payment and transaction processing system, ensuring secure monetary donations via a payment gateway.

**Table Representation:**

Table 16 Payment & Transaction Processing Class Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attributes** | **Methods** | **Relationships** |
| **Transaction** | id: String  amount: Double  status: String  timestamp: DateTime | processTransaction(): void  verifyTransaction(): void | Association with Donor and PaymentGateway |
| **Donor** | id: String  name: String | donate(): void | Association with Transaction |
| **Orphanage** | id: String  name: String | receiveFunds(): void | Association with Transaction |
| **PaymentGateway** | provider: String | processPayment(): void | Aggregation with Transaction |

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6 Payment & Transaction Processing Class Diagram

## **Interaction Diagram (Sequence** diagram)

The UML Sequence Diagrams for Safe Haven illustrate the flow of interactions between users (donors, orphanages, es, and admins) and the system components. These diagrams map out how data moves within the system, ensuring smooth and efficient operations.

1. **User Registration & Login:** Shows how users register and log in, with backend validation and secure authentication.
2. **Orphanage Registration & Profile Update:** Details how orphanages sign up, create profiles, and update their needs.
3. **Donor Browsing & Donation:** Demonstrates how donors search for orphanages, make donations, and receive confirmations.
4. **Orphanage Needs Posting:** Describes how orphanages post their current needs, which donors can view and address.
5. **Admin Management:** Illustrates how admins manage users, approve profiles, and handle flagged content.
6. **Donation Tracking & Reporting:** Displays how donations are tracked, with reports available to donors and orphanages.
7. **Notification & Email Alerts:** Highlights how users receive notifications and emails about key activities and updates.
8. **Search & Filter Orphanages:** Shows the search process, allowing users to filter orphanages based on various criteria.
9. **Feedback & Reviews:** Depicts how users leave reviews for orphanages, enhancing transparency and trust.

## Sequence Diagram for Each Actor

### Admin Dashboard Management:

A diagram with text on it

AI-generated content may be incorrect.

Figure 7 Admin managing orphanages.

**Purpose:** To illustrate how admins manage users and content.  
**Flow:**

* + Admin logs into the system.
  + Views pending orphanage profiles or flagged content.
  + Approves, edits, or deletes entries.
  + Database is updated accordingly, and notifications are sent to users.

### Donation Management:

**Make a Donation (Goods/Funds)**

**A diagram of a donation

AI-generated content may be incorrect.**

Figure 8 Donor selecting an orphanage

**Purpose:** To depict how a donor browses orphanages and donates.

**Flow:**

* + Donor searches for orphanages.
  + System filters and displays matching results.
  + Donor selects an orphanage and initiates a donation.
  + Payment is processed via the payment gateway.
  + Donation is recorded, and confirmation is sent to the donor.

### User Management:

**User Registration**

A diagram of a server

AI-generated content may be incorrect.

Figure 9 Steps for a new user (donor) signing up.

**Purpose:** To show the process of a user registering or logging into the platform.  
**Flow:**

* + User submits registration/login details.
  + Frontend sends data to Backend.
  + Backend validates and interacts with the Database.
  + User receives confirmation or error

### Orphanage Management:

**Orphanage Registration & Profile Update**

**A diagram of a software application

AI-generated content may be incorrect.**

Figure 10 Steps for orphanages registering and updating their profiles.

**Purpose:** To illustrate how an orphanage signs up and manages its profile.  
**Flow:**

* Orphanage registers and submits details.
* Backend validates and stores data.
* Orphanage can later update profile info and needs.
* Database reflects updated details.

## State Transition Diagram

### User authentication

**Description:**

The user authentication state diagram represents the process of logging into the system. It includes states such as entering credentials, verification, success, or failure due to incorrect input.

**State Table:**

Table 17 User authentication

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Description** | **Transition Event** | **Next State** |
| Idle | System is waiting for user input | User enters credentials | Processing Login |
| Processing Login | Credentials are validated | Credentials are correct | Login Successful |
| Processing Login | Credentials are invalid | Credentials are incorrect | Login Failed |
| Login Successful | User is authenticated | ------- | User Dashboard |
| Login Failed | User is redirected to retry login | User retries login | Processing Login |

A diagram of a login

AI-generated content may be incorrect.

Figure 11 User authentication

### **Donation process**

**Description:**

This diagram shows the different states of a donation, from initiation to completion, including payment processing and confirmation.

**State Table:**

Table 18 Donation process

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Description** | **Transition Event** | **Next State** |
| Idle | User decides to donate | User selects donation type | Selecting Donation Type |
| Selecting Donation Type | User chooses between monetary or item donation | User confirms selection | Filling Donation Details |
| Filling Donation Details | User enters details of the donation | User submits donation | Processing Donation |
| Processing Donation | System verifies details and payment | Payment successful | Donation Confirmed |
| Processing Donation | System verifies details and payment | Payment failed | Donation Failed |
| Donation Confirmed | Donation is completed and acknowledged |  | Idle |

**A diagram of a payment process

AI-generated content may be incorrect.**

Figure 12 Donation process

### Orphanage registration

**Description:**

This diagram represents the steps an orphanage follows to register on the platform.

**State Table:**

Table 19 Orphanage registration

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Description** | **Transition Event** | **Next State** |
| Idle | Orphanage decides to register | Orphanage clicks Register | Filling Registration Form |
| Filling Registration Form | Orphanage provides details | Orphanage submits form | Verification Pending |
| Verification Pending | Verification Pending | Approved by admin | Registration Successful |
| Verification Pending | Admin verifies details | Rejected by admin | Registration Failed |
| Registration Successful | Orphanage can now update its profile and needs |  | Idle |

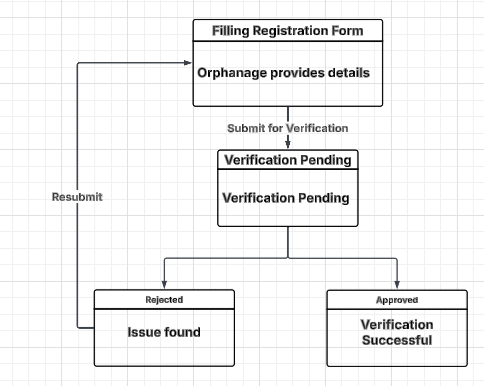
****

Figure 13 Orphanage registration

### **Donation center Inventory management**

**Description:**

This diagram represents how a donation center manages its inventory, from item addition to updating stock levels.

**State Table:**

Table 20 Donation center Inventory management

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Description** | **Transition Event** | **Next State** |
| Idle | System waits for inventory update | Donation center adds items | Adding Items |
| Adding Items | System processes new inventory items | Items successfully added | Inventory Updated |
| Adding Items | System processes new inventory items | Item entry failed | Error |
| Inventory Updated | Updated inventory is available |  | Idle |

**A diagram of a process

AI-generated content may be incorrect.**

Figure 14 Donation center Inventory management

### Search and filter

**Description:**

This diagram represents the different states involved when a user searches and filters orphanages or donation centers.

**State Table:**

Table 21 Search and filter

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Description** | **Transition Event** | **Next State** |
| Idle | System is waiting for search input | User enters query | Processing Search |
| Processing Search | System filters and sorts results | Results found | Displaying Results |
| Processing Search | System filters and sorts results | No results found | No Results Found |
| Displaying Results | Search results are displayed | User selects an option | Viewing Details |
| No Results Found | No matching data is found | User modifies search | Processing Search |

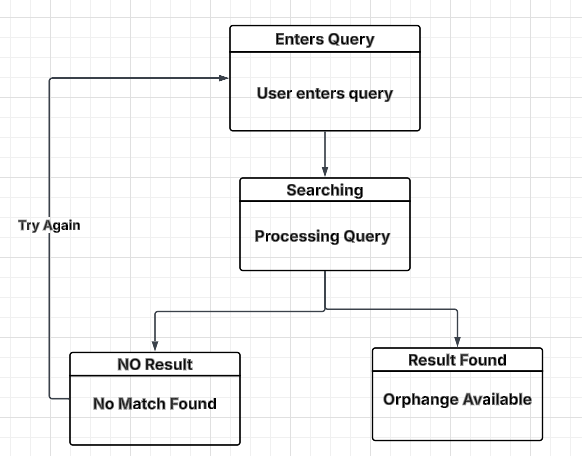
****

Figure 15 Search and filter

# System Architecture

The system architecture for **Safe Haven** is designed to efficiently connect donors, orphanages, and administrators through a robust, secure, and scalable web platform. The architecture follows a **three-tier model** consisting of the **client (Frontend), Application Server (Backend),** and **Database,** along with external service integrations like payment gateways and mapping APIs. This setup ensures smooth interactions, data flow, and scalability.

## ****Components:****

1. **Client Side (Frontend):**
   * **Technology:** React.js, HTML, CSS, JavaScript
   * **Responsibilities:**
     + Provides an interactive and user-friendly interface.
     + Allows users to register, browse orphanages, make donations, and track activities.
     + Communicates with the backend via RESTful APIs.
2. **Application Server (Backend):**
   * **Technology:** Node.js
   * **Responsibilities:**
     + Handles business logic and processes user requests.
     + Manages authentication using JWT for secure login.
     + Facilitates donation processes, orphanage management, and user interactions.
     + Integrates with payment gateways (**Easypaisa** and **JazzCash**) and **Google Maps API.**
3. **Database:**
   * **Technology:** MongoDB (NoSQL)
   * **Responsibilities:**
     + Stores user profiles, orphanage details, donation records, and transaction logs.
     + Manages relationships between users, donations, and orphanages.
     + Ensures data consistency and quick retrieval.
4. **External Services:**
   * **Payment Gateways:** Easypaisa and JazzCash for secure and local online transactions.
   * **Email Notifications:** Nodemailer for sending donation confirmations and updates.
   * **Mapping Service:** Google Maps API to display orphanage locations and directions.

## ****Interactions:****

1. **Frontend to Backend:**
   * Communication occurs over **secure HTTPS** using **RESTful APIs.**
   * Example: When a user donates, the frontend sends a donation request to the backend API.
2. **Backend to Database:**
   * The backend uses **Mongoose** (MongoDB ORM) to perform CRUD operations.
   * Example: Storing new user registration details or donation records.
3. **Backend to External Services:**
   * **Payment Integration:** The backend communicates with **Easypaisa** and **JazzCash** APIs to process payments and verify transactions.
   * **Email Notifications:** Nodemailer sends confirmation emails post donation.
   * **Mapping:** The backend fetches location data from **Google Maps API** to render orphanage locations on the frontend.

## System Architecture Diagram

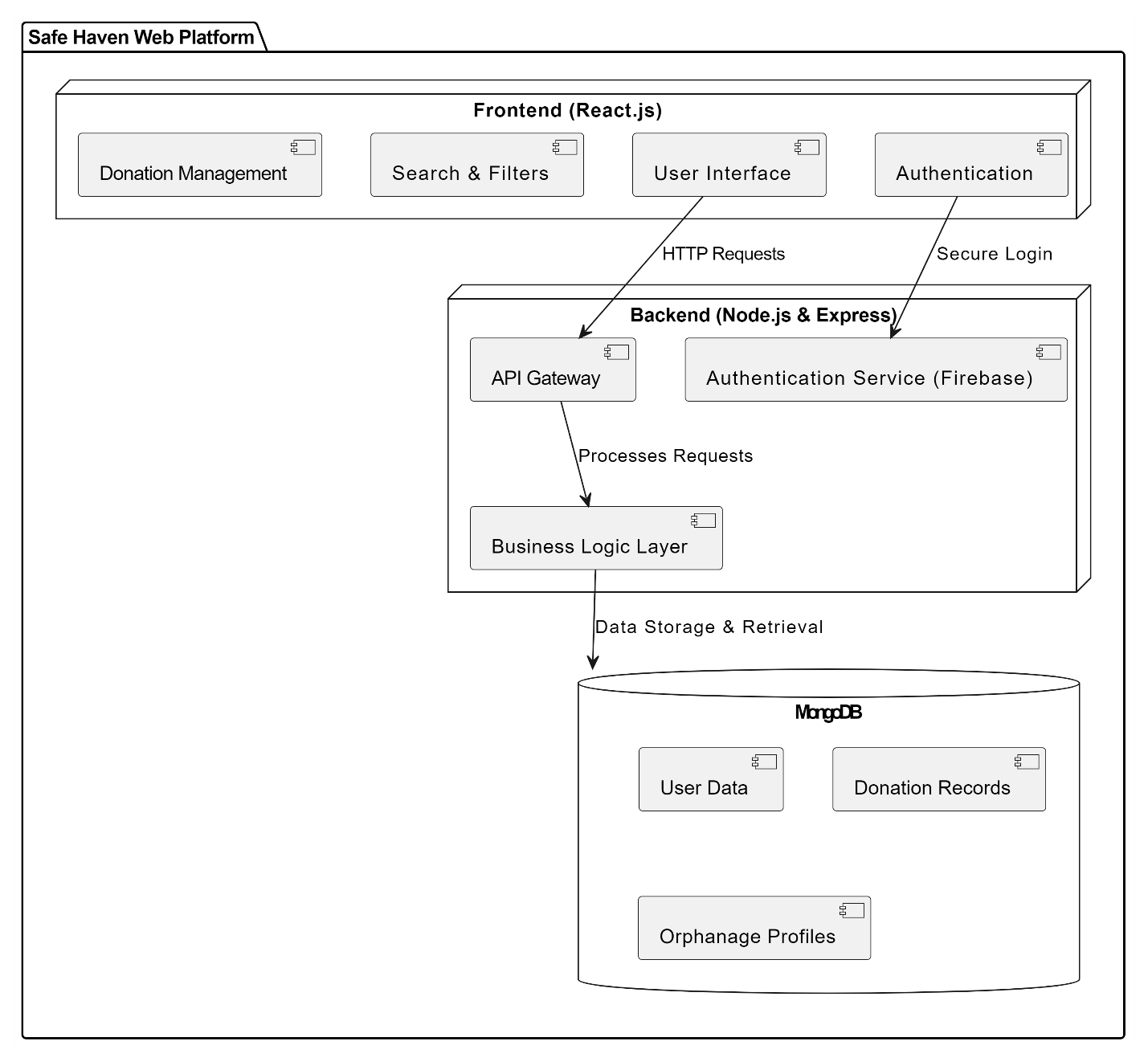


Figure 16 System Architecture Diagram

Here’s how it works:

**1. Frontend (React.js)**

* Users interact with the User Interface (UI) to perform actions.
* Key functionalities include:
  + **Donation Management** – Allows users to make and track donations.
  + **Search & Filters** – Users can search for orphanages and filter results.
  + **User Interface** – Handles the visual presentation and user experience.
  + **Authentication** – Manages user login and registration.

**2. Backend (Node.js & )**

* The Frontend sends HTTP Requests to the backend for processing.
* **API Gateway** acts as the entry point, routing requests to appropriate services.
* **Authentication Service (Firebase)** handles user authentication and secure login.
* **Business Logic Layer** processes user requests and manages the application's logic.
* Backend interacts with the MongoDB Database to fetch and store data.

**3. Database (MongoDB)**

* Stores and manages application data:
  + **User Data** – Stores registered users and their details.
  + **Donation Records** – Keeps track of all donations.
  + **Orphanage Profiles** – Stores information about orphanages.

**4. Data Flow**

* User initiates an action in the Frontend UI.
* HTTP Request is sent to the API Gateway in the Backend.
* If authentication is needed, the request goes through Firebase Authentication.
* The Business Logic Layer processes the request.
* The backend interacts with MongoDB for data retrieval/storage.
* The processed data is sent back to the Frontend UI for display.

# Data Design

## Database Design

### Data Requirements

The Safe Haven system comprises several key entities designed to facilitate seamless interactions between donors, orphanages, and administrators. Core entities include Users, who can register as donors, , or admins, and Orphanages, which provide information about their needs and location.

* The Donation entity tracks contributions
* Feedback allows users to review and rate orphanages, promoting transparency.
* Messages and Notifications support communication between users.
* The Location entity aids in mapping orphanages, ensuring users can find nearby centers easily.

Together, these entities create a robust system that simplifies donations, enhances user engagement, and promotes transparency.

### Entity and their Attributes

* 1. **User** (userID, name, email, password, phone, address, role (Dono/Admin), profilePicture, createdAt)
  2. **Orphanage** (orphanageID, name, location, contactPerson, phone, email, description, needs, registrationDate, image)
  3. **Donation** (donationID, userID, orphanageID, donationType (Money/Items), amountOrItem, donationDate, status)
  4. **Feedback/Review** (feedbackID, userID, orphanageID, rating, comment, date)
  5. **Message/Notification** (messageID, senderID, receiverID, content, timestamp, status)
  6. **Location** (locationID, orphanage, latitude, longitude, address, city, state, country)

### Primary Keys (PK) and Foreign Keys (FK)

|  |  |
| --- | --- |
| 1. **User** | 1. Primary Key: user\_id |
| 1. Foreign Keys: None |
| 1. **Orphanage** | 1. Primary Key: orphanage\_id |
| 1. Foreign Keys: None |
| 1. **Donation** | 1. Primary Key: donation\_id |
| 1. Foreign Keys: |
| * + 1. user\_id → References User(user\_id) |
| * + 1. orphanage\_id → References Orphanage(orphanage\_id) |
| 1. **Feedback/Review** | 1. Primary Key: feedback\_id |
| 1. Foreign Keys: |
| * + 1. user\_id → References User(user\_id) |
| * + 1. orphanage\_id → References Orphanage(orphanage\_id) |
| 1. **Admin** | 1. Primary Key: admin\_id |
| 1. Foreign Keys: None |
| 1. **Notifications** | 1. Primary Key: notification\_id |
| 1. Foreign Keys: |
| * + 1. user\_id → References User(user\_id) |

### Relationships

#### ****One-to-Many Relationships:****

* **User → Donations:**
  + One user (donor) can make multiple donations.
* **Orphanage → Donations:**
  + One orphanage can receive multiple donations.

#### ****One-to-One Relationships:****

**User ↔ Profile:**

Each user may have an extended profile (with details like profile picture, bio, etc.).

**Orphanage ↔ Location Details:**

Each orphanage may have a unique location entry with detailed address info (if handled separately).

### ER Diagram

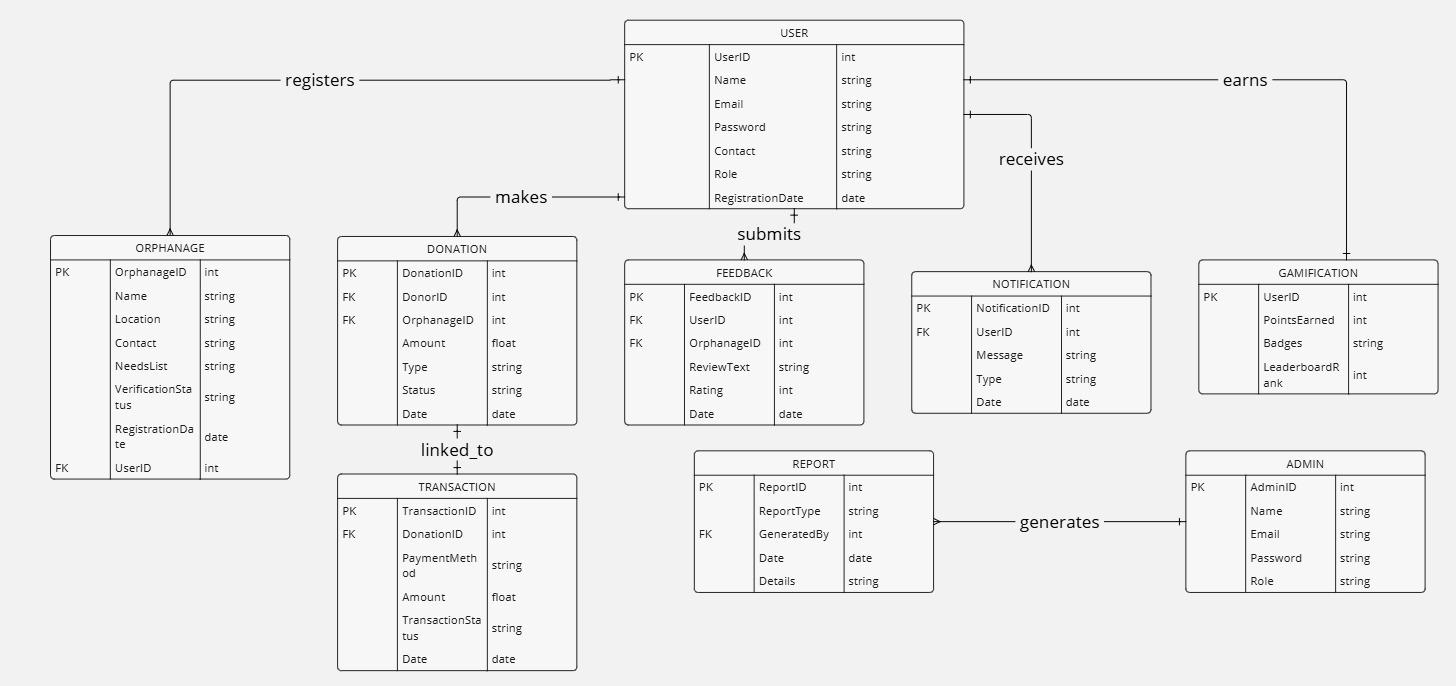


Figure 17 ER Diagram

## ****Data Flow Diagram (DFD)****

The **Data Flow Diagram (DFD)** for **Safe Haven** illustrates the flow of information within the system, highlighting how data moves between users, processes, and data stores. It helps in understanding the functional aspects of the system by visualizing inputs, processes, and outputs.

A diagram of a company

AI-generated content may be incorrect.

Figure 18 Data Flow Diagram

## Data Dictionary

* A**lphabetical List of System Entities or Major Data with Types and Descriptions:**

Table 22 Alphabetical List of System Entities

|  |  |
| --- | --- |
| **Terminology** | **Description (Data Type, Role, or Purpose)** |
| **Admin** | Entity representing system administrators who manage users, donations, and orphanages. |
| **Badges** | Achievements earned by users through engagement (String, Gamification). |
| **DonationStatus** | Status of a donation (Pending, Completed, Failed) (String). |
| **DonorID** | Foreign key linking a donation to the donor (Integer, FK). |
| **Feedback** | Entity storing user reviews and ratings for orphanages. |
| **Gamification** | Entity storing user engagement rewards like points and badges. |
| **LeaderboardRank** | User’s ranking based on engagement points (Integer, Gamification). |
| **NeedsList** | List of items or services requested by an orphanage (Text). |
| **NotificationID** | Unique identifier for each notification (Integer, Primary Key). |
| **OrphanageID** | Unique identifier for each orphanage (Integer, Primary Key). |
| **PointsEarned** | Points collected by users for donations and interactions (Integer, Gamification). |
| **RegistrationDate** | Timestamp when a user or orphanage was registered (DateTime). |
| **ReportID** | Unique identifier for each report (Integer, Primary Key). |
| **ReportType** | Specifies the type of report (Donations, Orphanages, etc.) (String). |
| **ReviewText** | User submitted feedback text for an orphanage (String). |
| **TransactionID** | Unique identifier for each transaction (Integer, Primary Key). |
| **TransactionStatus** | Status of a payment transaction (Success, Pending, Failed) (String). |
| **UserID** | Unique identifier for each user (Integer, Primary Key). |
| **VerificationStatus** | Status indicating whether an orphanage is verified (Boolean). |

* **Structured Approach: Functions and Function Parameters:**

Table 23 Structured Approach: Functions and Function Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Function Name** | **Parameters** | **Data Type** | **Description** |
| registerUser | name, email, password | String, String, String | Registers a new user (donor, orphanage admin). |
| loginUser | email, password | String, String | Authenticates user login. |
| searchOrphanage | location, category | String, String | Searches orphanages based on filters like location or type of needs. |
| makeDonation | donorID, orphanageID, amount, type | String, String, Float, String | Processes a donation (monetary or item based). |
| trackDonation | donationID | String | Tracks the status of a donation. |
| submitFeedback | userID, orphanageID, review, rating | String, String, String, Int | Submits feedback about an orphanage, including ratings. |
| sendNotification | userID, message | String, String | Sends notifications to users for updates. |
| processPayment | donationID, paymentMethod, amount | String, String, Float | Processes a donation payment securely. |
| updateProfile | userID, name, email, contact | String, String, | Updates user profile details. |

* **Object-Oriented (OO) Approach: Objects, Attributes, Methods, & Method Parameters:**

Table 24 ObjectOriented (OO) Approach

|  |  |  |  |
| --- | --- | --- | --- |
| **Object** | **Attribute** | **Methods** | **Method parameters** |
| Orphanage | |  | | --- | | orphanageID, name, location, needs | | requestDonations() | orphanageID, |
| Donation | |  | | --- | | donationID, donorID, orphanageID, amount | | makeDonation(), track donation() | donorID, orphanageID, amount |
| Review | |  |  | | --- | --- | |  | reviewID, userID, orphanageID, content | | submitFeedback | userID, orphanageID, review |
| Notification | |  | | --- | | notificationID, userID, message | | sendNotification() | userID, message |

# User Interface Design

The user interface design of Safe Haven is a visual blueprint that outlines the structure and layout of the platform's key pages, focusing on user experience (UX) and functionality without incorporating design elements like colors or images. It helps map out how users will navigate the site and interact with its features.

## Components

* 1. **Home Page:**
  + Header: Logo, Navigation Bar (Home, Orphanages, Donate, About Us, Contact)
  + Hero Section: Banner with mission statement and a collocation (e.g., Find Orphanages or Donate Now)
  + Features Overview: Brief on key platform features (donation tracking, orphanage locator)
  + Footer: Social media links, contact info, terms, and privacy policy
  1. **Profile Page:**
  + Orphanage Details: Name, location, verified badge, and description
  + Needs List: Items or services required (with donation buttons)
  + Media Section: Images and videos showcasing the orphanage
  + Reviews & Ratings: User generated reviews and star ratings
  1. **Donation Page:**
  + Donation Options: Onetime, recurring, or targeted donations (e.g., education, food)
  + Secure Payment Gateway: Form for payment details with encryption indicators
  + Donation Tracking: Progress bar showing fundraising goals and impact
  1. **About Us:**
* **Hero Section:** A powerful image/video with a mission driven tagline and a Donate Now button.
* **Testimonials:** Real stories from donors and orphanages in a carousel or grid format.
* **CalltoAction & Footer:** Encouraging visitors to donate, along with quick links and social media icons.
  1. **Subscription Page**
* **Header:** Logo, Navigation Menu (Home | About | Orphanages | Donate | Contact), Login/Sign Up Button
* **Subscription Plan Selection:** Title, Short description
  1. **FAQ Page**
* **Header:** Contains the website’s logo, navigation bar, and a search bar for quick FAQ lookups.
* **FAQ Categories:** Sections such as General Information, Donations, Orphanage Registration, Security & Privacy, and Technical Support**.**
* **Expandable Questions:** Each question can be clicked to reveal a concise answer, ensuring a clutter free interface.
* **Contact Support Section:** If users cannot find their answers, they can access customer support via live chat, email, or a contact form**.**

## Wireframe Of Each Component

**Home Page:**

|  |  |
| --- | --- |
|  | * Header: Logo, Navigation Bar (Home, Orphanages, Donate, About Us, Contact) * Hero Section: Banner with mission statement and a call-to-action (e.g., Find Orphanages or Donate Now) * Features Overview: Brief on key platform features (donation tracking, orphanage locator) * Footer: Social media links, contact info, terms, and privacy policy   Figure 19 home page |

**Profile Page:**

|  |  |
| --- | --- |
|  | * + Orphanage Details: Name, location, verified badge, and description   + Needs List: Items or services required (with donation buttons)   + Media Section: Images and videos showcasing the orphanage   + Reviews & Ratings: User generated reviews and star ratings   Figure 20 Profile Page: |

**About Us**

|  |  |
| --- | --- |
|  | * **Hero Section:** A powerful image/video with a mission driven tagline and a Donate Now button. * **Testimonials:** Real stories from donors and orphanages in a carousel or grid format. * **Call-to-Action & Footer:** Encouraging visitors to donate, along with quick links and social media icons.   Figure 21 About Us |

**Subscription Page**

|  |  |
| --- | --- |
|  | * **Header:** Logo, Navigation Menu (Home | About | Orphanages | Donate | Contact), Login/Sign Up Button * **Subscription Plan Selection:** Title, Short description   Figure 22 Subscription Page |

**FAQ Page**

|  |  |
| --- | --- |
| **A screenshot of a web page  AI-generated content may be incorrect.** | * **Header:** Contains the website’s logo, navigation bar, and a search bar for quick FAQ lookups. * **FAQ Categories:** Sections such as General Information, Donations, Orphanage Registration, Security & Privacy, and Technical Support**.** * **Expandable Questions:** Each question can be clicked to reveal a concise answer, ensuring a clutter free interface. * **Contact Support Section:** If users cannot find their answers, they can access customer support via live chat, email, or a contact form**.**   Figure 23 FAQ Page |

**Donation Page:**

|  |  |
| --- | --- |
|  | Donation Options: Onetime, recurring, or targeted donations (e.g., education, food)  Secure Payment Gateway: Form for payment details with encryption indicators  Donation Tracking: Progress bar showing fundraising goals and impact  Figure 24 Donation Page: |

## Screen Objects and Actions

**Use Case 1: User Registration & Login**

**Screen: Registration & Login Page**

Table 25 Screen: Registration & Login Page

|  |  |  |
| --- | --- | --- |
| **Screen Object** | **Description** | **Actions** |
| **Input Fields** (Name, Email, Password, Confirm Password) | Text boxes allow users to enter registration details. | User types required details.   The system validates inputs. |
| **Radio Button** (User Type Selection) | Allows users to select their role (Donor or Orphanage Admin). | The user selects a role.   UI updates based on selection. |
| **Submit Button** (Register) | Triggers the registration process. | The system verifies input data.   If successful, the account is created, and a confirmation message is displayed. |
| **Login Fields** (Email, Password) | Text boxes for logging into an existing account. | The user enters login credentials.   The system authenticates the user and redirects to the dashboard. |
| **Login Button** | Triggers the authentication process. | Validates credentials.   Redirects to appropriate dashboard based on user type. |
| **Forgot Password Link** | Allows users to recover their password. | The user clicks a link.   The system sends password reset instructions via email. |

**Use Case 2: Making a Donation**

**Screen: Donation Page**

Table 26 Screen: Donation Page

|  |  |  |
| --- | --- | --- |
| **Screen Object** | **Description** | **Actions** |
| **Dropdown Menu** (Select Orphanage) | Lists available in orphanages. | The user selects an orphanage from the list. |
| **Radio Button** (Donation Type) | Allows users to choose between Monetary or Item Based donations. | The user selects donation type, triggering different UI options. |
| **Input Field** (Amount) | Displays input field when Monetary donation is selected. | User enters amount to donate.   System validates the input. |
| **File Upload** (Upload Item Image) | Allows users to upload an image of the item they want to donate. | User selects and uploads an image. |
| **Textarea** (Item Description) | Allows users to describe the item being donated. | User enters item details. |
| **Payment Method Dropdown** | Lists available payment options for monetary donations. | User selects a preferred payment method. |
| **Donate Button** | Confirms and processes the donation. | System processes the donation.   Sends confirmation email and updates donation history. |
| **Cancel Button** | Allows users to cancel the donation process. | Clears all inputs and returns to the previous screen. |

# Design Decisions

1. **Architecture:** Used MVC Pattern for separation of concerns, scalability, and maintainability.
2. **Design Patterns:** Applied Factory, Singleton, Observer, and Strategy patterns for efficient code management.
3. **Database Normalization:** Followed 3NF to reduce redundancy and improve query performance.
4. **Algorithms:** Implemented Full Text Search for orphanage filtering and a Rule-Based Matching Algorithm for donation recommendations.
5. **Security:** UsedAES256 encryption and crypt hashing for secure authentication and payments.
6. **Frontend & Backend:** Choose React.js for a dynamic UI and Node.js for scalable backend operations.
7. **Payment Integration:** Integrated JazzCash API for secure and flexible donation processing.
8. **Notifications:** Used Firebase Cloud Messaging (FCM) for real-time updates.

# Summary

This chapter outlined the key design elements of the Safe Haven system, including class diagrams, state diagrams, and structured functions, ensuring a well-architected solution. The design phase involved critical decisions, such as adopting an object-oriented approach to ensure modularity and scalability. Various design patterns, including MVC (Model View Controller), were implemented for the separation of concerns, improving maintainability.

Key system components, such as user authentication, orphanage management, donation processing, and reporting modules, were refined to align with project objectives. Normalization techniques were applied to the database to enhance efficiency, and security measures were incorporated to protect sensitive user and transaction data.

**A blue heart with a hand holding a heart

AI-generated content may be incorrect.**

# CHAPTER 4

# Implementation

# 4.1 Algorithm

This section outlines the core algorithms implemented in the Safe Haven platform. Each algorithm addresses a major system function such as donation processing, orphanage filtering, review sorting, and secure login. These algorithms help ensure the system is efficient, reliable, and user-friendly.

## 4.1.1 Core Algorithms Used

### Algorithm 1: Orphanage Search with Filters

**Aspect Details**

**Input:** Location, category, and need type

**Output:** List of matched orphanages

**Complexity:**  Time: O(n), Space: O(k)

**Pseudocode:**

Table 27 Orphanage Search with Filters

|  |
| --- |
| 1: filteredList ← empty list |
| 2: For each orphanage in orphanageDatabase do |
| 3: If orphanage.location == userLocation AND orphanage.category == selectedCategory then |
| 4: Add orphanage to filteredList |
| 5: Return filteredList |

### Algorithm 2: Donation Processing

**Aspect Details**

**Input:** Donor ID, Orphanage ID, Amount

**Output:** Success or Failure message

**Complexity:** Time: O(1), Space: O(1)

**Pseudocode:**

|  |
| --- |
| 1: If validateUser(donorID) AND validateOrphanage(orphanageID) then |
| 2: status ← processPayment(donorID, amount) |
| 3: If status == success then |
| 4: updateDonationRecord(donorID, orphanageID, amount) |
| 5: sendConfirmation(donorID, orphanageID) |
| 6: Return "Donation Successful" |
| 7: Else |
| 8: Return "Payment Failed" |
| 9: Else |
| 10: Return "Invalid Donor or Orphanage" |

Table 28 Donation Processing

### Algorithm 3: Review & Rating Filter

**Aspect Details**

**Input:** Orphanage ID

**Output:** Sorted list of reviews

**Complexity:** Time: O(n log n), Space: O(n)

**Pseudocode:**

|  |
| --- |
| 1: reviews ← getReviews(orphanageID) |
| 2: sortedReviews ← sort |
| 3: Return sortedReviews |

Table 29 Review & Rating Filter

### Algorithm 4: Secure Login Authentication

**Aspect Details**

**Input:** Email and password

**Output:** Login success or error

**Complexity:** Time: O(1), Space: O(1)

**Pseudocode:**

|  |
| --- |
| 1: user ← findUserByEmail(inputEmail) |
| 2: If user.password == hash(inputPassword) then |
| 3: Return "Login Successful" |
| 4: Else |
| 5: Return "Invalid Credentials" |

Table 30 Secure Login Authentication

### Algorithm 5: Location-Based Orphanage Suggestions

**Aspect Details**

**Input:** User location

**Output:** Nearby orphanages

**Complexity:** Time: O(n), Space: O(k)

**Pseudocode:**

|  |
| --- |
| 1: For each orphanage in orphanageDatabase do |
| 2: distance ← calculateDistance(userLocation, orphanage.location) |
| 3: If distance < threshold then |
| 4: addToSuggestedList(orphanage) |
| 5: Return suggestedList |

Table 31 Location-Based Orphanage Suggestions

### Algorithm 6: Notification Trigger System

**Aspect Details**

**Input:** Orphanage need status

**Output:** Notifications to relevant users

**Complexity:** Time: O(n), Space: O(1)

**Pseudocode:**

|  |
| --- |
| 1: For each user in subscribedUsers do |
| 2: If user.location == orphanage.location then |
| 3: sendNotification(user, orphanage.need) |

Table 32 Notification Trigger System

### Algorithm 7: Monthly Report Generator

**Aspect Details**

**Input:** Donation logs and orphanage data

**Output:** PDF or CSV report

**Complexity:** Time: O(n), Space: O(1)

**Pseudocode:**

|  |
| --- |
| 1: report ← empty |
| 2: For each donation in donationHistory do |
| 3: accumulateDonationStats() |
| 4: updateOrphanageNeeds() |
| 5: Return formattedReport |

Table 33 Monthly Report Generator

# 4.2 External APIs/SDKs

This section describes the third-party APIs and SDKs integrated into the Safe Haven, outlining their purpose, and the specific functionalities or modules where they are utilized. These external services are crucial for enabling key features such as secure payments, location-based services, and real-time notifications.

## 4.2.1 Details of APIs

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API and version** | **Description of API** | **Purpose of usage** | **API Endpoint/Function/Class in use** |
| **Google Maps API** | A web mapping service application programming interface. | To help users locate orphanages and display their geographical locations on an interactive map. | Location-based orphanage search getOrphanageLocations() renderMapWithOrphanages() |
| **EasyPaisa** | A mobile banking and payment service widely used in Pakistan. | To facilitate secure monetary donations within Pakistan, providing a local and trusted payment option. | Payment processing for monetary donations processPayment() verifyTransaction() |
| **JazzCash** | Another prominent mobile banking and payment service in Pakistan. | To facilitate secure monetary donations within Pakistan, offering an additional local and trusted payment option. | Payment processing for monetary donations processPayment() verifyTransaction() |
| **Nodemailer** | A module for Node.js applications specifically designed for sending emails. | For sending various email notifications, including user registration verifications, donation confirmations, and other critical updates to users. | Sending verification emails Sending donation confirmation emails sendEmail() |
| **One Signal** | A free push notification service for websites and mobile applications. | For delivering real-time alerts and notifications to users, enhancing engagement and keeping them informed about donations, needs, and updates. | Sending real-time push notifications to users sendPushNotification() |
| **Firebase** | A comprehensive platform developed by Google for creating mobile and web applications. | Primarily utilized for its authentication services, ensuring secure user login and account management, and for real-time updates through Firebase Cloud Messaging (FCM). | User authentication authenticateUser()  Firebase Cloud Messaging (FCM) for real-time updates |

Table 34 Details of APIs

# 4.3 Code Repository

# 4.4 Summary

This chapter explores the core algorithms that underpin the Safe Haven platform, addressing critical functionalities such as orphanage search with filters, donation processing, review sorting, secure login, and location-based suggestions. These algorithms are designed to optimize performance, ensure data integrity, and enhance user interaction with the system.

Furthermore, the chapter highlighted the integration of key third-party APIs and SDKs—including Google Maps, EasyPaisa, JazzCash, Firebase, Nodemailer, and One Signal—which facilitate vital features like interactive mapping, secure and local payment processing, user authentication, email communication, and real-time notifications.

References

|  |
| --- |
|  |

**Webpages :**

|  |  |
| --- | --- |
|  | *Canva.com*. [Online]. Available: <https://www.canva.com>/. |
|  | Opensource tool that uses simple textual descriptions to draw beautiful UML diagrams, [*Plantuml.com*](file:///C:\Users\ayesh\AppData\Roaming\Microsoft\Word\Plantuml.com). [Online]. Available: https://plantuml.com/. |
|  | Material Design, *Material Design*. [Online]. Available: [https://material.io/design.](https://material.io/design.%20) |
|  | Web Content Accessibility Guidelines (WCAG) 2.1, *Www.w3.org*. [Online]. Available: [https://www.w3.org/TR/WCAG21/.](https://www.w3.org/TR/WCAG21/.%20) |
|  | Google Maps Platform, *Google for Developers*. [Online]. Available: <https://developers.google.com/maps>. |
|  | Figma: The Collaborative Interface Design Tool, *Figma*. [Online]. Available: <http://figma.com>/. |
|  | AI ERD diagram maker, *https://miro.com/*. [Online]. Available: [https://miro.com/ai/erddiagramai/.](https://miro.com/ai/erd-diagram-ai/.%20) |
|  | DiagramGPT – AI diagram generator, *Eraser.io*. [Online]. Available: <https://www.eraser.io/diagramgpt>. |
|  | Pakistan Sweet Home A heaven for the underprivileged, *Org.pk*. [Online]. Available: <https://www.pakistansweethome.org.pk>/. |
|  | sos children villages pakistan, SOS Children’s Villages Pakistan, *Org.pk*. [Online]. Available: <https://www.sos.org.pk>/. |
|  | Africa’s orphans support platform, *Africasorphans.org*. [Online]. Available: [http://www.africasorphans.org/.](http://www.africasorphans.org/.%20) |
|  | Miracle Foundation, *Miracle Foundation*, 05Apr2023. [Online]. Available: [https://www.miraclefoundation.org/.](https://www.miraclefoundation.org/.%20) |
|  | *Theworldorphanfund.org*. [Online]. Available: [https://www.theworldorphanfund.org/.](https://www.theworldorphanfund.org/.%20) |
|  | Alkhidmat foundation’s orphan care program, *Alkhidmat.org*. [Online]. Available: <https://alkhidmat.org/orphan>. |
|  | Orphan Care & education, *Khubaib Foundation*, 19Sep2022. [Online]. Available: [https://khubaibpakistan.org/orphancareprogram/.](https://khubaibpakistan.org/orphan-care-program/.%20) |
| [17] | Edhi welfare organization – serving humanity in the spirit of all religions, *Edhi.org*. [Online]. Available: [https://edhi.org/.](https://edhi.org/.%20) |
| [18] | *Almustafatrust.org*. [Online]. Available: [https://www.almustafatrust.org/appeals/sponsoranorphan/.](https://www.almustafatrust.org/appeals/sponsor-an-orphan/.%20) |
| [19] | Support the orphan children, *AlMustafa Welfare Society*, 03Apr2023. [Online]. Available: [https://almustafa.pk/donation/supporttheorphanchildren/.](https://almustafa.pk/donation/support-the-orphan-children/.%20) |

**BOOKS**

|  |  |
| --- | --- |
| [20] | *IEEE Standard for Software Design Descriptions*. IEEE Std, 2014. |
| [21] | R. C. Martin, *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, 2009. |
| [22] | P. Clements *et al.*, *Documenting software architectures: Views and beyond*. AddisonWesley Professional, 2011. |
| [23] | G. Booch, J. Rumbaugh, and I. Jacobson, *The unified modeling language user guide*, 2nd ed. Boston, MA: AddisonWesley Educational, 2005. |
| [24] | R. S. Pressman and B. R. Maxim, *Loose leaf for software engineering: A practitioner’s approach*, 9th ed. Columbus, OH: McGrawHill Education, 2019. |
| [25] | |  | | --- | |  | |  |   Sommerville, *Engineering software products: An introduction to modern software engineering, global edition Pearson eText (OLP)*. London, England: Pearson Education, 2023. |

**FIGMA**

[**https://www.figma.com/design/j5kDyBAepqwNdj7XaolBne/Untitled?node-id=209-830&t=15Jg6Lgwmx10qiph-0**](https://www.figma.com/design/j5kDyBAepqwNdj7XaolBne/Untitled?node-id=209-830&t=15Jg6Lgwmx10qiph-0)

ChatGPT, *Chatgpt.com*. [Online]. Available: [https://chatgpt.com/c/677aa2daf1d8800aadc020e9b50dd91f.](https://chatgpt.com/c/677aa2daf1d8800aadc020e9b50dd91f.%20)

**Diagrams**

[https://lucid.app/lucidchart/12a8bd00-0a12-4a73-b250 89208f4dd6f1/edit?invitationId=inv\_e8e114ec-4af9-4549-8121-ed3baea02abf&page=0\_0#](https://lucid.app/lucidchart/12a8bd00-0a12-4a73-b250%2089208f4dd6f1/edit?invitationId=inv_e8e114ec-4af9-4549-8121-ed3baea02abf&page=0_0)

<https://miro.com/app/board/uXjVIbaQBnw=/>

# Appendix A

# Use case Description

# Use Case Analysis

Table 35 User Registration & Authentication

|  |  |
| --- | --- |
| **UC Identifier** | **UC1** |
| **Requirements Traceability** | FR1 (User Registration & Authentication) |
| **Purpose** | Allow users (donors, orphanages, and admins) to register and log into the system securely. |
| **Priority** | High |
| **Preconditions** | Users must have a valid email address or social media account. |
| **Postconditions** | A user account is created, and the user is logged in. |
| **Actors** | Donor, Orphanage, Admin |
| **Extends** | N/A |
| **Main Success Scenario** | 1. User navigates to the registration page.  2. User fills in personal details and selects the user type.  3. User submits the registration form.  4. System sends a verification email.  5. User verifies email and logs in successfully. |
| **Alternate Flows** | 1a. User opts for social login via Google/Facebook.  2a. System redirects to social authentication service.  3a. Upon successful authentication, system logs in the user. |
| **Exceptions** | E1: If the email is already registered, the system displays an error message.  E2: If the verification email is not received, the user can request a resend. |

Table 36 Organization Registration

|  |  |
| --- | --- |
| **UC Identifier** | **UC2** |
| **Requirements Traceability** | FR3 (Organization Registration) |
| **Purpose** | Allow orphanages and donation centers to register, verify their authenticity, and create profiles. |
| **Priority** | High |
| **Preconditions** | Orphanage must provide valid registration details and documents. |
| **Postconditions** | Orphanage profile is created and displayed on the platform. |
| **Actors** | Orphanage Representative, Admin |
| **Extends** | UC1 (User Registration & Authentication) |
| **Main Success Scenario** | 1. Orphanage representative fills out the registration form.  2. Uploads verification documents.  3. Submit the form.  4. Admin reviews and approves the registration.  5. Organization profile is activated. |
| **Alternate Flows** | 4a. If documents are unclear, the admin requests additional verification.  4b. Orphanage resubmits documents. |
| **Exceptions** | E1: If invalid documents are uploaded, the system rejects the registration.  E2: If the organization is already registered, the system displays an error. |
| **Includes** | UC3 (Admin Verification) |

Table 37 Donation Management

|  |  |
| --- | --- |
| **UC Identifier** | **UC3** |
| **Requirements Traceability** | FR7 (Donation Management) |
| **Purpose** | Enable donors to contribute monetary and in-kind donations to orphanages. |
| **Priority** | High |
| **Preconditions** | Donors must be logged in and have valid payment details (if making a monetary donation). |
| **Postconditions** | The donation is recorded, and the donor receives confirmation. |
| **Actors** | Donor, Orphanage Representative, Admin |
| **Extends** | N/A |
| **Main Success Scenario** | 1. Donor selects an orphanage.  2. Chooses a donation type (money or items).  3. Enters donation details.  4. The system processes the transaction.  5. Orphanage receives donation notification.  6. System updates donation tracking. |
| **Alternate Flows** | 3a. Donor cancels the donation before submission.  4a. If donating items, donor arranges delivery. |
| **Exceptions** | E1: Payment failure due to insufficient funds.  E2: System rejects donation if required details are missing. |
| **Includes** | UC4 (Transaction Processing) |

Table 38 Gamification Features

|  |  |
| --- | --- |
| **UC Identifier** | **UC4** |
| **Requirements Traceability** | FR2 (Gamification Features) |
| **Purpose** | Encourage donor engagement through badges, points, and leaderboards. |
| **Priority** | Low |
| **Preconditions** | User must have performed donation activities. |
| **Postconditions** | User receives badges/points based on contributions. |
| **Actors** | Donor |
| **Extends** | N/A |
| **Main Success Scenario** | 1. User donates. 2. System tracks the activity.  3. User earns points and badges.  4. Leaderboard updates. |
| **Alternate Flows** | 3a. If a user reaches a milestone, system sends an achievement notification. |
| **Exceptions** | E1: System fails to update points due to a tracking issue. |
| **Includes** | UC6 (Notifications & Alerts) |

Table 39 Location Based Orphanage Finder

|  |  |
| --- | --- |
| **UC Identifier** | **UC5** |
| **Requirements Traceability** | FR14 (Location Based Orphanage Finder) |
| **Purpose** | Enable users to find nearby orphanages based on location. |
| **Priority** | High |
| **Preconditions** | User must grant location access or enter a location manually. |
| **Postconditions** | System displays orphanages near the user. |
| **Actors** | Donor |
| **Extends** | N/A |
| **Main Success Scenario** | 1. User accesses the orphanage finder.  2. Allows location access or enters a location manually.  3. System retrieves orphanage data from the database.  4. Displays results on a map. |
| **Alternate Flows** | 2a. User denies location access, enters location manually. |
| **Exceptions** | E1: No orphanages found in the entered location.  E2: Location services fail to load. |
| **Includes** | UC7 (Search & Filters) |

# ****Use Case Diagram:****

**Actors:**

* + Donor
  + Orphanage Admin
  + Platform Admin
  + Guest User

**Classes:**

* + **User** (Attributes: userID, name, email, role, password)
  + **Donor** (Inherits User)
  + **OrphanageAdmin** (Inherits User)
  + **Orphanage** (Attributes: orphanageID, name, location, description, needsList)
  + **Donation** (Attributes: donationID, donorID, orphanageID, amount/type, date)
  + **Payment** (Attributes: paymentID, method (Easypaisa/JazzCash), status, transactionID)

## Use Case Diagram for Each Actor

**Donor Use Cases**

A diagram of a person

AI-generated content may be incorrect.

Figure 25 Donor Use Cases

* + Register/Login
  + Search Orphanages
  + View Orphanage Profile
  + Make a Donation
  + Track Donation History
  + Receive Notifications

**Orphanage Admin Use Cases**

A diagram of a person with many circles

AI-generated content may be incorrect.

Figure 26 Orphanage Admin Use Cases

* + - Register/Login
    - Manage Orphanage Profile
    - List Needs
    - Confirm Received Donations

**System Admin Use Cases**

A diagram of a diagram

AI-generated content may be incorrect.

Figure 27 System Admin Use Cases

* + - Manage Users
    - Monitor Transactions
    - Handle Reports/Complaints
    - Manage Platform Content

**Guest User Use Cases**

A diagram of a person's work flow

AI-generated content may be incorrect.

Figure 28 Guest User Use Cases

* + - Register as a Donor
    - Browse the orphanage
    - Read the FAQ