Chapter 0
Synopsis

Title of the project
Volunteer Management

Statement about the problem
Organizations and NGOs often struggle with efficiently managing volunteer engagement for their social welfare programs. Without a centralized platform, it's challenging for them to effectively communicate upcoming events and opportunities to potential volunteers. Similarly, volunteers face difficulties in discovering relevant events and coordinating their participation due to the lack of a centralized resource. This fragmented approach leads to missed opportunities, decreased volunteer engagement, and inefficient utilization of resources for social welfare initiatives.

Why this topic?
Creating a Volunteer Management System for social welfare programs addresses a critical need in our community. By providing a centralized platform for organizations, NGOs, and volunteers, we aim to bridge the gap between those in need of support and those willing to contribute their time and skills. This project aligns with the values of community engagement, social responsibility, and effective resource utilization, making it a compelling and impactful endeavor. Through this system, we can streamline volunteer engagement, enhance the effectiveness of social welfare programs, and ultimately contribute to positive social change.

Objective and Scope

Scope:

The scope of the system is to develop a robust volunteer management system that facilitates seamless communication and co-ordination between organisations , NGO's and volunteers. The scope of the system is limited to Mumbai region, catering to organisations of various type including Non-Profit, Commercial, Educational, Government organisations, Healthcare, Charitable, Religious, Social organisations. This ensures that the system is tailored to specific needs and context of Mumbai community while being inclusive and accessible to a diverse range of organisations seeking volunteer support for their social welfare programs.

Objectives:

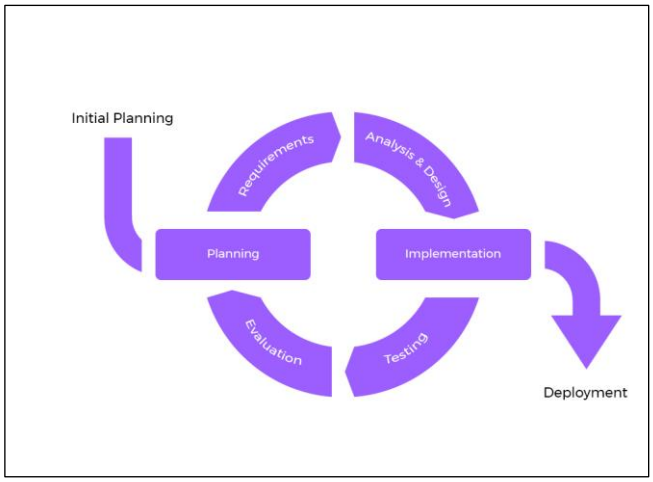
i. Provide a user-friendly interface for organisations and NGO's to post events , manage Volunteer requirements , and track volunteer engagement.
ii. Enable volunteers to easily discover , apply for, and participate in relevant social welfare events based on their interests, skills and availability.
iii. Improve efficiency in volunteer management , event co-ordination and resource allocation for social welfare initiatives.
iv. Foster a sense of community , collaboration and social impact among volunteers and organisations

v. Improving social interactions between volunteer by allowing them to participate and communicate with a diverse variety of people.

Methodology

For this System, Iterative development model would be more suitable because of the following reasons:

- Will allow continuous refinement and improvement based on feedback and evolving requirement.
- Offers flexibility in accommodating new features , addressing issues and incorporating user feedback at various stages.
- Encourages active involvement of stakeholders

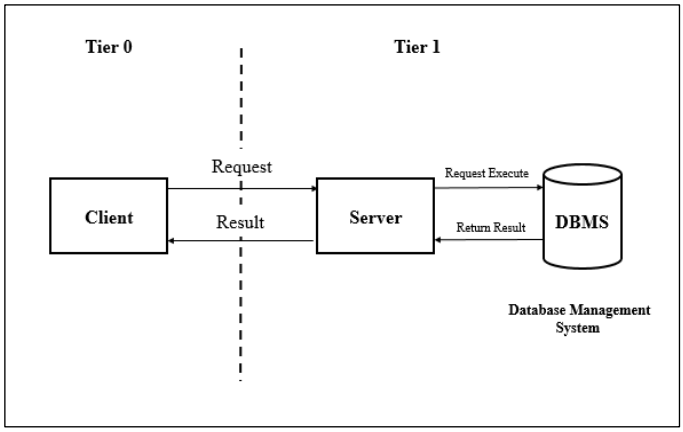


Proposed Architecture

The Project would be developed using a Client-Server 2 Tier Architecture, means organizing website in 2 main parts:

i. Client:
The User Interface and Presentation of the system will run on user's device. Client is responsible for initiating request to server.

ii. Server:
The Server component hosts the Application Processing and Data Storage, It responds to requests by processing the requests.



Requirements

1. Software Requirements:
    a. Database: MySQL
    b. Back-End: PHP
    c. Front-End: HTML,CSS, Javascript (Jquery), Bootstrap, AJAX

2. Hardware Requirements:
    a. A Basic computer with an decent processor such as intel i3 10^{th} Gen or 11^{th} Gen or AMD Ryzon 3 Processor
    b. Ram: Minimum 4GB
    c. HardDisk: 128GB or 256GB SSD, 4GB of available space
    d. Monitor: 1200x800 Minimum screen resolution

3. Platform:
    a. Visual Studio Code (VS Code)

Contribution To Society:

The Volunteer Management System contributes significantly to society in several ways:

1. Facilitating volunteer engagement and collaboration among organizations, NGOs, and volunteers.

2. Optimizing resource allocation for social welfare programs, enhancing their impact.

3. Improving communication, coordination, and feedback mechanisms for better program outcomes.

4. Fostering a culture of collaboration, knowledge sharing, and collective action within communities.

5. Empowering individuals to contribute meaningfully to social causes, promoting civic engagement and social responsibility.

Conclusion:

In conclusion, the development of a Volunteer Management System represents a significant step towards fostering a more connected, efficient, and impactful approach to social welfare initiatives. By bringing together volunteers, organizations, and NGOs on a centralized platform, the system streamlines volunteer engagement, optimizes resource utilization, and enhances the overall effectiveness of social welfare programs.

Through improved communication, collaboration, and empowerment of individuals, the system contributes to positive social change, stronger communities, and a more inclusive society.

Chapter 1

Introduction

1.1 Background

Traditionally, organizations have relied on methods such as word-of-mouth, flyers, local media advertisements, volunteer fairs, and referrals to recruit volunteers for their social welfare programs. While these methods are still relevant, they face several challenges in effectively managing volunteer engagement. These challenges include limited reach, resource-intensive processes, lack of centralization in information, communication barriers, and difficulties in tracking and reporting volunteer activities.

In response to these challenges, the proposed Volunteer Management System website aims to revolutionize volunteer hiring and management practices. By centralizing information about volunteer opportunities, providing efficient communication tools, automating processes such as volunteer matching and application management, implementing robust data tracking and reporting features, and expanding reach to a wider audience of potential volunteers, the website addresses the shortcomings of traditional methods. It streamlines processes, improves communication and coordination, enhances the volunteer experience, and ultimately contributes to more effective and impactful social welfare initiatives in the Mumbai region.

1.2 Objectives

Provide a centralized platform for organizations and NGOs to post volunteer opportunities for social welfare programs in the Mumbai region.

i. Provide a centralized platform for organizations and NGOs to post volunteer opportunities for social welfare programs in the Mumbai region.
ii. Enable volunteers to easily find and apply for relevant volunteer positions based on their skills, interests, and availability.
iii. Implement efficient communication tools to facilitate seamless interaction between organizations and volunteers, improving coordination and reducing communication barriers.
iv. Expand the reach of volunteer opportunities to a wider audience, including non-profit, commercial, educational, and other organizations, fostering collaboration and collective impact for positive social change.
v. To reduce the time and efforts to search and recruit potential volunteer for a social welfare program by organizations.

1.3 Scope, Purpose and Applicability

1.1 Purpose:

The purpose of the Volunteer Management System website is to streamline volunteer engagement, enhance communication between organizations and volunteers, and Encourage community involvement and social responsibility by providing individuals and organizations with opportunities to contribute meaningfully to social causes and make a positive difference in their communities.

1.2 Scope:

The scope of the system is to develop a robust volunteer management system that facilitates seamless communication and co-ordination between organizations of various types such as non-profit, commercial, educational, government, religious organisations and volunteer's for social welfare programs across Mumbai region.

1.3 Applicability:

The applicability of the Volunteer Management System website extends to various types of organizations and entities in the Mumbai region, including:

1. **Non-Profit Organizations:** Non-profit organizations focused on social causes such as education, healthcare, environmental conservation, and community development can use the platform to recruit volunteers for their programs and initiatives.
2. **Commercial Entities:** Commercial entities with corporate social responsibility (CSR) initiatives can leverage the platform to engage employees in volunteering activities, support local communities, and contribute to social welfare programs.
3. **Educational Institutions:** Schools, colleges, and universities can utilize the platform to involve students, faculty, and staff in community service projects, awareness campaigns, and educational outreach programs.
4. **Community Groups:** Local community groups, neighbourhood associations, and grassroots organizations can use the platform to organize volunteer-led activities, events, and campaigns for community improvement and social change.
5. **NGOs and Charitable Foundations:** NGOs, charitable foundations, and advocacy groups working on various social issues such as poverty alleviation, human rights, and disaster relief can utilize the platform to recruit volunteers, coordinate relief efforts, and raise awareness.

The Volunteer Management System website is designed to be adaptable and accessible to a wide range of organizations and entities, offering a centralized platform for efficient volunteer management, collaboration, and collective impact in addressing societal challenges and promoting positive social outcomes in the Mumbai region.

Chapter 2

Survey of Technology

Front End Languages:

- **HTML5 :**
  HTML5 is a markup language used for structuring and presenting content on the World Wide Web. HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves, and rationalizes the markup available for documents and introduces markup and application programming interfaces (APIs) for complex web applications.

- **JavaScript:**
  JavaScript is a versatile programming language used for creating interactive and dynamic web content. It runs on the client side, enabling functionalities like form validation, DOM manipulation, and AJAX requests. JavaScript is essential for web development as it adds interactivity to websites, creates responsive user interfaces, handles events, and communicates with servers asynchronously.

- **JQuery:**
  jQuery is a fast, lightweight, and feature-rich JavaScript library that simplifies DOM manipulation, event handling, and AJAX interactions. It provides a concise and efficient way to write JavaScript code by streamlining common tasks like selecting elements, animating content, making AJAX requests, and handling events across different browsers. jQuery abstracts complex JavaScript functionalities into simple methods, improving code readability and reducing development time.

- **CSS (Cascading Style Sheets):**
  CSS is a style sheet language used for styling and formatting web documents written in HTML or XML. It controls the visual presentation of web pages, including layout, colors, fonts, and spacing. CSS separates content from design, allowing developers to create visually appealing and responsive layouts by applying styles to HTML elements, classes, IDs, and pseudo-classes.

- **Tailwind CSS:**
  Tailwind CSS is a utility-first CSS framework that provides a set of pre-designed utility classes to style HTML elements. It focuses on rapid development and customization without writing custom CSS. Tailwind CSS allows developers to apply styles directly in HTML using classes like `bg-blue-500`, `text-lg`, `p-4`, defining background colors, text sizes, padding, margins, and more. It promotes a scalable and maintainable approach to styling web applications by composing utilities to create custom designs.

- **ASP.NET:**
  ASP.NET is a web application framework developed by Microsoft for building dynamic and scalable web applications, APIs, and services using languages like C# and VB.NET. It provides a robust set of tools, libraries, and APIs for web development, including MVC (Model-View-Controller) and Web Forms for building web applications, Web API for creating RESTful services, and ASP.NET Core for cross-platform development. ASP.NET supports features like authentication, authorization, routing, caching, and database access, making it a comprehensive platform for web development.

- **React**
  React is a JavaScript library for building user interfaces, primarily used for developing the front end of web applications. It allows developers to create reusable UI components and manage the state of the application efficiently. React follows a component-based architecture and uses a virtual DOM for optimal performance, enabling fast rendering and updates. It is maintained by Facebook and has a large and active community, along with extensive documentation and ecosystem of tools and libraries.

- **Angular:**
  Angular is a TypeScript-based web application framework maintained by Google, used for building dynamic and interactive single-page applications (SPAs). It follows the MVC (Model-View-Controller) architecture and provides features like data binding, dependency injection, routing, and form handling out of the box. Angular offers a comprehensive suite of tools and libraries for testing, debugging, and

optimizing applications, making it suitable for large-scale enterprise applications with complex requirements.

Back End Languages:

- **Next.js**
  Next.js is a React framework that provides functionality such as server-side rendering, routing, and generating static websites for React-based web applications. It simplifies the development process by offering built-in features like automatic code splitting, hot module replacement, and server-side rendering, enhancing performance and SEO. Next.js is highly flexible and scalable, allowing developers to build complex applications with ease while providing a great developer experience.

- **PHP:**
  PHP is a server-side scripting language designed for web development. It is widely used for creating dynamic web pages and web applications. PHP code is embedded within HTML, allowing developers to mix logic with presentation. It supports various databases and frameworks like Laravel, CodeIgniter, and Symfony, making it versatile for building robust backend systems.

- **Ruby:**
  Ruby is a dynamic, object-oriented programming language known for its simplicity and productivity. It is commonly used with the Ruby on Rails framework for web development. Ruby on Rails provides conventions over configurations, making it efficient for building database-backed web applications. Ruby's readability and expressiveness contribute to its popularity among developers.

- **Node.js**
  Node.js is a runtime environment that allows developers to run JavaScript code server-side. It uses an event-driven, non-blocking I/O model, making it lightweight and efficient for handling concurrent requests. Node.js is commonly used for building scalable, real-time web applications and APIs. It has a large ecosystem of packages available through npm (Node Package Manager).

- **Rust:**

Rust is a systems programming language known for its focus on safety, performance, and concurrency. While not as commonly used for traditional web development as other languages, Rust can be used for backend development, especially for performance-critical applications or services. Its strong type system and memory safety features make it suitable for building robust and secure systems.

- **Flask:**
  Flask is a lightweight Python web framework designed for simplicity and flexibility. It provides tools and libraries for building web applications, APIs, and microservices. Flask follows a minimalist approach, allowing developers to choose components and extensions as needed. It is often used for prototyping, small-scale projects, and rapid development of backend systems.

Database:

- **MySQL:**
  MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL). MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL provides an implementation of a SQL database very well suited for small to medium web pages. A database is just a structured collection of data that is organized for easy use and retrieval. Common applications for MySQL include php and java-based web applications that require a DB storage backend.

- **PostgreSQL:**
  PostgreSQL is a powerful, open-source object-relational database system known for its reliability, robustness, and extensibility. It uses SQL (Structured Query Language) for querying and managing data, offering features like transactions, foreign keys, triggers, and stored procedures. PostgreSQL supports various data types, including numeric, text, JSON, and arrays, making it versatile for different types of applications. It's widely used in enterprise-level applications due to its scalability, support for complex queries, and ACID compliance, ensuring data integrity.

- **MongoDB:**
  MongoDB is a popular NoSQL database that stores data in flexible, JSON-like documents, making it easy to store and manage unstructured

data. It offers high scalability, performance, and flexibility, making it suitable for handling large volumes of data in real-time applications. MongoDB's document model allows for quick and easy data retrieval, and it supports features like indexing, shading, and replication for scalability and fault tolerance. It's commonly used in modern web and mobile applications, IoT (Internet of Things), and analytics applications requiring flexible data storage and retrieval capabilities.

- **AWS Dynamo DB:**
  As part of a larger ecosystem of Amazon Web Services (AWS), this database is a NoSQL database that is known for its speed and efficiency when it comes to retrieval of information and data from the system. It is a key-value database, so there are variety of data types that can be stored within this system. It is a highly scalable and a complex database system for developers that are working with the applications that need to manage big user data and constant engagement.

Why these language?

The system will be developing using Html5, Tailwind CSS and Jquery for the user-friendly front-end, Php (Laravel) for the back-end, and MySQL for the database.

HTML , jQuery And Tailwind CSS

- **HTML:** provides the foundational structure of web pages, essential for any web application.
- **jQuery:** is essential for creating interactive and dynamic user interfaces. It simplifies JavaScript programming, making it easier to handle events, create animations, and manage HTML document traversing and manipulation.
- **Tailwind CSS:** is crucial for styling web pages, ensuring that the website is visually appealing and user-friendly. it allows for rapid development with its utility-first approach, providing a flexible and customizable design system that can be adapted to various project requirements.

Why PHP ?

It is a server-side scripting language that is particularly well-suited for web development. It is widely used due to its ability to interact with databases, handle sessions, and manage dynamic content. PHP is also compatible with

most web servers and databases, making it a versatile choice for backend development.

Why MySQL ?

It is chosen for its reliability, performance, and ease of use. It supports large-scale applications and can handle high volumes of data and transactions. MySQL's robust features, such as data security, backup, and recovery, make it a dependable choice for managing the database of the Volunteer Management System.

Chapter 3

Requirements and Its Analysis

3.2.2. Requirement Analysis

By analysing the Google Form responses, we gathered insightful data about the needs and preferences of both organizations and volunteers for the Volunteer Management System.

Organization Feedback:

Organizations mostly have difficulty in finding and recruiting volunteers for their events. Almost 90% of organizations prefer to switch to an online system for managing volunteer registrations. 50% of organizations currently do not have a dedicated platform for volunteer management.

Challenges faced by organizations include:

- Difficulty in reaching a wider audience.
- Inefficiencies in managing volunteer data manually.
- Lack of tools for effective communication with volunteers.

Additional desired features include:

- 75% want a communication platform to engage with volunteers.
- 70% want tools for feedback collection and reporting.

Volunteer Feedback:

- 75% of volunteers have experienced difficulty in finding suitable volunteer opportunities.
- 92% of volunteers prefer an online platform to browse and apply for volunteer opportunities.

Challenges faced by volunteers include:

- Lack of information about available opportunities.
- Difficulty in applying for events through traditional methods.
- Limited communication with organizations.

85% of volunteers want an easy-to-use platform for browsing and applying to events.
80% of volunteers want notifications for new volunteer opportunities.
Most of the volunteers desire a feature for receiving feedback and recognition from organizations.

Additional desired features include:

- 65% want to see reviews and ratings of organizations from past volunteers.

- 60% prefer a Communication Feature to Socialize with others volunteers

3.2.2.1 Functional Requirements

1. User Registration and Login:
   Organizations and volunteers must be able to register and create profiles. Secure login system for all users.

2. Volunteer Matching:
   Automated matching of volunteers to opportunities based on skills and interests.

3. Volunteering Event Management:
   - Organizations can create, manage, and edit event listings.
   - Volunteers can browse and apply for events.

4. Communication Tools:
   In-built messaging system for communication between organizations and volunteers.

5. Notifications and Alerts:
   Email and push notifications for new volunteer opportunities and updates.

7. Feedback and Reviews:
   - Volunteers can leave reviews and feedback for organizations.
   - Organizations can provide feedback and recognition to volunteers.

8. Mobile Compatibility:
   Mobile-friendly interface for accessing the platform on various devices.

9. Socialization Tool:

   Providing a features for volunteer and organisation to post images and comments related to volunteering activity.

3.2.2.2 Non-Functional Requirements

1. Scalability:
   The system must be able to handle an increasing number of users and data without performance degradation.

2. Security:
   - User data must be protected with encryption and secure authentication mechanisms.
   - Regular security audits to ensure the system's integrity.

3. Usability:
   - The platform should have an intuitive and user-friendly interface.
   - Clear navigation and accessible design for users.

4. Maintainability:
   - The codebase should be well-documented and modular to facilitate maintenance and updates.

6. Compatibility:
   - The system must compatible to all type of devices (Mobile phone, Laptop, Desktop PC etc)

3.2.2.3 System Requirement

1. **Registration:**
   Description: The user must first register an account in order to use the site.
   Input: Name, phone, email, password, confirm password.
   Source: User (Organization and Volunteer).
   Output: Input data is stored in the database.
   Destination: Database.

Action: After registration, an account for the user will be created.
Pre-condition: User should not have an existing account.
Post-condition: User can log in to the account with the registered email and password.
Exception: An error message is shown if any input is empty, email or phone number doesn't match the pattern, or if the account already exists.

2. **Login:**
   Description: The user must log in to access their account.
   Input: Email, password.
   Source: User (Organization and Volunteer).
   Output: User authentication.
   Destination: Database.
   Action: User is granted access to their account.
   Pre-condition: User must have a registered account.
   Post-condition: User is logged in and redirected to their dashboard.
   Exception: An error message is shown if the email or password is incorrect.

3. **Event Creation (For Organizations):**
   Description: Organizations can create new events.
   Input: Event name, description, date, time, location, number of volunteers needed, required skills.
   Source: Organization.
   Output: Event details are stored in the database.
   Destination: Database.
   Action: Event is created and listed on the platform.
   Pre-condition: Organization must be logged in.
   Post-condition: Event is visible to volunteers for application.
   Exception: An error message is shown if any input is empty or does not meet validation criteria.

4. **Event Browsing and Application (For Volunteers):**
   Description: Volunteers can browse and apply for events.
   Input: Event name or Type for browsing; Event ID for application.

Source: Volunteer.
Output: List of available events for browsing; application details stored in the database for event application.
Destination: Database.
Action: Volunteer can view event details and apply for events.
Pre-condition: Volunteer must be logged in.
Post-condition: Volunteer's application is recorded.
Exception: An error message is shown if the application cannot be processed.

5. **Volunteer Matching:**
   Description: The system automatically matches volunteers to events based on their skills and interests.
   Input: Volunteer profile information, event requirements.
   Source: Database.
   Output: List of matched events for volunteers.
   Destination: Volunteer's dashboard.
   Action: Volunteers are notified of suitable events.
   Pre-condition: Volunteer profile must be complete with skills and interests.
   Post-condition: Matched events are displayed to the volunteer.
   Exception: An error message is shown if no matching events are found.

6. **Communication Tool:**
   Description: Allows communication between organizations and volunteers.
   Input: Message content.
   Source: User (Organization and Volunteer).
   Output: Message is stored and displayed to the recipient.
   Destination: Database.
   Action: Message is sent and received.
   Pre-condition: Both users must be logged in.
   Post-condition: Messages are accessible in the user's inbox.
   Exception: An error message is shown if the message cannot be sent.

7. **Notifications:**

Description: Sends notifications to users about new events, applications, and updates.
Input: Event triggers (new event, application status, updates).
Source: System.
Output: Notification message.
Destination: User (Organization and Volunteer).
Action: Notification is sent via email or in-app notification.
Pre-condition: User must have notifications enabled.
Post-condition: User receives and views the notification.
Exception: An error message is shown if the notification cannot be sent.

8. **Review Posting and Viewing:**
   Description: Users can post and view reviews for events and organizations.
   Input: Review text, rating, event/organization ID.
   Source: User (Volunteer).
   Output: Review is stored in the database and displayed on the event/organization's profile.
   Destination: Database.
   Action: Review is saved and can be viewed by other users.
   Pre-condition: User must be logged in and have participated in the event.
   Post-condition: Review is visible to all users.
   Exception: An error message is shown if any input is empty or does not meet validation criteria.

9. **Event Details Updation (Change or Delete):**
   Description: Organizations can update or delete event details.
   Input: Event ID, updated event details (name, description, date, time, location, number of volunteers needed, required skills).
   Source: Organization.
   Output: Updated event details are stored in the database.
   Destination: Database.
   Action: Event details are updated or deleted from the platform.
   Pre-condition: Organization must be logged in and must be the creator of the event.

Post-condition: Updated event details are visible to volunteers; deleted events are removed from the listing.
Exception: An error message is shown if the update or delete action fails.

10. Profile Updation or Password Change:

Description: Users can update their profile information and change their passwords.
Input: Updated profile information (name, phone, email, etc.), old password, new password.
Source: User (Organization and Volunteer).
Output: Updated profile information and password are stored in the database.
Destination: Database.
Action: Profile information and password are updated.
Pre-condition: User must be logged in.
Post-condition: Updated profile information and password are saved.
Exception: An error message is shown if the input is invalid or if the old password does not match the current password.

11. Forgot Password (OTP Verification):

Description: Users can reset their passwords using OTP verification.
Input: Email, OTP, new password.
Source: User (Organization and Volunteer).
Output: OTP is sent to the user's email, new password is stored in the database.
Destination: User's email (for OTP), Database (for new password).
Action: OTP is sent to the user, and the user can reset their password using the OTP.
Pre-condition: User must provide a registered email.
Post-condition: Password is reset and the user can log in with the new password.
Exception: An error message is shown if the email is not registered or if the OTP is incorrect.

12. Admin Control (Monitoring and Managing Events

and Users):
Description: Admin can monitor and manage events and user accounts (volunteers and organizations).
Input: Admin credentials, user/event details for management actions.
Source: Admin.
Output: Admin actions (approve/reject events, activate/deactivate user accounts, view reports) are stored in the database.
Destination: Database.
Action: Admin performs management actions on events and user accounts.
Pre-condition: Admin must be logged in with administrative privileges.
Post-condition: Admin actions are applied, and system data is updated accordingly.
Exception: An error message is shown if the admin action fails or if the admin does not have the necessary privileges.

13. **Posting Pictures and Comments**:
Description: Users (volunteers and organizations) can post pictures related to volunteering events and comment on pictures, similar to Instagram or Facebook.
Input: Picture file, caption, event ID (for posting pictures); comment text, picture ID (for commenting).
Source: User (Volunteer and Organization).
Output: Picture and caption are stored in the database; comments are stored and displayed on the picture.
Destination: Database.
Action: Pictures are uploaded and displayed on the event's page; comments are added and displayed below the picture.
Pre-condition: User must be logged in.
Post-condition: Picture and comments are visible to all users.
Exception: An error message is shown if the upload or comment fails due to invalid input or system errors.