**CSR Management system**

**Project Report**

**By**

**Lokesh Gurjar – AF04963460**

**ABSTRACT**

The “SocialWave” project aims to address the challenges of transparency, accountability, and stakeholder engagement in Corporate Social Responsibility (CSR) efforts. This platform will provide companies with tools to streamline the tracking, reporting, and management of CSR activities, ensuring real-time insights into project progress, resource allocation, and impact metrics. By centralizing CSR data and facilitating better communication with stakeholders, “SocialWave” will help companies overcome common CSR challenges such as inconsistent reporting and lack of engagement. Ultimately, this project seeks to enhance the credibility and effectiveness of CSR initiatives, fostering a more responsible and sustainable business environment.

**Acknowledgment**

The toughest of endeavors in the world is not possible without the support of a helping hand which guides and motivates a person to take on any challenge head on. Inputs from such helping hands are always like very essential because more often or not certain mistakes which go unnoticed from our eyes.

I am heartily thankful to my guideguidance and support from the initial to the final level enabled me to develop an understanding of the subject.

I am thankful to who had provided all the required facilities to carry out the dissertation project work and nurturing my skills to execute the requirements.

Last but not the least I thank my dear parents, teachers and friends who have been source of support, strength, inspiration and encouragement for whatever I am today.

I thank the Almighty for all his blessings and encouraging me whenever I felt low and guiding me throughout

**TABLE OF CONTENT**

|  |  |
| --- | --- |
| **Topics** | **Page No.** |
| **Chapter 1: Introduction** | **9 - 12** |
| 1.1 Background | 10 |
| 1.2 Objectives | 10 |
| 1.3 Purpose, Scope & Applicability | 11 |
| 1.3.1 Purpose | 11 |
| 1.3.2 Scope | 11 |
| 1.3.3 Applicability | 11 |
| 1.4 Achievements | 12 |
| 1.5 Organization of Report | 12 |
| **Chapter 2: Survey of Technologies** | **13 - 20** |
| 2.1 HTML | 13 |
| 2.2 CSS | 14 |
| 2.3 Java | 15 |
| 2.4 JavaScript | 16 |
| 2.5 Node.JS | 17 |
| 2.6 Firebase | 18 |
| 2.7 .NET | 19 |
| 2.8 MongoDB | 20 |
| **Chapter 3: Requirements and Analysis** | **21 - 41** |
| 3.1 Problem Definition | 21 |
| 3.1.1 Problem Statement: | 21 |
| 3.1.2 Problem definition: | 21 |
| 3.1.3 Existing system: | 22 |
| 3.1.4 Proposed system: | 23 |
| 3.2 Requirements Specification | 23 |
| 3.2.1 Functional requirements | 23 |
| 3.2.2 Non-Functional requirements | 24 |

|  |  |
| --- | --- |
| 3.3 Planning and Scheduling | 26 |
| 3.3.1 GANTT Chart | 26 |
| 3.3.2 PERT Chart | 27 |
| 3.4 Software and Hardware Requirements | 28 |
| 3.4.1 Software requirements | 28 |
| 3.4.2 Hardware requirements | 28 |
| 3.5 Preliminary Product Description | 28 |
| 3.6 Conceptual Models | 28 |
| 3.6.1 ER Diagram | 30 |
| 3.6.2 Data Flow Diagram | 33 |
| 3.6.3 Class Diagram | 38 |
| 3.6.4 Flowchart | 40 |
| **Chapter 4: System Design** | **42 - 61** |
| 4.1 Basic Modules | 42 |
| 4.2 Data Design | 42 |
| 4.2.1 Schema Design | 43 |
| 4.2.2 Data Integrity and Constraints | 45 |
| 4.3 Procedural Design | 46 |
| 4.3.1 Logical Diagrams | 46 |
| 4.3.1.1 Use Case Diagram | 46 |
| 4.3.1.2 Activity Diagram | 49 |
| 4.3.2 Data Structures | 52 |
| 4.3.3 Algorithm design | 53 |
| 4.4 User Interface Design | 55 |
| 4.5 Security Issues | 58 |
| 4.6 Test Case Design | 60 |

**CHAPTER 1**

**INTRODUCTION**

The “SocialWave” project was developed as a solution to these challenges, offering a comprehensive platform to streamline CSR management. By centralizing tracking, reporting, and stakeholder engagement, “SocialWave” enables companies to monitor their CSR activities in real-time, providing valuable insights into their social and environmental impact. This platform not only improves transparency and accountability but also fosters stronger relationships with stakeholders, ultimately helping businesses enhance the credibility and effectiveness of their CSR initiatives and contribute to a more sustainable and responsible business environment.

Corporate Social Responsibility (CSR) has become a crucial aspect of modern business, where companies are expected to balance profitability with ethical, social, and environmental responsibilities. As consumers, governments, and organizations demand greater transparency and accountability, businesses are under increasing pressure to manage and report their CSR initiatives effectively. However, challenges such as inconsistent reporting, fragmented data, and limited stakeholder engagement have often hindered companies from realizing the full potential of their CSR efforts.

**1.1 BACKGROUND**

Corporate Social Responsibility (CSR) has evolved from a voluntary business practice to a critical aspect of corporate strategy, driven by increasing global awareness of environmental and social issues. Over the past few decades, stakeholders, including consumers, employees, investors, and governments, have begun to demand more responsible behavior from companies, pushing them to go beyond traditional profit-driven models. As businesses face pressure to demonstrate their commitment to social and environmental causes, CSR has become a key indicator of ethical governance and sustainable business practices.

The rise of CSR has been accompanied by a shift in how companies approach their responsibilities to society. While early CSR efforts were largely philanthropic, focused on charitable donations and community outreach, the modern approach integrates CSR into the core business model. Companies are expected to adopt sustainable practices, reduce their carbon footprint, and engage with local communities in meaningful ways. However, despite the widespread adoption of CSR programs, many organizations face significant challenges in implementing, tracking, and reporting their efforts. Common issues include inconsistent reporting standards, difficulty in measuring the actual impact of initiatives, and a lack of transparency in communicating with stakeholders.

This gap between CSR goals and effective execution has created a need for tools that can help companies better manage and track their initiatives. The “SocialWave” project was conceived to address these challenges by offering a platform that centralizes CSR activities, ensuring transparency, accountability, and enhanced engagement with stakeholders. By leveraging technology, “SocialWave” aims to provide companies with the ability to streamline their CSR operations, improve reporting consistency, and measure the true impact of their efforts, ultimately contributing to the advancement of sustainable and responsible business practices.

**1.2 OBJECTIVE**

The primary objective of the “SocialWave” project is to develop a user-friendly platform that enhances the management, transparency, and accountability of Corporate Social Responsibility (CSR) initiatives within organizations. The platform aims to centralize the tracking and reporting of CSR activities, providing real-time insights into project progress, resource allocation, and the social and environmental impact of each initiative. By doing so, “SocialWave” seeks to address common challenges such as inconsistent reporting, lack of stakeholder engagement, and difficulty in measuring CSR outcomes.

Additionally, “SocialWave” aims to foster stronger relationships between companies and their stakeholders by offering features that facilitate communication and feedback. This includes customizable dashboards for monitoring relevant CSR metrics and tools for stakeholders to engage with CSR projects, providing feedback and enhancing overall involvement. Ultimately, the project’s goal is to strengthen the effectiveness and credibility of CSR efforts, empowering companies to align their business practices with ethical and sustainable standards, while fostering a positive social and environmental impact.

**1.3 PURPOSE, SCOPE & APPLICABILITY**

**1.3.1 PURPOSE**

The purpose of the “SocialWave” project is to create an innovative platform that addresses the challenges companies face in managing their Corporate Social Responsibility (CSR) activities. By enhancing transparency, accountability, and stakeholder engagement, “SocialWave” aims to streamline CSR efforts and provide organizations with the tools they need to track, report, and optimize their social and environmental impact. The platform is designed to improve the effectiveness and credibility of CSR programs, helping businesses align their practices with ethical and sustainable standards, while meeting growing stakeholder expectations for responsible corporate behavior.

**1.3.2 SCOPE**

The scope of the “SocialWave” project encompasses the design, development, and implementation of a comprehensive CSR management platform. Key features of the platform include:

- “CSR Tracking and Reporting”: monitoring of CSR activities, including project progress, resource usage, and impact metrics.

- “Customizable Dashboards”: User-specific dashboards that allow businesses to focus on relevant CSR metrics and generate reports tailored to their needs.

- “Training and Support”: Ongoing user training and technical support to ensure effective utilization of the platform’s features.

- “Performance Monitoring”: Regular assessments of the platform's performance and its impact on CSR management to make necessary adjustments for continuous improvement.

**1.3.3 APPLICABILITY**

The “SocialWave” platform is applicable to businesses of all sizes and industries that are engaged in or looking to enhance their CSR efforts. It is particularly useful for organizations seeking to improve their CSR reporting, foster better communication with stakeholders, and ensure that their CSR initiatives are aligned with their broader sustainability goals. The platform can be used by company administrators, CSR managers, employees, volunteers, and external stakeholders to collaborate on, track, and optimize CSR projects. Whether an organization is running small local initiatives or large-scale global programs, “SocialWave” offers the tools needed to manage, measure, and maximize the social and environmental impact of CSR activities.

**1.4 ACHIEVEMENT**

**1.4.1 “Development of a Centralized Platform”**

Successfully creating a comprehensive, user-friendly platform that consolidates CSR activities into a single system. This has streamlined the tracking, reporting, and management of CSR initiatives, allowing organizations to monitor their projects in real-time.

**1.4.2 “Improved Transparency and Accountability”**

Through the tracking and reporting features, “SocialWave” has enabled companies to ensure that their CSR activities are transparent and aligned with their stated goals. This has fostered greater trust and credibility among stakeholders.

**1.4.3 “Enhanced Stakeholder Engagement”**

By integrating tools that facilitate communication and feedback, the platform has bridged the gap between organizations and their stakeholders, encouraging active participation from employees, volunteers, customers, and communities. This engagement has resulted in more impactful and meaningful CSR outcomes.

**1.4.4 “Customizable Reporting Dashboards”**

The achievement of designing customizable dashboards has allowed users to focus on relevant CSR metrics, tailoring reports to the specific needs of their organization and stakeholders, thus making data-driven decisions easier.

**1.4.5 “Positive Social and Environmental Impact”**

With the help of “SocialWave”, companies have been able to measure and optimize the social and environmental impact of their CSR initiatives, leading to more effective use of resources and greater contributions to sustainability.

**1.5 Organization of Report**

The “SocialWave” project report is organized as follows: It begins with an acknowledgment and declaration of the project's originality, followed by an abstract summarizing its purpose. The introduction provides an overview of CSR and the need for the platform, while the background outlines CSR challenges. The report details the project’s objectives, scope, and methodology, explaining how each module (Organization, Volunteer, Admin) functions. Key achievements, system limitations, and testing strategies are also discussed. The report concludes with a summary of the platform's impact and references.

**CHAPTER 2**

**Survey of Technologies**

**2.1 HTML**

“HTML (Hypertext Markup Language)” is the fundamental technology for building websites and web applications. It structures content and provides a framework for displaying text, images, links, and multimedia on the web. Below is an overview of the key applications and features of HTML:

**Applications of HTML:**

* **“Website Development”:** HTML is the backbone of every website, used to create the structure of webpages. It defines headings, paragraphs, links, lists, forms, images, and other content that makes up a web page. Every website you visit on the internet is fundamentally built using HTML.
* **“Web Application Interface”:** HTML is used in web applications to design and structure user interfaces. In applications like “SocialWave”, HTML is responsible for rendering elements such as registration forms, feedback sections, dashboards, and data displays.
* **“Email Templates”:** HTML is widely used for designing email templates, especially in marketing and communication campaigns. HTML-based emails can include images, buttons, and styled text to improve engagement.
* **“Documentation and eBooks”:** HTML is used in creating structured documents, such as user manuals, guides, and eBooks. Its semantic structure enables clear, hierarchical document organization.
* **“Mobile Applications (with frameworks)”:** HTML, when combined with CSS and JavaScript, can be used to create mobile applications through hybrid frameworks like Apache Cordova or Ionic. These frameworks enable developers to use web technologies to build mobile apps for iOS and Android.
* **“Gaming”:** HTML5, along with technologies like the `<canvas>` element and JavaScript, allows developers to build simple, browser-based games that do not require additional plugins or software.

**Key Features of HTML:**

* **“Semantic Structure”:** HTML provides tags that give meaning to web content. For example, `<header>`, `<footer>`, `<article>`, and `<section>` provide a clear structure, improving SEO and accessibility. HTML5 introduced many of these semantic elements to make web pages more meaningful and easier for search engines to crawl.
* **“Multimedia Support”:** HTML5 natively supports multimedia elements like audio (`<audio>`) and video (`<video>`), allowing developers to embed media content directly into web pages without the need for external plugins (e.g., Flash). This feature enhances the user experience on media-rich sites.
* **“Cross-Platform Compatibility”:** HTML works on all major browsers (Chrome, Firefox, Safari, Edge, etc.) and across devices (desktops, tablets, smartphones). Its universal compatibility makes it a go-to technology for building web applications that run on any platform.
* **“Forms and Input Elements”:** HTML offers a variety of form controls (`<input>`, `<textarea>`, `<select>`, `<button>`) to gather user data. HTML5 introduced new input types like `date`, `email`, `number`, and `range`, improving form usability and reducing the need for custom JavaScript validation.
* **“Graphics and Animation”:** HTML5 includes the `<canvas>` element, which allows developers to draw 2D graphics, build animations, or render game graphics directly in the browser using JavaScript. SVG (Scalable Vector Graphics) can also be embedded to create vector-based illustrations that scale perfectly on any screen size.
* **“Offline Browsing”:** HTML5 includes features that enable offline browsing. Using the Application Cache and Service Workers, web applications can store data locally, allowing users to access content without an internet connection, which is especially useful in web apps.
* **“Local Storage and Session Storage”:** HTML5 offers web storage APIs for saving data directly in the user's browser. Local storage and session storage enable web applications to store data on the client-side (such as user preferences or session data), providing a more personalized experience and faster load times without requiring a server.
* **“Responsive Design”:** HTML supports responsive web design through the use of `<meta>` tags and media queries (in conjunction with CSS). This allows web pages to adapt to various screen sizes and resolutions, ensuring that websites are accessible and user-friendly on all devices.
* **“Hyperlinks and Navigation”:** One of HTML's core features is its ability to create hyperlinks (`<a>` tag), enabling users to navigate between web pages easily. HTML's use of hyperlinks connects documents and resources across the internet, creating the vast, interconnected network we know as the web.
* **“Accessibility Features”:** HTML includes features like `alt` attributes for images, ARIA (Accessible Rich Internet Applications) attributes, and semantic tags to make web content more accessible to people with disabilities, improving usability for screen readers and other assistive technologies.

**Conclusion:**

HTML is a versatile and essential language for structuring and presenting content on the web. Its ability to build websites, support multimedia, create responsive designs, and integrate seamlessly with other technologies like CSS and JavaScript makes it indispensable for web development. The continuous advancements in HTML, particularly with HTML5, have significantly expanded its capabilities, allowing for the creation of complex and interactive web applications.

**2.2 CSS**

“CSS (Cascading Style Sheets)” is a stylesheet language used to control the presentation and layout of HTML elements on a webpage. Here’s a concise overview of its key applications and features:

**Applications of CSS:**

* **“Website Styling”:** CSS is primarily used to style websites, allowing developers to define colors, fonts, spacing, and layout for various HTML elements, ensuring a visually appealing user experience.
* **“Responsive Design”:** CSS enables responsive web design through media queries, allowing web pages to adapt to different screen sizes and orientations, providing a seamless experience on mobile and desktop devices.
* **“Animations and Transitions”:** CSS supports animations and transitions, allowing developers to create dynamic effects, such as hover effects and smooth content changes, enhancing user interaction.
* **“Theming”:** CSS can be used to implement themes, enabling easy changes to the overall look and feel of a website without altering the HTML structure, which simplifies maintenance and updates.
* **“Print Styles”:** CSS allows the creation of print-specific styles, enabling users to print web pages in a format that is clean and readable, enhancing usability for printed content.

**Key Features of CSS:**

* **“Selectors and Properties”:** CSS utilizes selectors to target HTML elements and properties to define their styles, providing fine-grained control over the appearance of content.
* **“Box Model”:** CSS uses the box model to define how elements are sized and spaced, including margins, borders, padding, and content area, crucial for layout design.
* **“Cascading and Inheritance”:** CSS follows a cascading order, allowing multiple styles to apply to the same element, with rules for specificity and inheritance determining the final appearance.
* **“Flexbox and Grid Layout”:** CSS provides advanced layout techniques, such as Flexbox and Grid, which enable responsive and complex layouts without the need for additional frameworks or JavaScript.
* **“Custom Properties (CSS Variables)”:** CSS allows the use of custom properties (variables) to store values for reuse, making styles more maintainable and adaptable.

**Conclusion**

CSS is an essential technology for web development, providing the tools to style and layout HTML content effectively. Its applications range from basic website design to complex responsive layouts, while its features empower developers to create visually engaging and user-friendly web experiences.

**2.3 Java**

“Java” is a versatile, high-level programming language widely used for various applications due to its platform independence, security features, and robust performance. Below are the key applications and features of Java:

**Applications of Java:**

* **“Web Development”:** Java is commonly used for server-side applications, employing frameworks like Spring and JavaServer Faces (JSF) to build dynamic websites and web applications.
* **“Mobile Applications”:** Java is the primary language for developing Android applications. Its rich libraries and frameworks, like Android SDK, enable developers to create feature-rich mobile apps.
* **“Enterprise Applications”:** Java is widely used in enterprise-level applications due to its scalability and reliability. Technologies like Java EE (Enterprise Edition) provide a robust platform for building large-scale applications.
* **“Desktop Applications”:** Java can be used to create cross-platform desktop applications using frameworks like JavaFX and Swing, which provide rich user interfaces.
* **“Embedded Systems and IoT”:** Java is utilized in embedded systems and Internet of Things (IoT) devices due to its portability and performance, allowing for the development of applications that run on various devices.
* **“Game Development”:** Java is used in game development for creating 2D and 3D games using libraries such as LibGDX, making it easier to develop cross-platform games.

**Key Features of Java:**

* **“Platform Independence”:** Java programs are compiled into bytecode, which can run on any device with a Java Virtual Machine (JVM), making Java highly portable.
* **“Object-Oriented”:** Java is based on object-oriented programming principles, which promote code reusability, modularity, and maintainability through concepts like inheritance, encapsulation, and polymorphism.
* **“Robust and Secure”:** Java emphasizes error handling and security features, such as automatic garbage collection and strong memory management, reducing the risk of crashes and memory leaks.
* **“Multithreading”:** Java supports multithreading, allowing concurrent execution of multiple threads. This is particularly useful for developing high-performance applications that require multitasking.
* **“Rich API”:** Java provides a comprehensive standard library (API) that includes built-in classes and methods for networking, data manipulation, and graphical user interface (GUI) development, facilitating rapid application development.
* **“Community Support”:** With a vast developer community and extensive documentation, Java benefits from robust support, making it easier for developers to find solutions and share knowledge.

**Conclusion**

Java’s versatility, platform independence, and extensive libraries make it a powerful language for developing a wide range of applications, from web and mobile apps to enterprise solutions and embedded systems. Its strong object-oriented principles and community support further enhance its effectiveness and popularity among developers.

**2.4 JavaScript**

“JavaScript” is a dynamic programming language primarily used for enhancing interactivity and functionality on web pages. Here are its key applications and features:

**Applications of JavaScript:**

* **“Web Development”:** JavaScript is essential for front-end development, enabling interactive elements like forms, sliders, and animations to enhance user experience on websites.
* **“Single-Page Applications (SPAs)”:** It powers SPAs, allowing content to load dynamically without refreshing the page, resulting in faster and smoother user interactions.
* **“Server-Side Development”:** With environments like Node.js, JavaScript can also be used for server-side development, enabling full-stack JavaScript applications.
* **“Mobile App Development”:** Frameworks like React Native and Ionic allow developers to build cross-platform mobile applications using JavaScript.
* **“Game Development”:** JavaScript is used in developing browser-based games, utilizing libraries like Phaser and Three.js for graphics and game mechanics.

**Features of JavaScript:**

* **“Dynamic and Weakly Typed”:** JavaScript allows variable types to change dynamically, providing flexibility in coding.
* **“Event-Driven”:** The language is built around events, making it easy to respond to user actions, such as clicks and key presses.
* **“Object-Oriented”:** JavaScript supports object-oriented programming (OOP) principles, allowing for the creation of reusable code through objects and prototypes.
* **“Cross-Platform Compatibility”:** JavaScript runs on all major browsers and platforms, ensuring consistent performance across different environments.
* **“Rich Ecosystem”:** A vast number of libraries and frameworks (e.g., React, Angular, Vue) enhance JavaScript capabilities, simplifying development processes and improving efficiency.

**Conclusion**

JavaScript is a versatile language that plays a crucial role in modern web development. Its applications range from enhancing user interfaces to building full-stack applications, while its dynamic features and rich ecosystem make it an essential tool for developers.

**2.5 Node.js**

“Node.js” is a powerful runtime environment built on Chrome's V8 JavaScript engine, enabling developers to execute JavaScript code server-side. Here are its key applications and features:

**Applications of Node.js:**

* **“Web Applications”:** Node.js is widely used for building scalable web applications, enabling real-time functionalities like chat applications, collaborative tools, and online gaming.
* **“API Development”:** It is ideal for developing RESTful APIs, allowing for efficient handling of multiple requests, making it popular for microservices architectures.
* **“Single-Page Applications (SPAs)”:** Node.js supports the development of SPAs that provide a smooth user experience by dynamically loading content without refreshing the entire page.
* **“Internet of Things (IoT)”:** Node.js is suitable for IoT applications due to its event-driven architecture, allowing it to handle numerous simultaneous connections with minimal resources.
* **“Data Streaming”:** It enables the creation of data streaming applications, facilitating real-time processing of audio and video files.

**Features of Node.js:**

* **“Asynchronous and Event-Driven”:** Node.js operates on a non-blocking I/O model, allowing multiple operations to run concurrently without waiting for each to finish, leading to high performance.
* **“Single-Threaded”:** Despite being single-threaded, Node.js can handle many connections simultaneously, making it efficient for I/O-intensive tasks.
* **“NPM (Node Package Manager)”:** NPM provides access to a vast ecosystem of libraries and modules, simplifying the integration of functionalities into applications.
* **“Cross-Platform”:** Node.js applications can run on various operating systems, including Windows, macOS, and Linux, ensuring broad compatibility.
* **“Scalability”:** Node.js is designed for scalability, supporting microservices architecture and allowing applications to handle increased load by easily adding more servers.

**Conclusion**

Node.js is a versatile platform that excels in building fast, scalable applications across various domains, from web applications to IoT. Its asynchronous nature and extensive package ecosystem make it a popular choice among developers for modern application development.

**2.6 Firebase**

“Firebase” is a comprehensive app development platform provided by Google, designed to help developers build high-quality applications quickly. Here are its key applications and features:

**Applications of Firebase:**

* **“Mobile and Web Applications”:** Firebase is widely used for developing mobile (iOS and Android) and web applications, providing backend services without the need to manage servers.
* **“Real-Time Databases”:** It allows developers to build applications that require real-time data synchronization, such as chat applications, collaborative tools, and live dashboards.
* **“Authentication”:** Firebase simplifies user authentication processes with various methods (email/password, social logins, phone authentication), making it easy to secure applications.
* **“Analytics and Performance Monitoring”:** Firebase integrates analytics tools that help developers understand user behavior and app performance, enabling data-driven improvements.
* **“Cloud Functions and Hosting”:** It offers serverless computing capabilities through Cloud Functions, allowing developers to run backend code in response to events, along with hosting static and dynamic content.

Features of Firebase:

* **“Real-Time Database”:** Firebase's NoSQL database allows for real-time data updates and offline capabilities, ensuring seamless user experiences.
* **“Cloud Firestore”:** A flexible, scalable database for storing and syncing app data in real-time with enhanced querying capabilities.
* **“Firebase Authentication”:** A robust authentication system that supports various sign-in methods and integrates easily with other Firebase services.
* **“Cloud Messaging”:** Firebase Cloud Messaging (FCM) enables developers to send notifications and messages to users across platforms.
* **“Analytics and Crash Reporting”:** Built-in tools for monitoring app performance and user engagement, helping developers identify issues and improve app quality.

**Conclusion**

Firebase provides a powerful suite of tools and services that streamline the development of mobile and web applications. Its real-time database capabilities, authentication features, and integration with analytics make it an ideal choice for developers looking to build scalable, high-performance applications quickly.

**2.7 .NET**

“.NET” is a versatile development framework created by Microsoft, designed for building a wide range of applications. Here are its key applications and features:

**Applications of .NET:**

* **“Web Applications”:** .NET is widely used for developing dynamic web applications and services using ASP.NET, allowing developers to build robust, scalable websites.
* **“Desktop Applications”:** It supports the creation of rich desktop applications for Windows using Windows Forms and WPF (Windows Presentation Foundation).
* **“Mobile Applications”:** With Xamarin, .NET enables the development of cross-platform mobile applications for iOS and Android, allowing code sharing across platforms.
* **“Cloud-Based Applications”:** .NET is used to build cloud applications that leverage Microsoft Azure services, making it easy to develop, deploy, and scale applications in the cloud.
* **“Game Development”:** The Unity game engine uses .NET for scripting, allowing developers to create cross-platform games.

**Features of .NET:**

* **“Multi-Language Support”:** .NET supports multiple programming languages, including C#, VB.NET, and F#, enabling developers to choose their preferred language.
* **“Robust Libraries”:** It comes with a rich set of libraries and frameworks (like ASP.NET, Entity Framework) that simplify common programming tasks.
* **“Cross-Platform Development”:** With .NET Core (now part of .NET 5 and later), developers can build applications that run on Windows, macOS, and Linux.
* **“Memory Management”:** .NET features automatic memory management through garbage collection, which helps in optimizing resource usage.
* **“Security”:** The framework provides built-in security features, including authentication, authorization, and encryption, helping developers create secure applications.

**Conclusion**

.NET is a powerful framework suitable for developing a wide range of applications, from web and desktop to mobile and cloud-based solutions. Its multi-language support, robust libraries, and cross-platform capabilities make it a popular choice among developers for building scalable and secure applications.

**2.8 MongoDB**

“MongoDB” is a popular NoSQL database known for its flexibility and scalability, designed to handle large volumes of unstructured data. Here are its key applications and features:

**Applications of MongoDB:**

* **“Web Applications”:** Ideal for web applications requiring rapid data access and flexible schema design, MongoDB is commonly used for social networks, content management systems, and e-commerce platforms.
* **“Big Data Analytics”:** MongoDB is well-suited for handling large datasets in real-time analytics, enabling businesses to extract valuable insights from their data quickly.
* **“Internet of Things (IoT)”:** It supports IoT applications by managing vast amounts of data generated from connected devices, allowing for efficient data storage and retrieval.
* **“Mobile Applications”:** MongoDB is used in mobile app development to handle diverse data types and provide fast data access, essential for delivering responsive user experiences.
* **“Content Management”:** It excels in content management systems where data structures can evolve over time, allowing for dynamic content storage.

**Features of MongoDB:**

* **“Document-Oriented Storage”:** MongoDB stores data in flexible, JSON-like documents, allowing for varying fields and structures, which simplifies data modeling.
* **“Scalability”:** It supports horizontal scaling through sharding, enabling databases to expand easily across multiple servers as data volume increases.
* **“Indexing”:** MongoDB offers powerful indexing options to improve query performance, including support for compound, geospatial, and full-text indexes.
* **“High Availability”:** The database provides robust replication features through replica sets, ensuring data availability and redundancy.
* **“Aggregation Framework”:** MongoDB's aggregation framework enables complex data processing and transformation, making it easier to perform analytical operations directly within the database.

**Conclusion**

MongoDB is a versatile NoSQL database that offers flexibility, scalability, and ease of use, making it suitable for a wide range of applications from web development to big data analytics. Its document-oriented approach and rich feature set enable developers to efficiently manage and utilize data in modern applications.

**CHAPTER 3**

**REQUIREMENTS AND ANALYSIS**

**3.1. Problem Definition**

**3.1.1 Problem Statement**

In today's business environment, companies are increasingly expected to demonstrate their commitment to Corporate Social Responsibility (CSR), balancing profit with social and environmental impact. However, many organizations face significant challenges in managing and reporting their CSR initiatives effectively. Common issues include:

* **“Lack of Transparency”:** CSR activities are often poorly tracked, leading to inconsistent reporting and a lack of clarity around the actual impact of these initiatives. This undermines stakeholder trust and limits the ability to showcase meaningful outcomes.
* **“Inefficient Stakeholder Engagement”:** Organizations struggle to engage key stakeholders, such as employees, customers, and communities, in their CSR efforts. Without effective communication channels and feedback mechanisms, stakeholder involvement remains limited, weakening the overall impact of CSR projects.
* **“Difficulty in Measuring Impact”:** Many companies find it challenging to accurately measure and report the social and environmental benefits of their CSR activities. This results in missed opportunities to improve initiatives or optimize resource allocation.
* **“Fragmented CSR Management”:** CSR initiatives are often managed using disparate tools and processes, making it difficult for organizations to streamline operations, monitor progress, and integrate feedback in real-time.

Given these challenges, there is a clear need for a comprehensive solution that can centralize CSR management, enhance transparency, improve stakeholder engagement, and provide actionable insights into the impact of CSR initiatives. The “SocialWave” project seeks to address these problems by providing an integrated platform for efficient CSR tracking, reporting, and engagement.

**3.1.2. Problem Definition**

The increasing emphasis on Corporate Social Responsibility (CSR) in modern business highlights the need for companies to actively engage in ethical, social, and environmental initiatives. However, many organizations face critical challenges in managing, measuring, and reporting their CSR efforts effectively. The key problems include:

* **“Fragmented CSR Management”:** CSR initiatives are often handled using multiple tools or manual processes, leading to inefficient tracking and disorganized reporting. This fragmentation limits organizations' ability to have a clear overview of their projects, resource usage, and outcomes.
* **“Lack of Transparency and Accountability”:** Companies struggle to provide transparent, consistent, and accountable reporting on CSR activities. Inadequate data tracking leads to discrepancies in reporting, which can damage stakeholder trust and corporate reputation.
* **“Poor Stakeholder Engagement”:** Organizations often fail to effectively engage employees, customers, and communities in CSR activities due to a lack of interactive tools. The absence of real-time communication and feedback mechanisms results in limited involvement and reduced social impact.
* **“Inability to Measure CSR Impact”:** Many companies find it difficult to accurately quantify and report the social and environmental benefits of their CSR initiatives. This lack of measurable insights hinders continuous improvement and strategic alignment with long-term sustainability goals.

The “SocialWave” platform aims to solve these issues by providing a centralized solution that enhances transparency, streamlines CSR management, and fosters stronger stakeholder engagement, while also offering tools to measure and optimize CSR impact effectively.

**3.1.3. Existing System**

The current landscape of Corporate Social Responsibility (CSR) management in many organizations is fragmented and inefficient. Most companies rely on a combination of manual processes, spreadsheets, or disconnected software tools to track and report their CSR initiatives. This results in several key challenges:

* **“Manual Data Collection and Reporting”:** In many cases, CSR activities are tracked manually using spreadsheets or standalone applications. This process is time-consuming and prone to errors, leading to inconsistent and unreliable data. Manual reporting makes it difficult to maintain transparency and accountability, especially when organizations need to present detailed progress and impact reports to stakeholders.
* **“Lack of Centralized Platform”:** Existing systems typically do not provide a single, integrated platform for managing all aspects of CSR. Organizations use different tools for project management, stakeholder communication, volunteer coordination, and impact measurement. This results in a lack of cohesion and makes it difficult to track all CSR activities and outcomes in one place.
* **“Limited Stakeholder Engagement”:** Current systems offer limited functionality for engaging stakeholders like employees, customers, and the broader community. Many organizations struggle to actively involve stakeholders in CSR initiatives due to the absence of real-time communication and feedback tools.
* **“Poor Real-Time Tracking”:** Existing systems often fail to offer real-time insights into CSR activities. As a result, organizations lack the ability to monitor progress dynamically, adjust resource allocation, or make data-driven decisions to improve project impact.
* **“Ineffective Impact Measurement”:** Measuring the social and environmental impact of CSR initiatives is a major challenge in existing systems. Most tools focus only on activity tracking, without providing robust features for evaluating the outcomes and real benefits of CSR programs.

**3.1.4. Proposed System**

The “SocialWave” platform is designed to address the limitations of existing CSR management systems by offering a comprehensive, centralized solution that streamlines the tracking, reporting, and management of Corporate Social Responsibility (CSR) initiatives. The key features and benefits of the proposed system are as follows:

* **“Centralized CSR Management”:** “SocialWave” offers a unified platform where organizations can manage all CSR activities in one place. From project creation and volunteer coordination to resource allocation and reporting, the platform consolidates these processes, ensuring streamlined operations.
* **“Enhanced Transparency and Accountability”:** The platform provides real-time insights and automated reporting, allowing companies to produce transparent, consistent, and accountable reports for internal and external stakeholders. This improves trust and credibility with stakeholders like customers, investors, and employees.
* **“Real-Time Stakeholder Engagement”:** “SocialWave” includes tools to actively involve stakeholders (volunteers, employees, customers, and communities) in CSR activities. feedback systems, and engagement features, stakeholders can participate more effectively.
* **“Accurate Impact Measurement”:** The platform includes features for measuring the social and environmental impact of CSR initiatives. Organizations can assess key metrics such as resource usage, outcomes, and community impact, providing a comprehensive evaluation of the benefits achieved.
* **“Scalability and Flexibility”:** “SocialWave” is built to scale as CSR efforts grow. The platform supports organizations of all sizes, from small businesses to large enterprises, and can accommodate an increasing number of projects, volunteers, and stakeholders without compromising performance.

**3.2. Requirements Specification**

**3.2.1 Functional Requirements**

The “SocialWave” platform must fulfill several functional requirements to meet the needs of users and deliver an efficient CSR management solution. These functional requirements are grouped into categories based on user roles and core functionalities.

**1. “User Management”**

* + **“User Registration and Login”:** Organizations, volunteers, and admins must be able to create accounts and securely log in to the platform using credentials such as email and password or via third-party authentication (e.g., Google, social media logins).
  + The system should allow user profile management, including updating personal information, organization details, and password changes.
  + **“Role-Based Access Control”:** The system must assign different access levels based on the user's role (Organization, Volunteer, Admin). Organizations manage CSR activities, volunteers participate in events, and admins oversee platform operations.

**2. “CSR Campaign and Event Management”**

* + **“Create and Manage CSR Projects”:** Organizations should be able to create CSR projects, including donation campaigns, volunteer events, and community outreach programs.
  + The platform must support setting goals, timelines, and resources required for each project.
  + **“Event Scheduling”:** Organizations should be able to create and schedule events, allowing volunteers to register for participation. The system must send event reminders and updates.

**3. “Volunteer Management”**

* + **“Volunteer Registration”:** Volunteers should be able to browse and register for CSR events hosted by organizations. The platform must list available events based on location, date, or type of activity.
  + **“Volunteer Feedback and Rating”:**After participating in events, volunteers should be able to provide feedback and rate the campaigns or projects, offering valuable insights to the organization.
  + **“Volunteer Certificates”:** Upon successful participation in CSR events, the platform should generate digital certificates for volunteers, recognizing their contributions.

**4. “Reporting and Data Analytics”**

* + **“Automated Reporting”:** The platform must generate automated reports on CSR activities, project outcomes, resource usage, and stakeholder engagement. These reports should be exportable in various formats (PDF, CSV).
  + **“Customizable Dashboards”:** The system should provide customizable dashboards for organizations to monitor specific CSR metrics such as project progress, volunteer participation, and financial contributions.

**5. “Security and Data Protection”**

* + **“Data Encryption”:** All sensitive user data (e.g., passwords, personal details) must be encrypted to protect against unauthorized access.
  + **“Role-Based Access Control”:** Only authorized users should be able to access certain functionalities (e.g., only admins can manage user accounts, only organizations can create projects).

**6. “Admin Panel”**

* + **“User Management”:** Admins must have the ability to manage user accounts, including creating, deleting, and modifying accounts for both organizations and volunteers.
  + **“Platform Monitoring”:** Admins should have access to reports on platform usage, project outcomes, and volunteer participation to monitor system performance.
  + **“System Performance Tracking”:** The platform must include tools for monitoring its overall performance, ensuring smooth operation, and addressing any issues that arise.

**3.2.2. Non-Functional Requirements**

The non-functional requirements define the system's performance, reliability, security, and other operational aspects that ensure the “SocialWave” platform runs efficiently and meets user expectations.

1**. “Performance Requirements”**

* + **“Response Time”:** The platform should have a response time of less than 2 seconds for standard operations, such as logging in, viewing project dashboards, and submitting feedback.
  + **“Scalability”:** The system must be scalable to handle a growing number of users, projects, and events without significant degradation in performance. It should support hundreds of concurrent users and efficiently manage large data volumes.
  + **“Data Processing”:** The platform must efficiently process real-time data, such as event registrations and project updates, ensuring that users receive timely information and notifications.

**2. “Availability and Reliability”**

* + **“Uptime”:** The platform should maintain an uptime of at least 99.9%, ensuring continuous availability for users. Any downtime for maintenance or updates must be scheduled outside of peak hours to minimize disruption.
  + **“Fault Tolerance”:** In case of system failures or crashes, the platform must automatically recover and ensure no data is lost. Redundant systems should be in place to handle failures seamlessly.

**3. “Security Requirements”**

* + **“Data Encryption”:** All sensitive data, including personal information, passwords, and financial details, must be encrypted both in transit (using HTTPS/SSL) and at rest.
  + **“User Authentication and Authorization”:** The system must support secure authentication mechanisms, including multi-factor authentication (MFA) for enhanced security. Role-based access control (RBAC) should be implemented to ensure that only authorized users can access specific features.
  + **“Data Privacy”:** The platform must comply with data protection regulations, such as GDPR, ensuring users’ personal data is handled securely and only used for its intended purpose.
  + **“Audit Logs”:** The system should maintain detailed audit logs of key actions, such as project updates, volunteer registrations, and data modifications, to ensure transparency and accountability.

**4. “Usability and User Experience”**

* + **“Intuitive User Interface”:** The platform must provide a user-friendly and intuitive interface that is easy to navigate for users with varying technical expertise (organizations, volunteers, admins).
  + **“Accessibility”:** The system should adhere to accessibility standards (WCAG 2.1), ensuring that it is usable by people with disabilities, including screen reader compatibility and keyboard navigation.
  + **“Mobile Responsiveness”:** The platform must be fully responsive, providing a seamless experience on both desktop and mobile devices, ensuring accessibility for users on various screen sizes.

**5. “Scalability and Flexibility”**

* + **“Horizontal Scalability”:** The platform should be able to scale horizontally, allowing for the addition of more servers to handle increased traffic without affecting performance.
  + **“Modular Architecture”:** The platform should be designed with a modular architecture, allowing easy updates, feature additions, or changes without

**6. “Maintainability”**

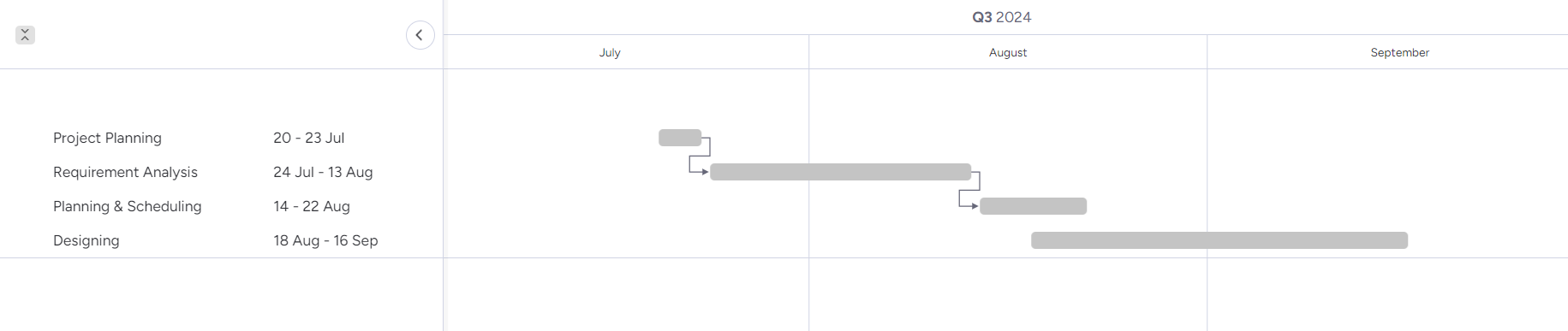
* + **“Modular Codebase”:** The code should be modular and well-documented, allowing for easy maintenance and updates. Regular code reviews and quality assurance checks should be implemented to maintain system integrity.
  + **“Automated Testing”:** The platform should incorporate automated testing (unit, integration, and performance tests) to ensure that new updates or features do not introduce bugs or degrade system performance.

**3.3 Planning and Scheduling**

The planning and scheduling of the **SocialWave** project involve a structured approach to ensure timely development and deployment. The project is divided into key phases: Requirement Gathering, System Design, Development, Testing, and Deployment. Each phase has specific milestones and deliverables. Initial planning includes gathering detailed user requirements, followed by designing the system architecture. The development phase is broken down into sprints, focusing on the creation of modules (Organization, Volunteer, Admin). Testing will be conducted concurrently, covering unit, integration, and user acceptance tests.

**3.3.1 GANTT CHART**

The **Gantt chart** provides a visual timeline for the development of the **Socialwave**, an web-based application. It divides the project into key phases and tasks, each with specific start and end dates. This ensures that the project remains on schedule and provides clarity on task dependencies and team responsibilities

****

**Fig 3.1** Gantt Chart

**3.3.2 PERT CHART**

The **Program Evaluation Review Technique** (PERT) chart is a project management tool used to plan, schedule, and coordinate tasks within the **Socialwave** ( development project. It helps visualize the project’s timeline, dependencies, and critical path, ensuring efficient resource allocation and timely

**Fig 3.2** Pert Chart

**3.4 Software and Hardware Requirements:**

### 3.4.1 Software Requirements:

* Operating system: Windows XP or above
* HTML [Hyper Text Markup Language]
* CSS [Cascading Style Sheets]
* JS [JavaScript]
* MongoDB
* NodeJs
* Visual Studio Code
* .NET

### 3.4.2 Hardware Requirements:

* Processor: Intel dual-core or above
* Processor Speed:1.0GHZ or above
* RAM: 4GB RAM or above
* Hard Disk: 20 GB hard disk or above

**3.5 Preliminary Product Description**

“SocialWave” is a comprehensive CSR management platform designed to streamline and enhance the tracking, reporting, and execution of Corporate Social Responsibility initiatives. It allows organizations to create, manage, and monitor CSR projects such as donation campaigns, volunteer events, and community outreach programs. The platform provides real-time data on project progress, stakeholder engagement, and resource usage through customizable dashboards. Additionally, it offers features for volunteer registration, feedback collection, and the generation of impact reports, making CSR efforts more transparent, accountable, and effective. SocialWave fosters stronger stakeholder participation while ensuring that CSR initiatives are measurable and aligned with organizational goals.

**3.6 Conceptual Models**

The “SocialWave” platform is based on a multi-layered conceptual model that includes three core components: “User Interaction Layer”, “Business Logic Layer”, and “Data Layer”.

**1. “User Interaction Layer”:**

* + This layer represents how users (organizations, volunteers, and admins) interact with the platform through web-based user interfaces. It includes:
  + “Web Interface”: Provides a user-friendly interface for organizations to create and manage CSR projects, for volunteers to register and participate, and for admins to oversee system operations.
  + “Mobile Access”: A responsive design ensures access via mobile devices, offering a seamless experience across platforms.

**2. “Business Logic Layer”:**

* This layer contains the core functionality of the platform, managing the business rules and processes for CSR activities.
* “CSR Project Management”: Handles the creation, progress tracking, and reporting of CSR campaigns and events.
* “Stakeholder Engagement”: Manages communication, notifications, and feedback between organizations, volunteers, and other stakeholders.
* “Reporting & Analytics”: Provides real-time insights into project outcomes, resource usage, and impact measurement.

**3. “Data Layer”:**

* This layer manages all data storage and retrieval, ensuring the integrity and accessibility of user, project, and engagement data.
* “Database (MongoDB)”: Stores structured and unstructured data, including user profiles, project details, and event registrations.
* “Data Security”: Ensures that data is encrypted and protected against unauthorized access.

**Implementation Methodology**

The implementation of “SocialWave” follows the “Agile Development Methodology” to ensure flexibility and responsiveness to changes. The key phases are:

**1. “Requirement Gathering”:**

* Initial phase where detailed user and system requirements are gathered from organizations and stakeholders to define the platform’s scope.

**2. “System Design”:**

* The system architecture is designed, focusing on the user interaction layer, business logic layer, and data storage.
* Key decisions include using “Node.js” for the backend, “MongoDB” for the database, and a “REST API” for connecting the front-end with the server.

**3. “Development (Sprints)”:**

* The development is divided into short, iterative sprints focusing on building different modules:
  + “Sprint 1”: User registration, authentication, and role management.
  + “Sprint 2”: CSR project and event management functionalities.
  + “Sprint 3”: Real-time reporting, stakeholder engagement, and notifications.

**4. “Testing”:**

* Unit, integration, and user acceptance testing (UAT) are conducted to identify and resolve issues. Performance and security testing ensure system robustness.

**5. “Deployment”:**

* The platform is deployed in a cloud environment, ensuring scalability and availability.

**6. “Post Support”:**

* Continuous monitoring and feedback-driven updates are implemented to refine the platform, incorporating new features or enhancements based on user input.

This iterative approach ensures that “SocialWave” is built efficiently, with flexibility for adjustments based on user needs or changing requirements.

### 3.1.1ER Diagram:

An entity-relationship diagram (ERD) is a graphical representation of an information system that depicts the relationships between people, objects, places, concepts, or events. An ERD is a data modeling technique that can be used to help define business processes and as the basis for a relational database. If the customer is unhappy with the first prototype, he or she may lose interest in the product. The three main cardinal relationships are:

* **One-to-one (1:1)**: For example, suppose each customer in a database has only one mailing address.
* **One-to-many (1: M)**: A single customer, for example, may place an order for multiple products. The customer is associated with several entities, but all of them are linked back to the same customer.
* **Many-to-many (M: N)**: In a company where all call center agents work with multiple customers, for example, each agent is associated with multiple customers, and multiple customers may also be associated with multiple agents.

Steps involved in creating an ERD include:

* + Identifying and classifying entities, identifying all interactions between entities
  + determining the cardinality of relationships by analyzing the nature of interactions

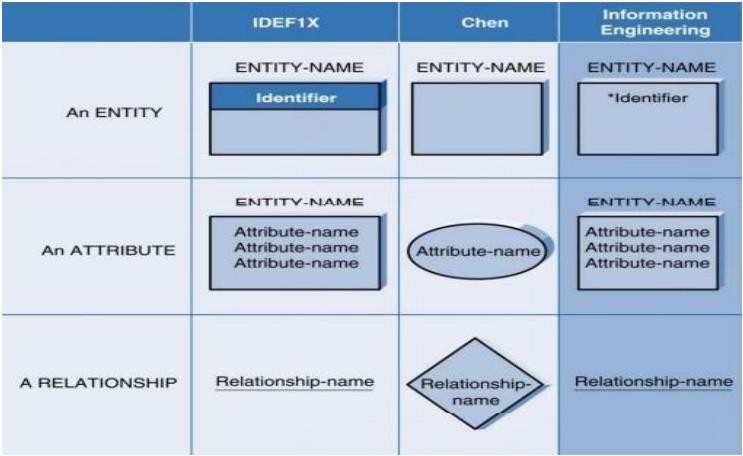
* + Making the ERD

Table 3.6.1.1 ER elements table

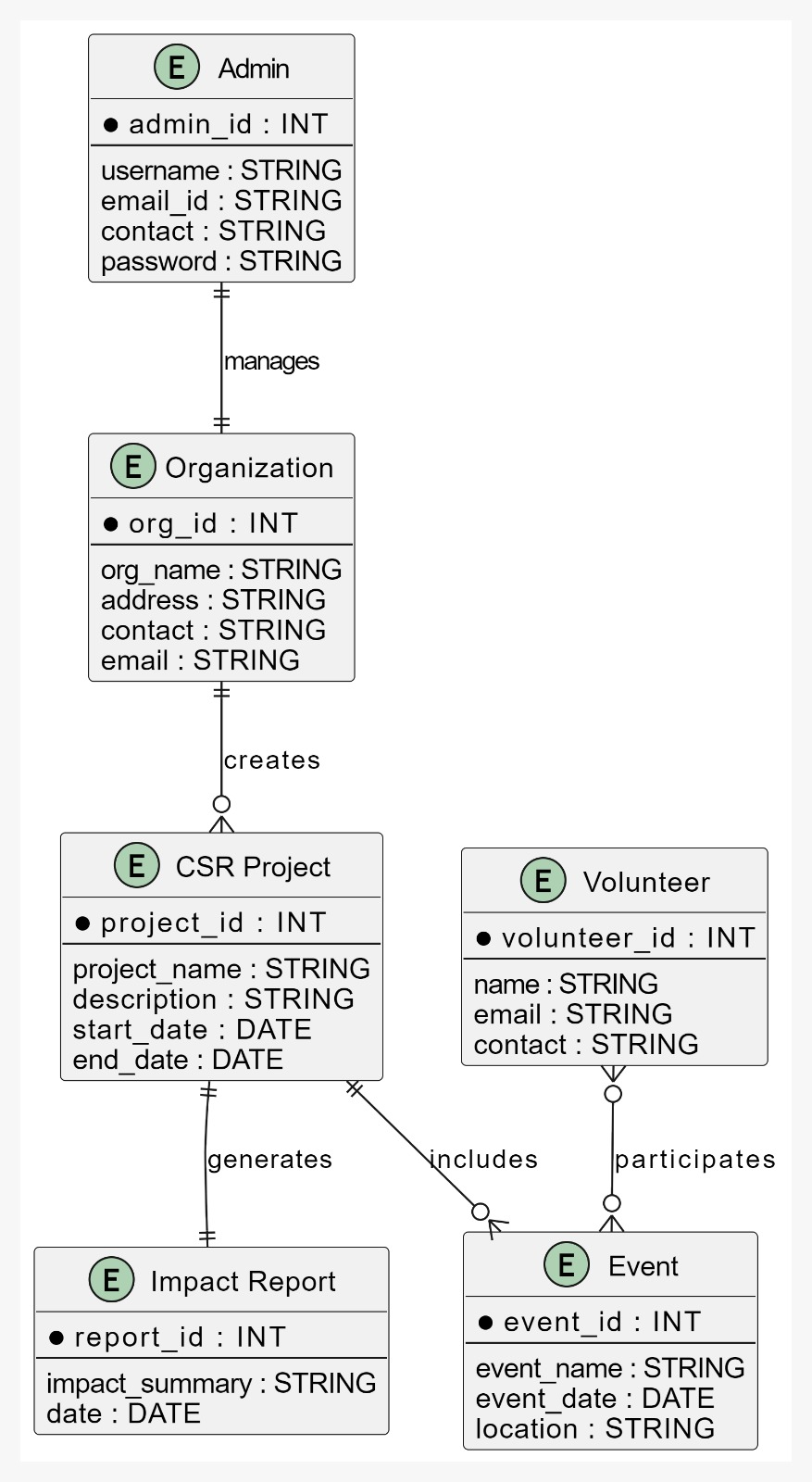


Fig 3.6.1.1 ER Diagram

### Data Flow Diagram (DFD):

A data flow diagram depicts the flow of information through a process or system. It includes data inputs and outputs, data stores, and the various subprocesses through which the data moves. DFDs are created by describing various entities and their relationships using standardized symbols and notation.

Data flow diagrams graphically represent systems and processes that would be difficult to describe in text. These diagrams can be used to map out an existing system and improve it, or to plan out a new system for implementation. Visualizing each element makes it simple to identify inefficiencies and create the best system possible.

DFD symbols differ slightly depending on the methodology (Gane and Sarson vs. Yourdon and Coad). However, the fundamental concepts remain the same. A data flow diagram consists of four basic elements: processes, data stores, external entities, and data flows. The illustration below depicts the standard shapes for both methodologies.

Elements table used in DFD:

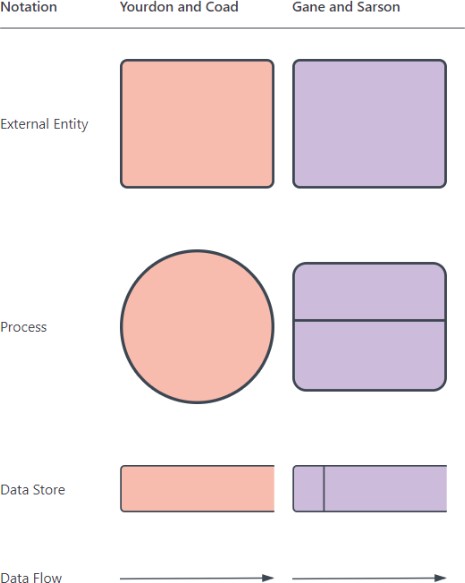


Table 3.6.2.1 DFD elements table

Level 0 DFDs, also known as context diagrams, are the most basic data flow diagrams. They provide a broad view that is easily digestible but offers little detail. Level 0 data flow diagrams show a single process node and its connections to external entities.

#### Level 0:

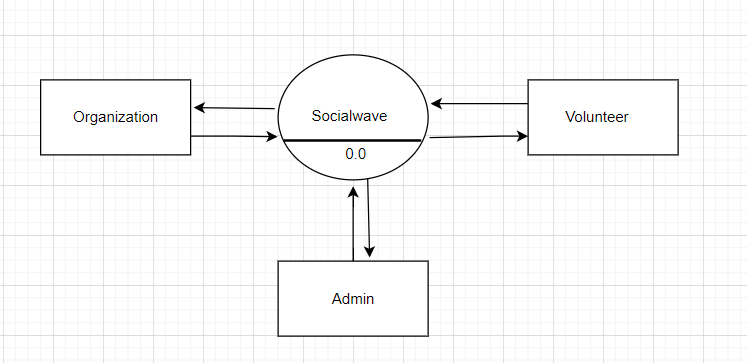
****

Fig 3.6.2.1 DFD Level 0

Level 1 DFDs are still a general overview, but they go into more detail than a context diagram. In a level 1 data flow diagram, the single process node from the context diagram is broken down into sub-processes. As these processes are added, the diagram will need additional data flows and data stores to link them together.

Level 1: Admin

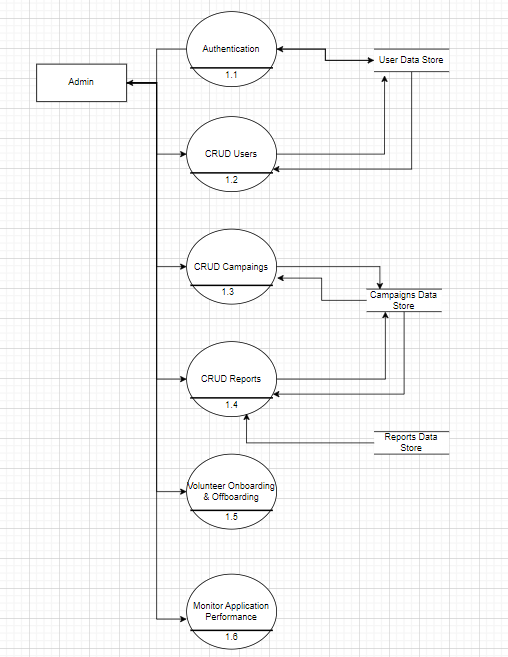


Fig 3.6.2.2 DFD Level 1(Admin)

Level 1 Organization

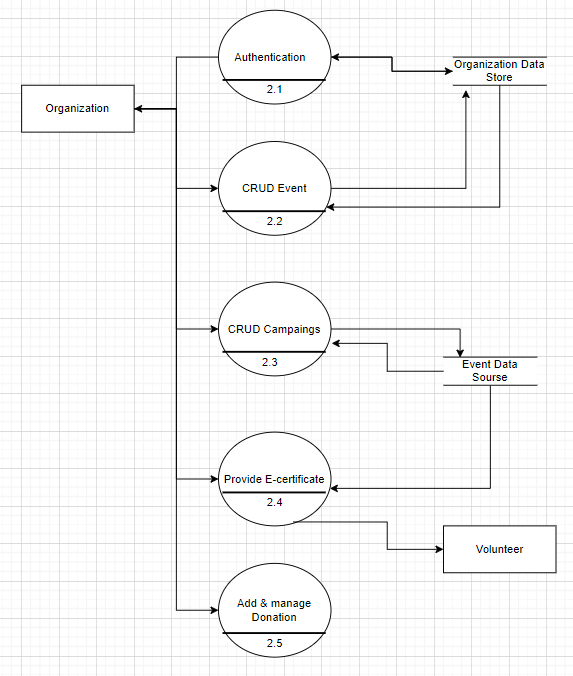
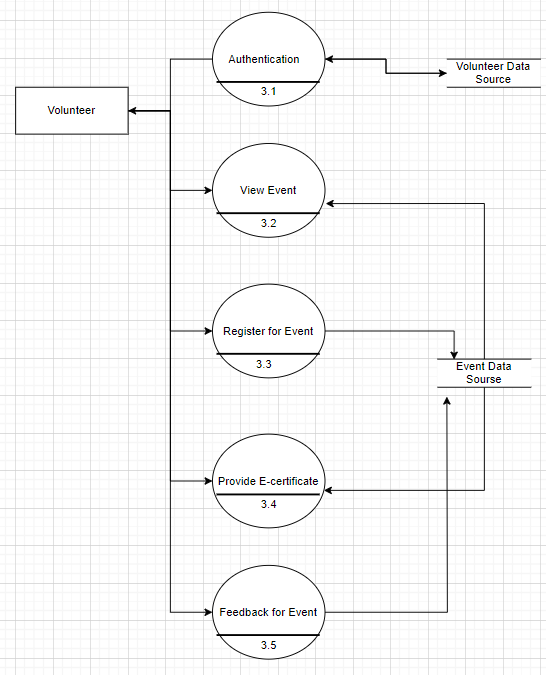


Fig 3.6.2.3 DFD Level 1(Organization)

Level 1: Volunteer



#### 

Fig 3.6.2.4 DFD Level 1(Volunteer)

### 3.1.1Class Diagram:

* In object-oriented modeling, the class diagram is the primary building block.
* They are used for both general conceptual modeling of the application’s system and detailed modeling that translates the models into programming code.
* A class diagram’s classes represent both the application’s main objects and interactions.
* Class diagrams are used to identify and categorize the objects that comprise a system.
* It also includes the critical characteristics of the objects that must be captured.

Class Diagram Notations:

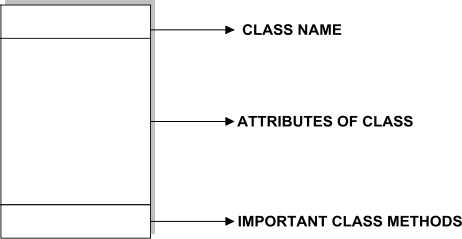
Rectangles represent Classes in Class diagrams, and the lines connecting the rectangles show the relationships between Classes.

Fig 3.6.3.1 Class diagram elements

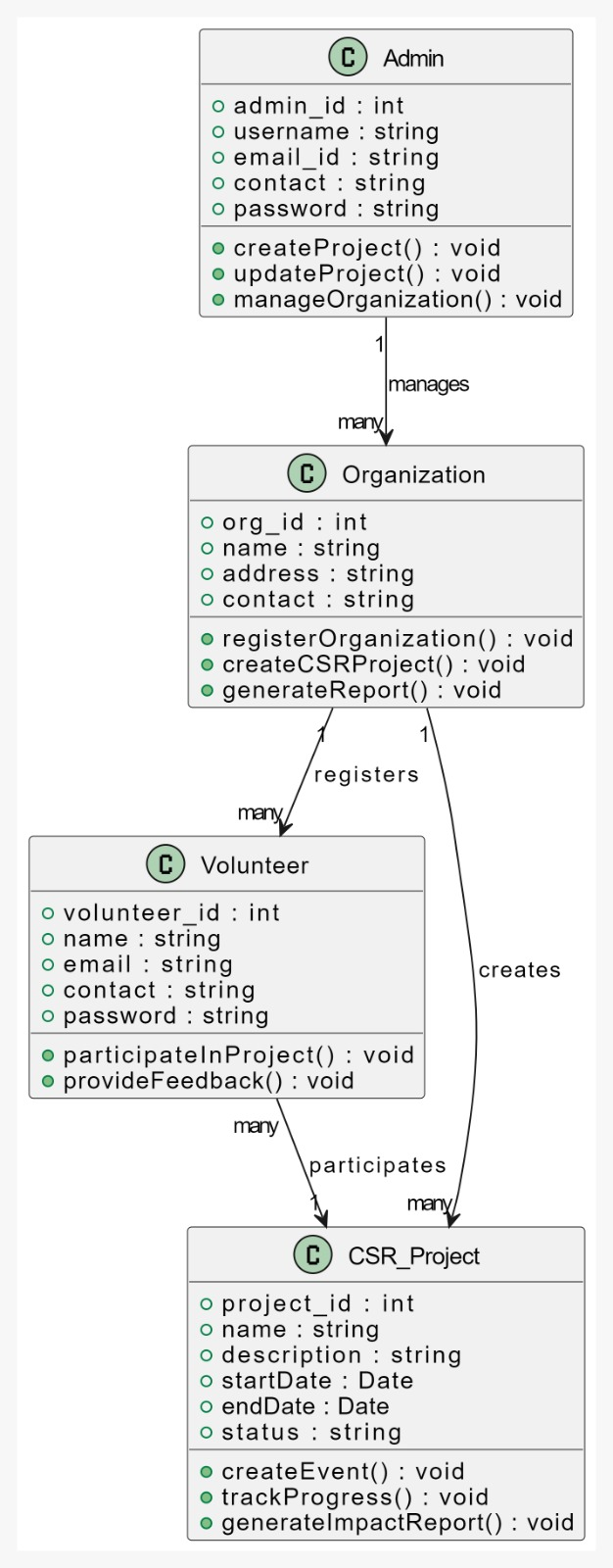


Fig 3.6.3.2 Class Diagram

### 3.1.2Flowchart:

A flowchart, a type of diagram, represents an algorithm, workflow, or process. The stages are depicted in the flowchart as a series of boxes of varying sizes, with arrows connecting the boxes in the correct order. This illustrative graphic depicts a paradigm for resolving a problem.

Using flowcharts to analyze, design, document, or manage multiple processes or programs. Symbols for Diagrams Flowcharts use special shapes to depict various operations or processes in a process. Lines and arrows depict the order of the stages and their interactions.

Symbols for flowcharts: -

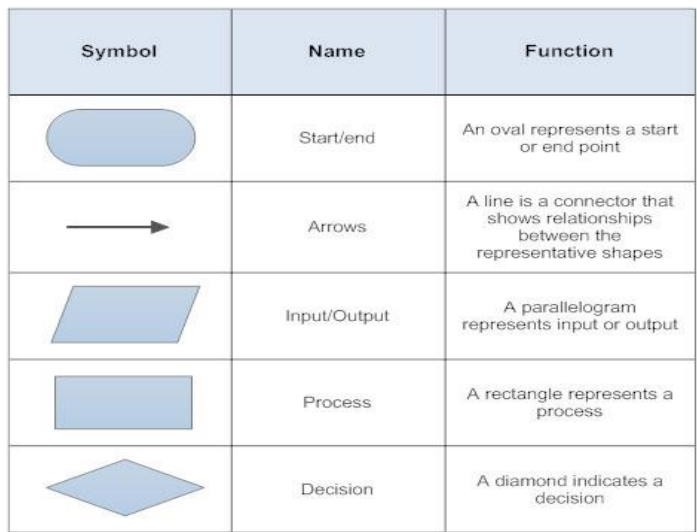


Table 3.6.4.1 Flowchart elements table

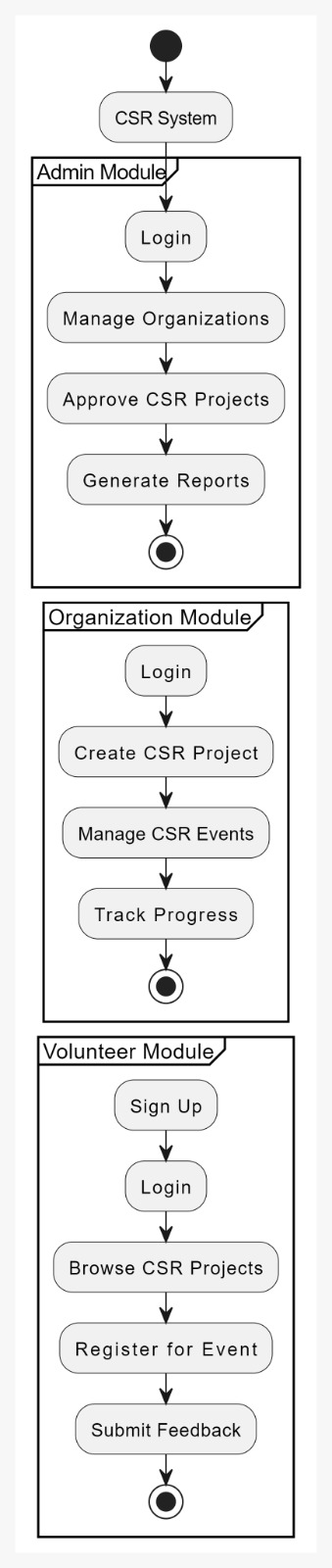


Fig 3.6.4.1 Flowchart diagram

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 Basic Modules**

For a basic outline of the modules in your CSR management system project, we can break them down into key functional areas. These would typically be:

**1. “Admin Module:”**

* “User Management:” Admins can add, edit, and remove users (organizations and volunteers).
* “Project Management:” Admins oversee the CSR projects, ensuring proper execution.
* “Report Generation:” Admins generate reports based on data collected from organizations and volunteers.

**2. “Organization Module:”**

* + - * “Project Creation:” Organizations can create and manage CSR projects, including event planning.
    - “Event Management:” Organizations can schedule events and invite volunteers to participate.
    - “Impact Reporting:” Organizations generate reports to highlight the impact of their CSR activities.

**3. “Volunteer Module:”**

* “Registration:” Volunteers can register and create profiles.
* “Event Participation:” Volunteers can view and sign up for CSR events.
* “Feedback and Ratings:” Volunteers can provide feedback on the events they participated in.

**4.2 Data Design**

The “Data Design” of the CSR management system focuses on structuring data to ensure efficient storage, retrieval, and management across the platform. The core entities include “Admin”, “Organization”, “Volunteer”, “CSR Project”, “Event”, and “Impact Report”, each represented by distinct tables in the database. The “Admin” entity is responsible for managing users and overseeing CSR initiatives, thus storing details like admin credentials, contact information, and privileges. The “Organization” entity includes data related to the companies or non-profits involved in CSR activities, such as organization details, projects created, and event management. “Volunteers” store personal data, including contact information, participation history, and feedback they provide on CSR events.

The “CSR Project” and “Event” entities track the projects created by organizations and the events associated with them, including the project's name, description, schedule, and outcomes. Lastly, “Impact Reports” store data generated by the system to assess the effectiveness of CSR initiatives, with metrics on volunteer involvement, event success, and social impact. The data is normalized to reduce redundancy, and relationships between entities are defined using foreign keys to ensure data integrity. This data design ensures that the system is scalable, maintains performance, and supports easy retrieval and reporting of CSR data.

**4.2.1 Schema Design**

The Schema Design for the CSR management system outlines the structure of the database and the relationships between key entities. Below is a simplified schema, showing the primary tables, attributes, and relationships for each core module: Admin, Organization, Volunteer, CSR Project, Event, and Impact Report.

**Admin Table**

* admin\_id (Primary Key)
* username
* email
* contact
* password
* role (defines privilege level)

**Organization Table**

* organization\_id (Primary Key)
* name
* address
* contact\_person
* email
* contact\_number
* registration\_date

**Volunteer Table**

* volunteer\_id (Primary Key)
* username
* email
* contact
* password
* registration\_date

**CSR Project Table**

* project\_id (Primary Key)
* organization\_id (Foreign Key, references Organization)
* project\_name
* description
* start\_date
* end\_date

**Event Table**

* event\_id (Primary Key)
* project\_id (Foreign Key, references CSR Project)
* event\_name
* description
* event\_date
* location
* max\_volunteers

**Impact Report Table**

* report\_id (Primary Key)
* project\_id (Foreign Key, references CSR Project)
* event\_id (Foreign Key, references Event)
* total\_volunteers
* impact\_score (qualitative/quantitative metric of impact)

**4.2.2 Data Integrity & Constraint**

“Data integrity and constraints” in the CSR management system are essential to ensure that the data stored and processed within the system remains accurate, consistent, and reliable. To maintain data integrity, various constraints are applied at the database level.

* “Primary Key Constraints” ensure that each record in tables like “Admin”, “Organization”, “Volunteer”, and “CSR Project” is uniquely identifiable. For example, each organization has a unique `organization\_id`, and each volunteer has a unique `volunteer\_id`.
* “Foreign Key Constraints” maintain relationships between related tables. For instance, the “Event” table references the “CSR Project” table through the `project\_id`, ensuring that an event cannot exist without a valid project. Similarly, the “Impact Report” table references both the “Project” and “Event” tables, linking reports to actual data.
* “Not Null Constraints” are applied to critical fields such as user emails, project names, and event dates to prevent incomplete data entries that could lead to errors or inconsistencies.
* “Unique Constraints” enforce that certain fields, like user email addresses and project names, must be unique across the system to avoid duplication, ensuring that each user or project is easily identifiable.
* “Check Constraints” ensure data validity by restricting the values that can be entered into certain fields. For instance, event dates must be in the future, and volunteer capacities must be positive numbers.

These constraints collectively ensure that data integrity is maintained, preventing issues such as duplicate entries, orphaned records, or incomplete data. They help the system enforce business rules and ensure consistency, even as data grows and changes.

**4.3 Procedural Design**

The “Procedural Design” of the CSR management system focuses on defining the processes, workflows, and operations that guide how the system functions. It describes how different modules interact and how tasks are executed in a step-by-step manner. In this system, the key modules include the “Admin”, “Organization”, and “Volunteer” modules, each with its own procedures and responsibilities.

The “Admin” module governs system control and management, including user role assignment, project approval, and event oversight. Admins have procedures for adding new organizations, verifying their legitimacy, and ensuring they comply with CSR standards. They also manage user accounts, moderate activities, and generate impact reports.

The “Organization” module involves processes for creating and managing CSR projects. Organizations follow a defined workflow for project creation, including setting goals, defining timelines, and planning events. Once a project is created, it follows through various phases, such as project approval, event scheduling, volunteer recruitment, and impact tracking. The system ensures that organizations adhere to the rules and provide meaningful impact reports post-project.

The “Volunteer” module handles volunteer registration, project engagement, and event participation. Volunteers can browse available events, sign up for those that match their interests, and track their involvement history. The procedural design for volunteers also includes processes for feedback, communication with organizations, and receiving updates on the project's progress.

**4.3.1 Logical Design**

**4.3.1.1 Use Case Diagram**

A use case describes the behavior of a system from the perspective of the user. It describes the functional aspects of a system and its major processes. It provides a graphical representation of system users and the types of interactions that occur within the system.

A use case diagram’s primary goal is to illustrate a system's dynamic nature. The system's requirements, which take into account both internal and external factors, are accumulated. It refers to individuals, use cases, and a number of other things that refer to the actors and components responsible for putting use case diagrams into practise. It displays the potential interactions between a system component and an external entity.

#### Graphical user interface, text, application, email Description automatically generated Elements used in Use Case diagram:

Fig 4.3.1.1.1 Use-Case elements

**4.3.1.1.1 Use-Case Diagram(Admin,Organization,Volunteer)**

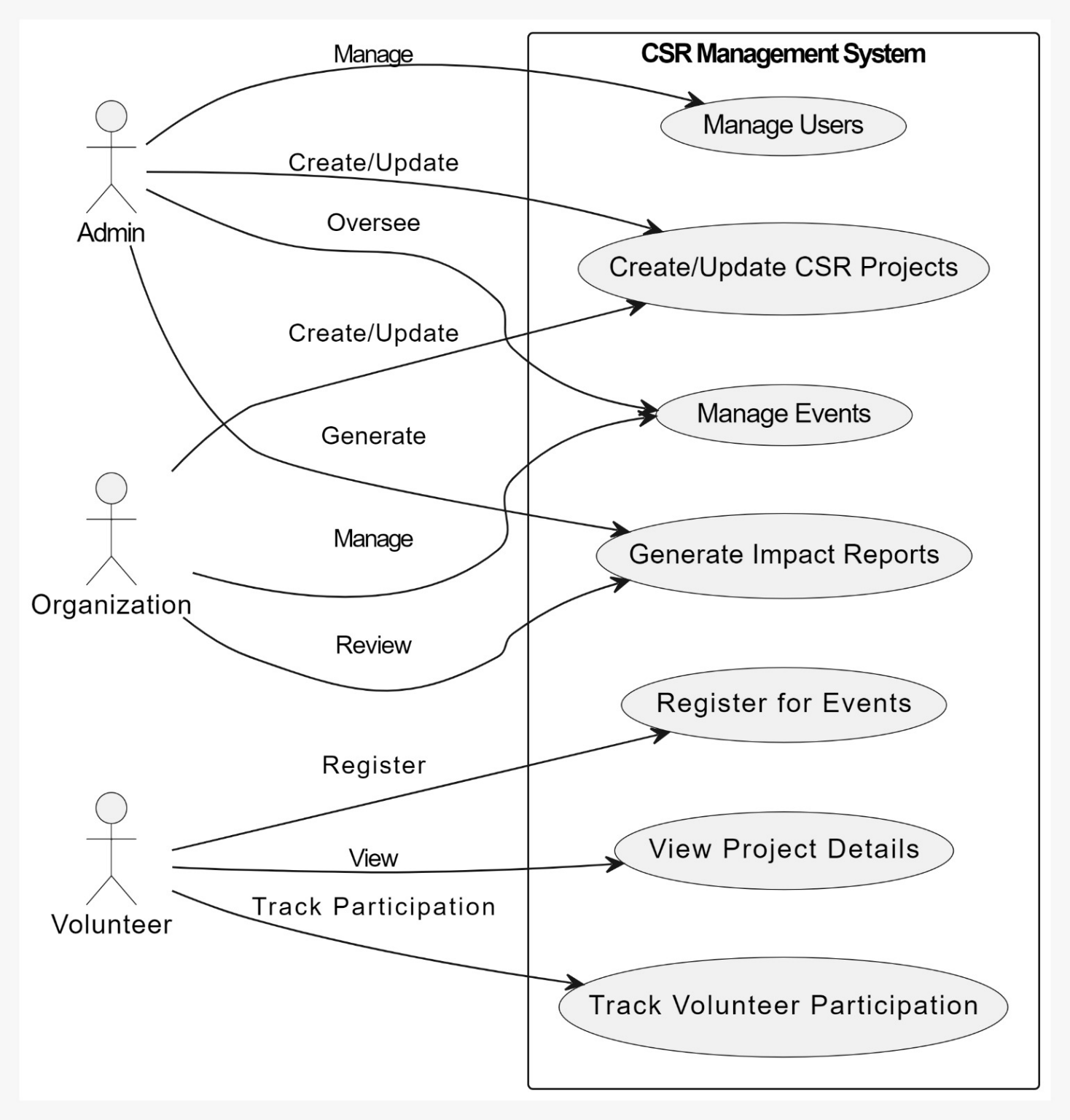
****

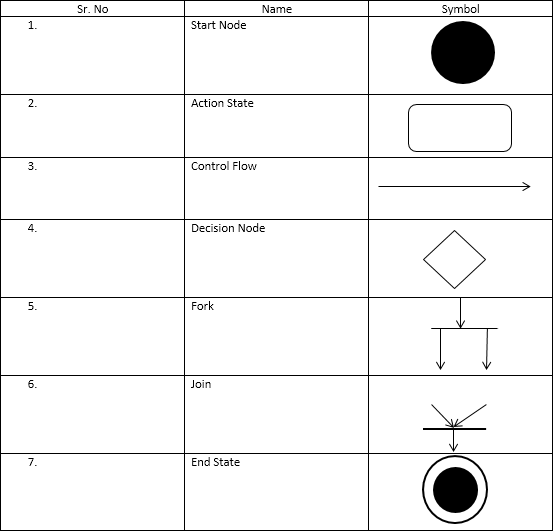
Fig 4.3.1.1.1 Use-Case(Admin,Organization,Volunteer)

**4.3.1.2 Activity Diagram**

An activity diagram depicts the flow of activities. An activity is a non-atomic execution that is ongoing within a state machine. Activities eventually lead to an action, which is composed of executable atomic computations that result in a change in the state of a system or the return of a value. An activity diagram is made up of vertices and arcs. Activity diagrams typically include activity states and action states, as well as transitions and objects.

An action state is a programmable atomic computation that can be used to perform an operation on an object, send a signal to an object, or even create or destroy an object.

An activity state is a composite whose control flow is made up of activity states and action states. An activity state may include extra components such as entry and exit actions.

 Table 4.3.1.2.1 Activity elements table

**4.3.1.2.1 Activity Diagram(Admin,Organization,Volunteer)**

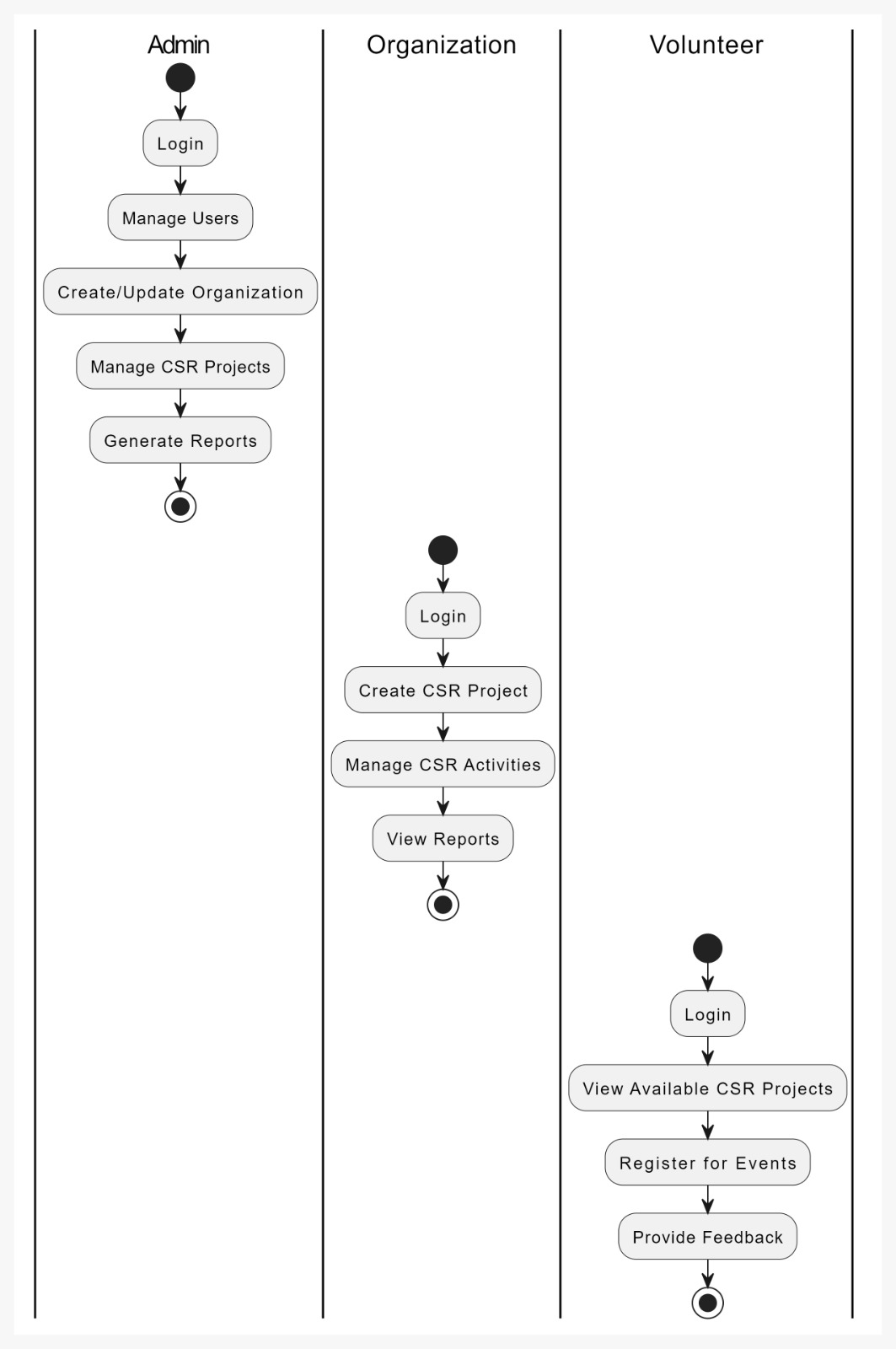


Fig 4.3.1.2.1 Activity Diagram

**4.3.2 Data Structure**

Hash Table (also known as Hash Map)

A Hash Table is a fundamental data structure that allows for fast data access and efficient storage, making it a key component in systems where quick retrieval of information is necessary. In the context of your CSR management system, the hash table would be especially useful for managing user details, project data, and other frequently accessed information.

How It Works

A hash table works by using a hash function to convert keys (like a user ID, email, or project name) into a specific index that maps to the data stored in the table. Instead of searching through an entire list or array, the system can directly access the correct location where the data is stored, making lookup operations extremely fast—usually in constant time (O(1)).

Applications in the CSR Management System

User Management (Admins, Organizations, Volunteers): Each user in the system (admins, organization members, and volunteers) is assigned a unique identifier like an email or user ID. A hash table can store these user records with the key being the user ID and the value being the user’s profile (such as name, contact information, and role). When a user logs in or an admin needs to manage user information, the system can quickly locate the corresponding data using the hash table.

Example Use Case:

When a volunteer logs in, the system looks up their username or email in the hash table, retrieves their information, and verifies their password—all in a matter of milliseconds. This ensures a smooth user experience, especially in systems with many users.

CSR Project and Event Lookup: CSR projects and events are key elements of the system. The hash table can be used to store project information where each project’s name or ID acts as a key, and the corresponding value holds project details (such as description, timeline, status, and associated events). Similarly, events under each project can be stored with quick access keys.

Example Use Case:

When an organization or admin needs to update a CSR project or view an event, the hash table enables the system to instantly locate the relevant project by its ID or title. This significantly reduces time, especially when there are a large number of ongoing projects.

Volunteer Registration and Event Participation: For event participation, the system could use a hash table to track which volunteers have registered for specific events. The key would be the event ID, and the value would be a list or set of volunteer IDs. This makes it very efficient to add, remove, or check which volunteers are registered for each event.

**4.3.3 Algorithm Design**

The algorithm design for the CSR management system defines how core operations are handled, ensuring efficiency, scalability, and responsiveness. Below are the key algorithms for major functions within the system:

**1. “User Registration and Authentication (Admin, Organization, Volunteer)”**

* + “Input”: User details (email, password, role)
  + “Process”:
    1. Check if the email already exists in the system (using a “hash table” for fast lookup).
    2. If not, hash the password for security.
    3. Store the new user in the appropriate table based on the role (admin, organization, volunteer).
    4. Send a confirmation email or notification.
  + “Output”: Success message or error if the email already exists.
  + “Time Complexity”: O(1) for email lookup using a hash table.

**2. “CSR Project Creation”**

* + “Input”: Project details (project name, description, start date, organization ID)
  + “Process”:
    1. Verify the organization’s credentials.
    2. Check for any overlapping projects within the same time frame (using a “sorted list” or “binary search” to quickly find conflicting dates).
    3. Add the project details to the “project table”.
    4. Notify stakeholders (volunteers, admins) about the new project.
  + “Output”: Project created message or error if conflicts exist.
  + “Time Complexity”: O(log n) for date conflict checking, O(1) for project addition using a hash table.

**3. “Event Scheduling and Volunteer Registration”**

* + “Input”: Event details (event name, project ID, location, date), volunteer ID
  + “Process”:
    1. Retrieve the project ID from the “project table” to ensure the event is linked to an active CSR project.
    2. Add event details to the “event table”.
    3. Volunteers can register for the event by entering their ID.
    4. Verify event capacity by comparing the number of registered volunteers to the maximum allowed.
    5. Add the volunteer’s ID to the “event’s participant list”.
    6. Send a notification to the volunteer about their registration status.
  + “Output”: Event scheduled and volunteer registered, or error if event is full.
  + “Time Complexity”: O(1) for project/event lookup and volunteer registration, O(n) for capacity check.

**4. “Impact Reporting”**

* + “Input”: Project and event data (volunteer count, resources used, project goals achieved)
  + “Process”:
    1. Collect all event data related to a project.
    2. Calculate total resources used, number of volunteers, and whether project goals were met.
    3. Generate a “report” containing metrics like the number of participants, project success rate, and impact score.
    4. Store the report in the “impact report table”.
    5. Notify the organization and admin of the report’s completion.
  + “Output”: Impact report generated and stored.
  + “Time Complexity”: O(n) for collecting event data and generating reports (n is the number of events in the project).

**5. “Search for Projects and Events”**

* + “Input”: Search query (project name, event name, date, location)
  + “Process”:
    1. Parse the search query and extract the key terms.
    2. Use a “hash table” or “indexing” for fast project/event lookups based on the name, date, or location.
    3. Return a list of relevant projects or events.
  + “Output”: List of matching projects or events.
  + “Time Complexity”: O(1) for hash table lookups, O(log n) for searches using indexed data.

**6. “Volunteer Feedback Submission”**

* + “Input”: Event ID, volunteer ID, feedback text, rating
  + “Process”:
    1. Verify that the volunteer participated in the event (cross-check using the “event participant list”).
    2. Store the feedback and rating in the “feedback table” associated with the event and volunteer.
    3. Optionally, calculate the average rating for the event.
    4. Notify the organization of the feedback.
  + “Output”: Feedback stored and event rating updated.
  + “Time Complexity”: O(1) for participant verification and feedback storage.

4.4 UI Design

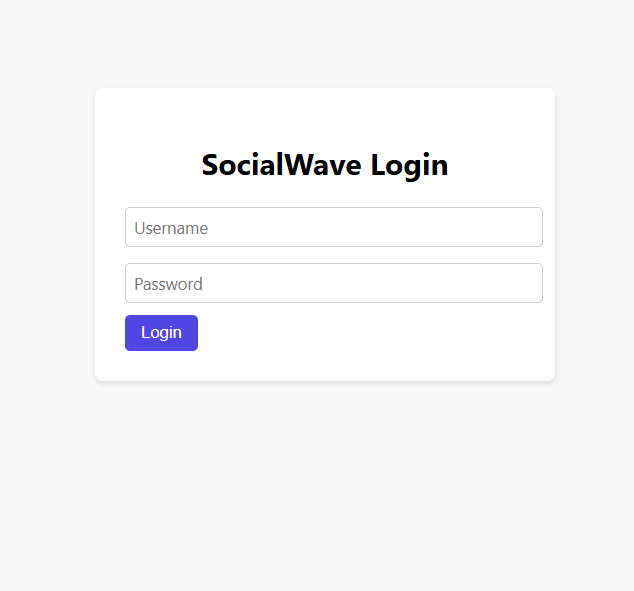


Fig 4.4.1 Socialwave Login Page UI

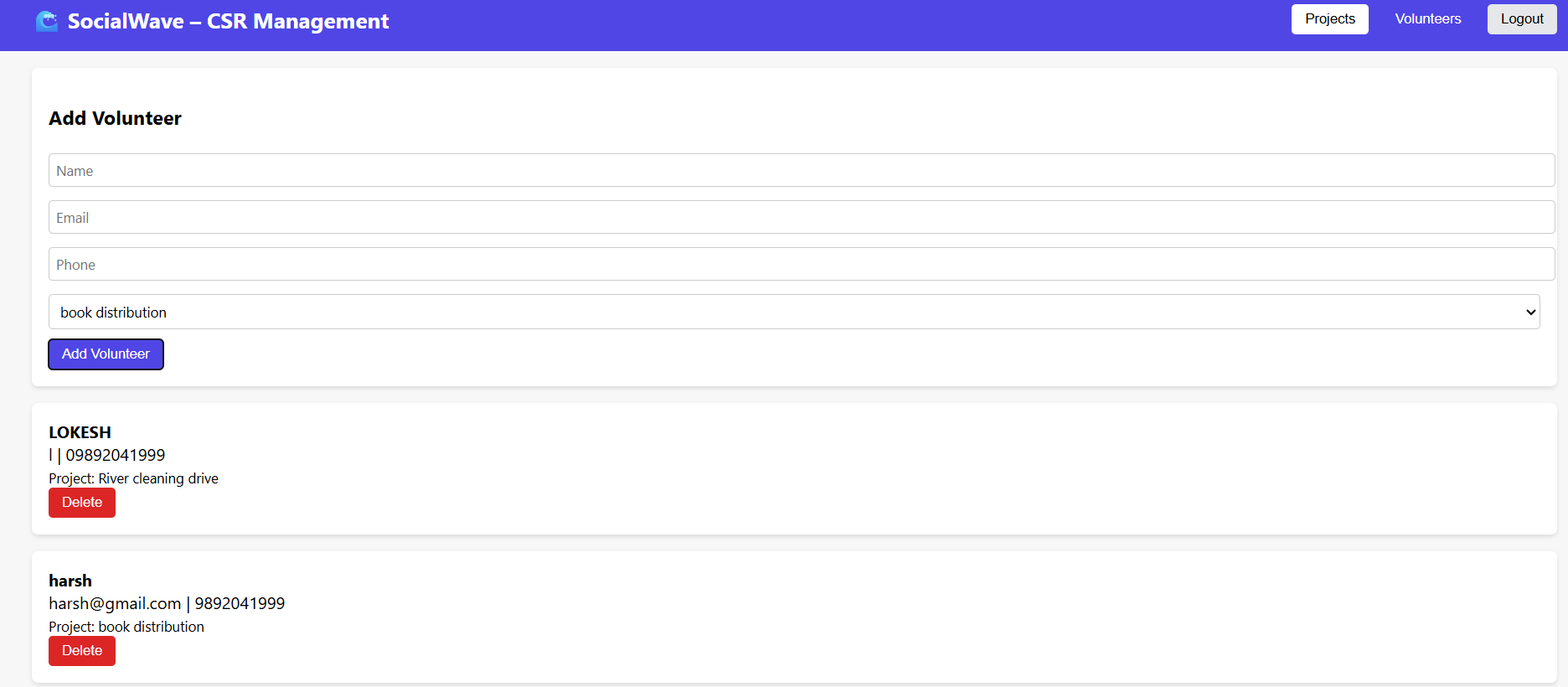


Fig 4.4.2 Event Management Dashboard

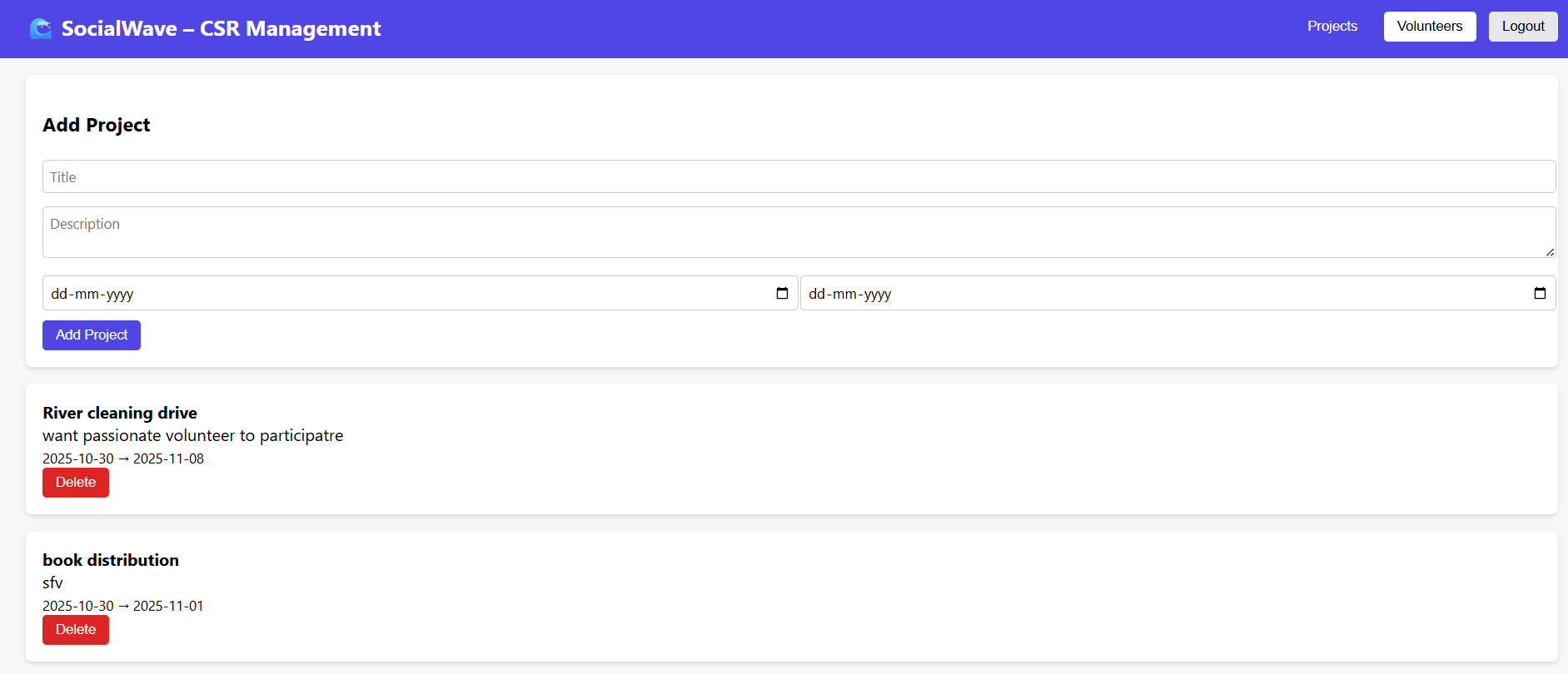


Fig 4.4.3 Organization Dashboard

**4.5 Security Issues**

While developing and managing a CSR (Corporate Social Responsibility) management system, it’s important to address a range of security concerns to protect user data, ensure system integrity, and prevent malicious activity. Below are some of the key security issues to consider:

**1. “Data Breaches”**

* + “Issue”: Unauthorized access to sensitive user data (e.g., admin, organization, and volunteer details).
  + “Consequences”: Exposure of personal information, including emails, passwords, and contact information, can lead to identity theft or misuse of information.
  + “Mitigation”:
  + Use encryption (e.g., AES, SSL/TLS) for data both in transit and at rest.
  + Ensure robust access control mechanisms to prevent unauthorized access.
  + Employ data masking techniques for displaying sensitive data only to authorized users.

**2. “Weak Passwords and Poor Authentication”**

* “Issue”: Users may use weak passwords, making it easier for attackers to compromise accounts through brute force attacks or credential stuffing.
* “Consequences”: Attackers may gain unauthorized access to admin, organization, or volunteer accounts.
* “Mitigation”:Implement strong password policies (e.g., minimum length, complexity requirements).
* Enforce multi-factor authentication (MFA) for additional security.
* Regularly prompt users to update passwords and monitor login attempts for suspicious activity.

**3. “SQL Injection and NoSQL Injection”**

* + “Issue”: Poorly constructed or unvalidated input fields can allow attackers to manipulate queries, injecting malicious SQL/NoSQL commands to gain unauthorized access or manipulate database records.
  + “Consequences”: Attackers can access, modify, or delete sensitive data, or even gain administrative control over the system.
  + “Mitigation”: Always use parameterized queries or prepared statements to interact with the database.
  + Validate and sanitize all user input to prevent the injection of malicious code.
  + Employ web application firewalls (WAF) to detect and block injection attempts.

**4. “Cross-Site Scripting (XSS)”**

* + “Issue”: XSS vulnerabilities occur when attackers inject malicious scripts into web pages viewed by other users, potentially stealing session cookies or manipulating the site.
  + “Consequences”: Attackers can hijack user sessions, steal credentials, or spread malware.
  + “Mitigation”:Sanitize and escape all user-generated content before rendering it in the browser.
  + Use security headers like Content Security Policy (CSP) to block the execution of untrusted scripts.
  + Encode all special characters in output to prevent script execution.

**5. “Cross-Site Request Forgery (CSRF)”**

* + “Issue”: CSRF attacks trick users into performing actions they did not intend, such as submitting forms or changing settings, by exploiting their authenticated session.
  + “Consequences”: An attacker could alter user account settings, modify data, or perform actions on behalf of the user without their consent.
  + “Mitigation”: Implement CSRF tokens for all state-changing requests.
  + Validate the origin of requests to ensure they are coming from legitimate sources.
  + Use SameSite cookies to restrict how cookies are sent with cross-site requests.

**6. “Session Hijacking”**

* + “Issue”: Attackers can steal session tokens (through XSS or network sniffing) and impersonate legitimate users.
  + “Consequences”: Attackers can gain access to a user’s account without needing their credentials.
  + “Mitigation”:Use secure cookies with HTTPOnly and Secure flags to prevent theft of session cookies.
  + Regularly rotate session tokens and enforce automatic session expiration.
  + Implement strict idle and absolute session timeouts.

**7. “Insufficient Access Controls”**

* + “Issue”: Improper role-based access controls (RBAC) can allow unauthorized users to access restricted areas or functionalities within the system.
  + “Consequences”: Unauthorized users, such as volunteers or low-level users, may gain access to sensitive data or administrative functions.
  + “Mitigation”:Enforce strict role-based access control (RBAC) policies to ensure users only have access to the data and functions necessary for their role.
  + Implement least privilege principles, ensuring users are assigned minimal necessary permissions.
  + Regularly audit access controls and user privileges to detect any anomalies.

**8. “Insecure APIs”**

* + “Issue”: Unprotected or insecure API endpoints could be exploited by attackers to gain unauthorized access to data or perform actions on behalf of users.
  + “Consequences”: Exposing sensitive data or allowing unauthorized actions through APIs can lead to data leaks or service disruptions.
  + “Mitigation”:Secure all API endpoints with proper authentication (e.g., OAuth 2.0) and authorization mechanisms.

**4.6 Test Case Design**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Description** | **Test Steps** | **Test Inputs** | **Expected Output** |
| Register New User (Volunteer) | 1. Open Registration Page 2. Enter valid details 3. Submit | Username, email, password | New user should be created and confirmation displayed |
| Register User with Existing Email | 1. Open Registration Page  2. Enter existing email  3. Submit | Existing email | Error message indicating email already exists |
| Login with Correct Credentials | 1. Open Login Page 2. Enter valid credentials 3. Submit | Email, password | User should successfully log in |
| Login with Incorrect Credentials | 1. Open Login Page 2. Enter wrong password 3. Submit | Email, incorrect password | Error message for incorrect credentials |
| Create New CSR Project | 1. Login as Organization 2. Open "Create Project" 3. Submit | Project name, description, dates | New project created and visible in project list |
| Create Project with Empty Fields | 1. Leave required fields empty 2. Submit | Empty project details | Error message for mandatory fields |
| Schedule New Event | 1. Open "Create Event" 2. Enter valid details 3. Submit | Event name, date, location | Event created and linked to project |
| Register Volunteer for Event | 1. Login as Volunteer 2. Browse events 3. Click "Register" | Volunteer ID, event ID | Volunteer registered for event |
| Event Capacity Exceeded | 1. Try registering more volunteers than capacity | Event ID, max capacity | Error message indicating event is full |
| Submit Feedback for Event | 1. Navigate to completed events 2. Submit feedback | Volunteer ID, feedback text, rating | Feedback saved and visible under the event |
| Generate Impact Report | 1. Login as Admin 2. Select project 3. Submit | Project ID | Impact report generated and available for download |
| SQL Injection Test | 1. Attempt SQL injection in input fields | Malicious SQL query | Input sanitized, rejected |
| Cross-Site Scripting (XSS) | 1. Inject script in form fields | Malicious script | Script doesn’t execute, input sanitized |
| Load Test for Event Registration | 1. Simulate multiple users registering for the same event simultaneously | 100+ simultaneous registrations | System handles high load without crashing |
| Response Time Test | 1. Measure response time for user login and project creation | N/A | System responds within acceptable time (<2s) |

**Conclusion**

The CSR Management System was developed to simplify and automate the management of Corporate Social Responsibility projects, volunteers, and organizational activities. This system allows easy registration, login, and project management while ensuring transparency and accountability. It connects organizations with volunteers through a unified platform, promoting social impact and collaboration.

Through the implementation of this project, we successfully demonstrated how technology can support social responsibility initiatives, improve data handling, and provide real-time interaction between users. Future enhancements can include analytics dashboards, report generation, and integration with mobile platforms to make CSR efforts more accessible and impactful.

**REFERENCES**

 Carroll, A. B. (1999). *Corporate Social Responsibility: Evolution of a Definitional Construct.* Business & Society, 38(3), 268–295.

 ISO 26000:2010 – *Guidance on Social Responsibility.* International Organization for Standardization.

 World Business Council for Sustainable Development (WBCSD). *Corporate Social Responsibility: Making Good Business Sense*, 2000.

 TutorialsPoint. *Node.js and MySQL Tutorial.* Available at: https://www.tutorialspoint.com/nodejs/nodejs\_mysql.htm

 ReactJS Official Documentation. *Building User Interfaces with React.* Available at: <https://react.dev>

 W3Schools. *HTML, CSS, and JavaScript Tutorials.* Available at: <https://www.w3schools.com>

 MySQL Documentation. *MySQL Developer Guide.* Available at: https://dev.mysql.com/doc

 GitHub – *CSR Management System open-source projects and templates.*