PRO-VERTOS

Software Requirements Specification

Course Code : INT252

Student Name : Sunil Kumar Mehta

Reg Number : 12218623

Prepared for
Continuous Assessment 3
Autumn 2024

PROJECT REPORT

Table of Contents

1. Introduction

Pro-vertos is a Unique and dynamic web platform designed to facilitate seamless connections among vertos, students, and volunteers, fostering collaboration between diverse student organizations by providing the single platform to manage their event and showcase. Our platform serves as a catalyst for synergy, enabling individuals and groups to effortlessly engage, share resources, and amplify their impact within the academic community.

In today's interconnected world, building meaningful connections is paramount and Events play a crucial role in connecting peoples. Pro-vertos emerges as a vital bridge, bridging the gap between vertos seeking to join events, students eager to contribute, and volunteers ready to lend their expertise in the events. By harnessing the power of connectivity, we aim to cultivate a vibrant ecosystem where knowledge exchange and collaboration thrive.

1.1 Purpose

The purpose of Pro-vertos is to revolutionize the landscape of student engagement and collaboration within the academic and Non academic communityies and manage their events. It aims to provide a comprehensive and intuitive platform that empowers users to connect, communicate, and collaborate effectively, ultimately enhancing the academic experience for all stakeholders involved**.**

1.2 Scope

Connection Facilitation: The platform facilitates connections among vertos, students, and volunteers within the academic Non academic communities. It provides a user-friendly interface for individuals and groups to connect with each other, fostering collaboration and knowledge exchange.

Collaboration between Student Organizations: Pro-vertos serves as a catalyst for synergy by enabling collaboration between diverse student organizations. It provides tools and features that empower organizations to work together, share resources, and amplify their impact on campus.

Resource Sharing: The platform allows users to effortlessly share resources, such as event information, documents, and expertise. It

provides a centralized repository for users to access and contribute to shared resources, enhancing collaboration and productivit.

Knowledge Exchange: Pro-vertos facilitates knowledge exchange within the academic community by providing forums, discussion boards, and other interactive features. It encourages users to share ideas, ask questions, and engage in meaningful discussions to enhance learning and collaboration.

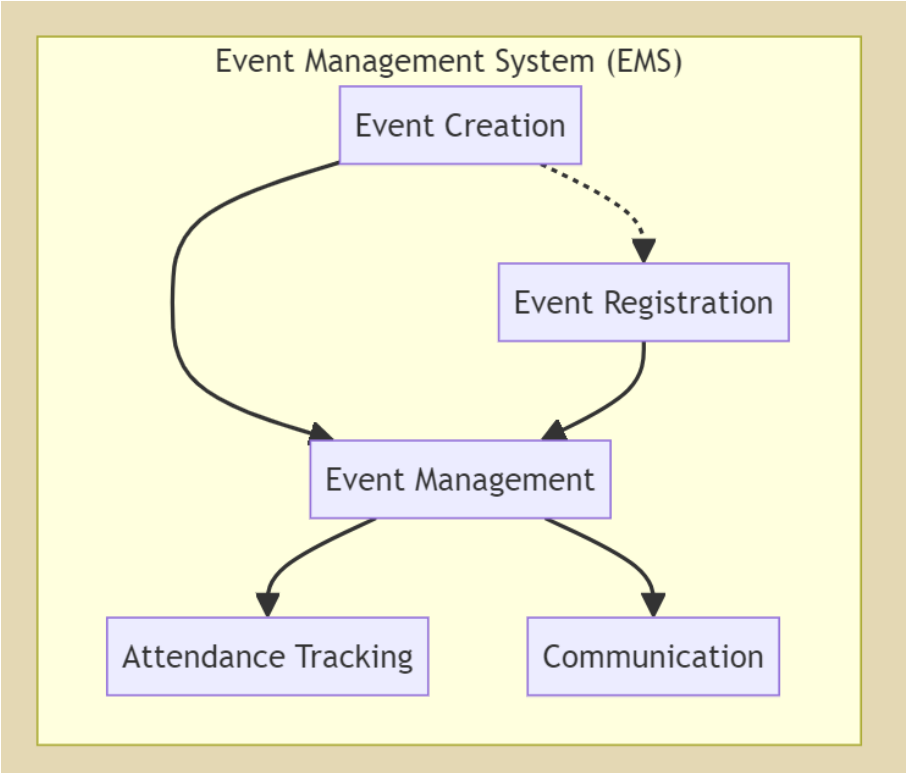
Bridge for Vertos, Students, and Volunteers: The platform serves as a vital bridge between vertos seeking guidance, students eager to contribute, and volunteers ready to lend their expertise. It provides opportunities for mentorship, networking, and professional development, enriching the academic experience for all stakeholders.

Ecosystem Cultivation: By harnessing the power of connectivity, Pro-vertos aims to cultivate a vibrant ecosystem where knowledge exchange and collaboration thrive. It provides a supportive environment for individuals and groups to connect, collaborate, and make a positive impact within the academic community**.**

1.3 Definitions, Acronyms, and Abbreviations

Event Management System (EMS):

Definition: A software application designed to facilitate the planning, organization, and coordination of events, including scheduling, attendee management, and logistics.

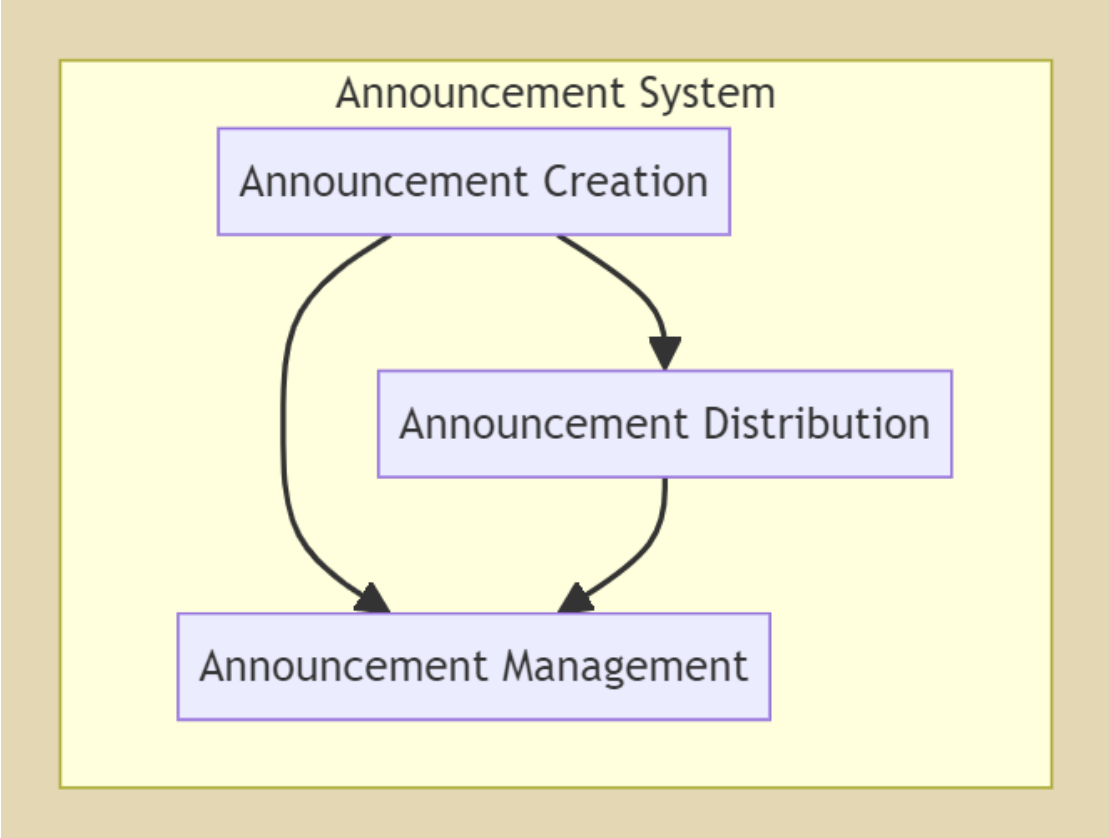


Student Organization Management System (SOMS):

Definition: A platform specifically tailored for managing student organizations within an academic institution, providing tools for membership management, event planning, and communication.

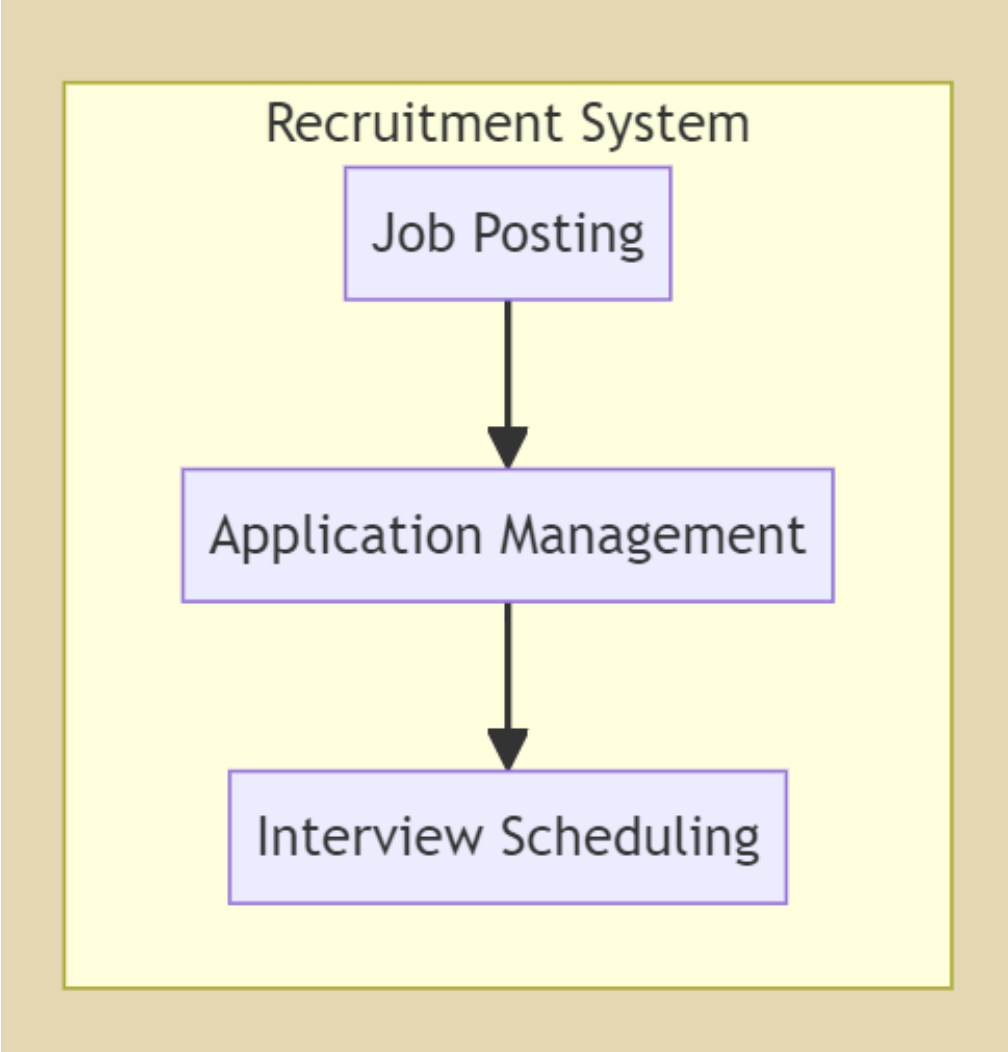
Announcement System:

Definition: A feature or module within a larger software platform that enables the dissemination of important announcements, updates, or notifications to users.


Announcement System
Announcement Creation
Announcement Distribution
Announcement Management

Recruitment System:

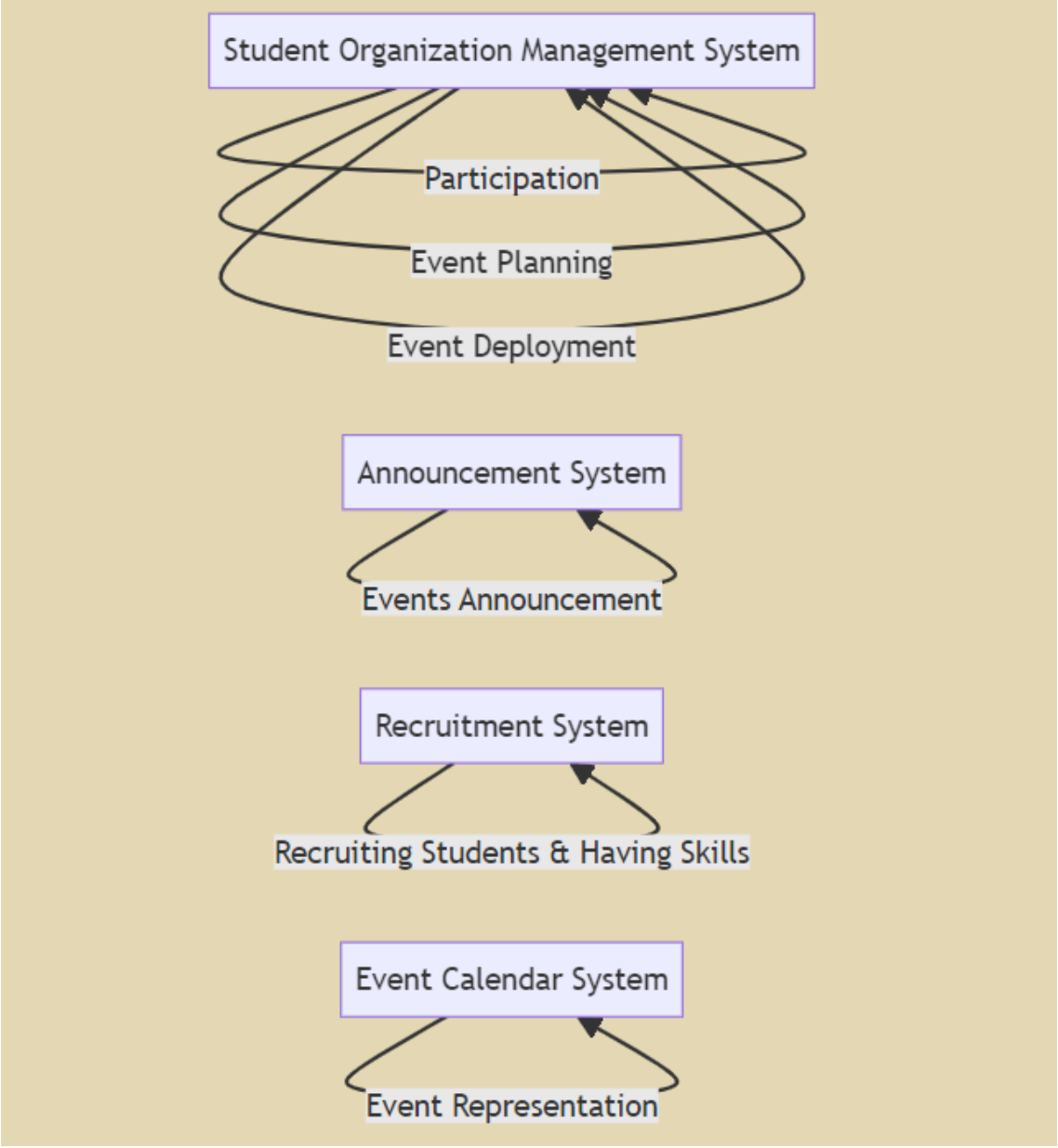
Definition: A software application or process used for recruiting and onboarding individuals, including job applicants, volunteers, or members, into an organization or program.



Event Calendar System (ECS):

Definition: A component of a software platform that provides a visual representation of scheduled events, allowing users to view, search, and manage event details and schedules**.**

Combined DFD for the project



1.4 References

Australian Centre for Event Management. (2000) Events Beyond 2000: Setting the Agenda Proceedings of Conference on Event Evaluation, Research and Education

By **School of Leisure, Sport and Tourism. University of Technology, Sydney**

Your 2024 Event Management Guide: Tips, Tools, and Resources on https://www.eventbrite.com/blog/what-is-event-management/

1.5 Overview

The Software Requirement Specification (SRS) for the Pro-vertos project, outlines the detailed requirements and functionalities of each system. These systems are essential components of a comprehensive software platform designed to streamline event planning, organizational management, communication, recruitment, and scheduling processes within academic institutions.

(1) What the rest of the SRS contains:

The remainder of the SRS provides a comprehensive breakdown of the requirements, functionalities, constraints, and dependencies of each system. It includes detailed descriptions of user requirements, system features, performance metrics, security measures, and integration points. Additionally, the SRS outlines the system's interaction with external interfaces, data flows, and error handling procedures. Moreover, it specifies non-functional requirements such as performance, reliability, security, maintainability, and portability aspects.

(2) How the SRS is organized:

The SRS is organized into sections dedicated to each system, namely the Event Management System (EMS), Student Organization Management System (SOMS), Announcement System, Recruitment System, and Event Calendar System (ECS). Each section begins with a concise definition of the system, followed by detailed requirements and specifications. The organization of the SRS ensures clarity and ease of reference, allowing

stakeholders to navigate through the document efficiently and understand the specific requirements and functionalities of each system

2. General Description

This section of the SRS provides a broad overview of the systems and their related components, without specifying detailed requirements.

2.1 Product Perspective

These systems exist within the broader context of software applications designed to streamline various organizational processes. They may interact with other systems such as scheduling software, communication platforms, or information systems within academic institutions

2.2 Product Functions

The primary functions of these systems are to facilitate efficient planning, organization, and coordination of events, recruitment processes, announcement dissemination, and management of student organizations. This includes features such as event scheduling, attendee management, logistical coordination, membership management, communication tools, recruitment processes, and announcement distribution.

2.3 User Characteristics

The eventual users of these systems may include event organizers, administrative staff, student organizations, job applicants, volunteers, and attendees. User characteristics such as familiarity with organizational processes, technological proficiency, and organizational roles may influence system requirements.

2.4 General Constraints

Constraints affecting the development of these systems may include budgetary limitations, technological infrastructure constraints, compatibility with existing systems, and regulatory compliance requirements.

2.5 Assumptions and Dependencies

Assumptions and dependencies related to these systems include factors such as the availability of necessary hardware and software infrastructure, adherence to organizational policies and procedures, and the cooperation of stakeholders in providing relevant information and resources for system implementation. These factors may impact the requirements and development process of the systems.

3.Specific Requirements

Event Management System (EMS):

1. **Membership Management:**
   - Manage user accounts and memberships for student organizations.
   - Allow users to join or leave organizations.
   - Maintain a database of active members for each organization.
   - Provide administrative controls for managing memberships.

2. **Event Planning:**
   - Assist in planning and organizing events for student organizations.
   - Allow creation and management of event schedules, tasks, and assignments.
   - Provide collaboration features for event planning among organization members.
   - Integrate with the Event Management System for seamless event creation and management.

3. **Communication:**
   - Enable communication within student organizations.
   - Support announcements, discussions, and group messaging features.
   - Ensure that communication channels are accessible to all organization members.
   - Allow for sharing of files, documents, and resources among members.

4. **Announcement System:**
   - **Announcement Creation:**
     - Allow authorized users to create announcements.

- Include features for formatting, attaching files, and scheduling announcements.
- Ensure that announcements are relevant and informative.

- **Announcement Distribution:**
  - Distribute announcements to targeted recipients or organization members.
  - Support multiple distribution channels such as email, website, or mobile app notifications.

- **Announcement Management:**
  - Provide controls for managing and organizing announcements.
  - Allow for archiving or deleting outdated announcements.
  - Enable search and filter functionalities for easy retrieval of announcements.

Recruitment System:

1. **Job Posting:**
   - Allow organizations to post job vacancies or volunteer opportunities.
   - Include details such as job description, requirements, and application deadlines.
   - Ensure that job postings are visible to relevant users.

.

3.1 External Interface Requirements

3.1.1 User Interfaces

User Login Interface:

Provide a secure login interface for users to access the system.
Include fields for username and password.
Implement mechanisms for password recovery and account registration if applicable.
Ensure the interface is intuitive and user-friendly.

Event Creation Interface:

Design an interface for users to create new events.
Include fields for event details such as name, date, time, location, and description.
Implement validation mechanisms to ensure data accuracy.
Provide options for adding additional event features if necessary, such as image uploads or event categories.

Event Registration Interface:
Develop a registration interface for users to sign up for events.
Display event details and registration options clearly.
Allow users to select desired events and provide necessary registration information.
Provide feedback on successful registrations and error messages for unsuccessful attempts.

Announcement Creation Interface:
Create an interface for authorized users to compose announcements.
Include formatting options for text, attachment upload functionality, and scheduling options if required.
Ensure that the interface supports multimedia content if necessary.

3.1.2 Hardware Interfaces

Server Hardware:
Specify the hardware requirements for hosting the software system.
Define the server specifications including CPU, RAM, storage capacity, etc.
Ensure compatibility with the software and expected user load.

User Devices:
Identify the supported user devices such as desktops, laptops, tablets, and smartphones.
Specify any hardware requirements or constraints for accessing the system from different devices.
Ensure compatibility with various operating systems and browsers

3.1.3 Software Interfaces

Database Management System (DBMS):

Specify the software interface requirements for interaction with the database.
Identify the DBMS to be used (e.g., MongoDB).
Define data exchange protocols and query languages if necessary.

Integration with External Systems:
Identify any external systems or APIs (Application Programming Interfaces) that the software system needs to interact with.

Define the communication protocols and data formats required for integration.
Specify security measures for protecting data during exchange.

3.1.4 Communications Interfaces

Internal Communication:
Define the communication protocols for internal system components.
Specify how different modules or subsystems will communicate with each other.
Ensure reliability, security, and efficiency of communication channels within the system.

External Communication:
Identify external communication channels such as email, SMS, or social media platforms.
Specify the protocols and APIs required for sending notifications, announcements, or reminders to users.
Ensure compliance with relevant privacy regulations and standards for external communication.

3.2 Functional Requirements

3.2.1 User Authentication:
The system shall provide user authentication using bcrypt for password hashing and jwt for token generation. This ensures that only authorized users can access certain routes and functionalities.

3.2.2 Data Storage:
The system shall store user data and ticket data in a MongoDB database using Mongoose as an Object Data Modeling (ODM) tool.

3.2.3 Data Transfer:
The system shall use express.js to handle HTTP requests and responses. It shall use JSON format for data transfer.

3.2.4 Cross-Origin Resource Sharing:
The system shall handle Cross-Origin Resource Sharing (CORS) to allow or restrict requested resources on a web server depending on where the HTTP request was initiated.

3.5 Non-Functional Requirements

3.5.1 Performance

- Response Time: 95% of user interactions, such as event registration and ticket booking, shall be completed within 2 seconds.
- Scalability: The system should be able to handle a concurrent user load of at least 1000 users without significant degradation in performance.
- Throughput: The system should support a minimum of 100 event registrations per minute during peak usage hours.

3.5.2 Reliability

- Mean Time Between Failures (MTBF): The system should have a MTBF value of more than 30 days, ensuring reliable operation over extended periods.
- Fault Tolerance: The system should be resilient to hardware failures, software bugs, and network disruptions, minimizing the impact of such events on system availability.
- Error Handling: The system shall provide informative error messages and graceful degradation in case of unexpected errors or exceptions.

3.5.3 Availability

- System Uptime: The system shall maintain an uptime of at least 99.9%, ensuring continuous availability for users.
- Downtime Limit: System downtime may not exceed 5 minutes per week, including scheduled maintenance and unplanned outages.

3.5.4 Security

- Data Encryption: User data, including passwords and personal information, shall be encrypted using industry-standard hashing algorithms (e.g., bcrypt).
- Access Control: Access to sensitive system functionalities and data shall be restricted based on user roles and permissions, following the principle of least privilege.
- Data Privacy: The system shall comply with relevant data protection regulations (e.g., GDPR) to ensure the privacy and confidentiality of user information.

3.5.5 Maintainability

- Code Quality: The system codebase shall adhere to coding standards and best practices to ensure readability, maintainability, and ease of future enhancements.

- Documentation: Comprehensive documentation shall be provided for the system architecture, codebase, and deployment processes to facilitate ongoing maintenance and support.
- Modularity: The system shall be modularly designed, allowing for independent updates and modifications to different components without affecting overall system functionality.

3.5.6 Portability

- Cross-Platform Compatibility: The system shall be compatible with major web browsers (e.g., Chrome, Firefox, Safari) and device types (e.g., desktop, tablet, mobile), ensuring consistent user experience across different platforms.
- Deployment Flexibility: The system shall support deployment on various hosting environments, including on-premises servers, cloud platforms (e.g., AWS, Azure), and containerized environments (e.g., Docker, Kubernetes).

3.7 Design Constraints

The system is designed to run on Node.js runtime environment.

The system uses MongoDB as a database, which may limit its compatibility with other types of databases.

The system uses JWT for authentication, which requires secure storage of tokens on the client side.

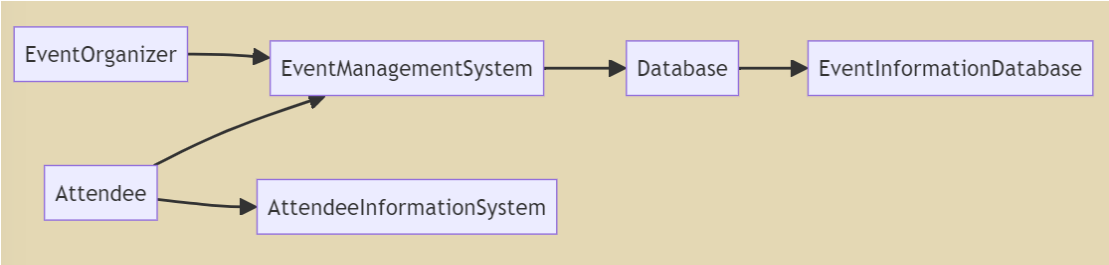
3.9 Other Requirements

The system requires the dotenv package for loading environment variables from a .env file into process.env.

The system requires the cookie-parser middleware to parse Cookie header and populate req.cookies with an object keyed by the cookie names.

4. Analysis Models

4.1 Data Flow Diagrams (DFD)



Event Organizer: The individual or entity responsible for planning and organizing events interacts with the Event Management System (EMS) to input event details, schedule, and manage various aspects of event planning.

Attendee: Users who wish to attend events interact with the system to view event information, register for events, and manage their attendance.

Event Management System (EMS): The central component of the system receives inputs from event organizers, processes them, and stores relevant data in the database. It also provides functionalities for attendees to access event information and manage their attendance.

Attendee Information System: Stores information about attendees, including their profiles, preferences, and attendance records.

Database: Stores all data related to events, including event details, schedules, attendee information, and other relevant data.

Event Information Database: A subset of the main database dedicated to storing information specific to events, including event details, schedules, and related data.

5. GitHub Link : Sunilkumarmehta2002/Pro-vertos

Live Link : Pro-Vertos

A. Appendices

A.1 Appendix 1: Initial Project Scalability Proposal

The application is hosted on a cloud infrastructure and hence it is highly scalable.
The components required to achieve scalability are

1) Automatic load balancing using an AWS Load balancer
The load balancer provides automatic balancing of network traffic between instances. Hence prevents overloading of instances, thus achieving high availability
2) Community instances using AWS EC2
AWS EC2 instances are used to host the application and the instances can be increased based on the number of users
Hence provides high scalability.

A.2 Appendix 2: Expected Billing Module

The project implements a 'pay as you go' utility service model. This module is available for the Event Organizers. They are charged based for the events hosted and the number of students that register to events through our application 'Event Management System'.
Billing – Utility Model is described as below:

1) Event Registrations incurs a basic fee of $10 for each event registered in the application by the Event Organizer
2) For 'Basic' users (Event Organizers having users Less than or Equal to 25), the usage cost is $0
3) For 'Premium' users (Event Organizers having 26 to 100 users), the usage cost is $2/day
4) For 'Premium' users (Event Organizers having 101 to 500 users), the usage cost is $5/day
5) For 'Premium' users (Event Organizers having users greater than 500), the usage cost is $10/day