EVENT MANAGEMENT SYSTEM, GOVERNING COUNCIL DATA COLLECTION SYSTEM, PRODUCT OPTIMIZATION AND ENHANCEMENT

SRIBALAKUMARAN S
Roll No. 21PW36

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

FIVE YEAR INTEGRATED
M.Sc. SOFTWARE SYSTEMS

OF ANNA UNIVERSITY

NOVEMBER 2024

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCES

PSG COLLEGE OF TECHNOLOGY
(Autonomous Institution)
COIMBATORE – 641 004.

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

COIMBATORE – 641 004.

Seventh Semester
Project Work

EVENT MANAGEMENT SYSTEM, GOVERNING COUNCIL DATA COLLECTION SYSTEM, PRODUCT OPTIMIZATION AND ENHANCEMENT

Bona fide record of work done by

SRIBALAKUMARAN S
Roll No. 21PW36

Submitted in partial fulfilment of the requirements for the degree of

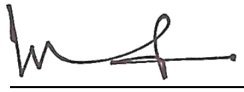
FIVE YEAR INTEGRATED
M.Sc. SOFTWARE SYSTEMS
of Anna University

NOVEMBER 2024

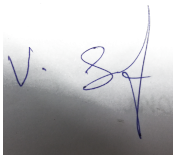
Academic Guide

Head of the Department

Submitted for the Viva-Voce Examination held on 16/11/2024



Internal Examiner



External Examiner

CONTENTS

ACKNOWLEDGEMENT

I am extremely grateful to **Dr. K. Prakasan**, Principal, PSG College of Technology, for permitting me to undertake this project at JarvisLabs.ai, Kovaipudur, Coimbatore.

I sincerely thank **Dr. Nadarajan R**, Director, Department of Applied Mathematics and Computational Sciences and **Dr. Shina Sheen**, Professor and Head, Department of Applied Mathematics and Computational Sciences, for their continual support and motivation.

My deepest appreciation goes to, **Dr. Periakaruppan R M,** Program Coordinator and Associate Professor, Department of Applied Mathematics and Computational Sciences, and to my internal guide and class tutor, **Dr. Mohan K**, Assistant Professor (Sl. Gr.), Department of Applied Mathematics and Computational Sciences, for their valuable encouragement and guidance.

I am thankful to my external guide, **Mr. Vishnu Subramanian**, Founder and CEO, JarvisLabs.ai, Kovaipudur, Coimbatore, and **Dr. Senthil Kumaran V**, Associate Professor, Department of Applied Mathematics and Computational Sciences, for their guidance and support in completing my project work.

I extend my sincere thanks to all the other external guides and mentors for their help in various aspects to do this project work.

Finally, I wish to acknowledge the support of my teammates, the staff of the Department of Applied Mathematics and Computational Sciences, PSG College of Technology, Coimbatore and my family and friends for their encouragement and support.

SYNOPSIS

During my internship period, contributions were made to two organizations, focusing on developing solutions that enhanced operational efficiency and user experience.

At the Department of Applied Mathematics and Computational Sciences, PSG College of Technology, two significant systems:

1. Event Management System (EMS): A comprehensive web platform was designed to organize and manage college events, featuring end-to-end capabilities that foster cross- departmental collaboration and maintain detailed digital records.
2. Governing Council Data Collection System: Data gathering from stakeholders was streamlined, automating the compilation of biannual reports. Key features were implemented to ensure efficient data entry, approval, and report generation, improving transparency and enabling data-driven decision-making.

At JarvisLabs.ai, a cloud computing platform specializing in GPU services for AI/ML applications, several key initiatives were undertaken to optimize platform resources and enhance the user experience. ComfyUI workflows for AI art generation were developed using stable diffusion models, with a focus on ComfyUI Deploy and product development to reduce GPU consumption in ComfyUI. Additionally, the company's analytical capabilities were strengthened through the implementation of data visualization solutions using Streamlit. Various user interface features, such as invoice generation and file system UI, were developed to improve the platform's web presence, while several website bugs were resolved.

These projects contributed significantly to resource optimization, workflow efficiency, and digital transformation initiatives across both organizations.

CHAPTER 1
INTRODUCTION

A detailed description of the organization for which the application is developed is explained in this chapter. It also specifies the system environment used in the development of the interface and technologies and tools used in the development of the interface.

1.1 ORGANIZATION PROFILE

1.1.1 Company Overview

Established in 2019, JarvisLabs.ai is a leading cloud platform provider that specializes in on-demand GPU computing solutions for artificial intelligence (AI) and machine learning (ML) applications. Based in India, JarvisLabs.ai has rapidly gained recognition in the AI infrastructure space, offering cost-effective and accessible cloud services to a global audience. Through their innovative approach, the company has simplified high-performance GPU computing, empowering organizations of all sizes to access the resources needed for advanced AI development.

1.1.2 Core Business

JarvisLabs.ai's primary offering, GPU as a Service (GaaS), is a one-click GPU cloud service designed to streamline AI development and deployment. The platform provides immediate access to high-performance computing resources while abstracting complex technical infrastructure management. This service enables organizations and individuals to focus on their core AI development work without being burdened by infrastructure setup and maintenance challenges.

1.1.3 Hardware Offerings

JarvisLabs.ai offers a comprehensive portfolio of high-performance GPUs to cater to various computing needs. Their range includes premium NVIDIA GPUs such as the H100, A100, A6000, and RTX5000 series. In addition to GPU solutions, the platform also supports CPU instances for workloads requiring traditional processing power. By incorporating serverless solutions and automatic resource scaling, the platform ensures optimal resource allocation and cost-efficiency, making it adaptable to both small-scale and large-scale AI projects.

1.1.4 Platform features

The platform is designed with user productivity and convenience in mind. At its core, JarvisLabs.ai offers seamless access to pre-configured Python environments and popular AI frameworks such as PyTorch, ComfyUI, and HuggingFace AutoTrain. Integrated with JupyterLab, the platform provides a familiar and powerful development interface for data scientists and researchers. Additionally, automated CUDA configuration eliminates common technical challenges, allowing users to begin their projects with minimal setup time. Secure SSH access further enhances the platform's accessibility and security, enabling users to manage their environments remotely.

1.1.5 Key Differentiators

JarvisLabs.ai has revolutionized the GPU cloud pricing model by significantly reducing costs - often by 3 to 5 times - compared to traditional solutions. Their pay-per-minute billing ensures that users are only charged for the actual usage of computing resources. Coupled with flexible resource scaling, this pricing model provides a cost-effective solution for a wide range of users, from individual developers to large organizations. This unique approach has broadened access to high-performance computing, democratizing AI infrastructure.

1.1.6 Technical Advantages

The technical architecture of JarvisLabs.ai emphasizes automation, efficiency, and ease of use. Key features such as automated CUDA configuration and pre-installed AI libraries remove common setup barriers, allowing users to focus directly on their AI development work. The platform's secure, high-availability infrastructure ensures optimal performance and reliability, making it a preferred choice for organizations that require both stability and scalability in their computing environments.

1.1.7 Primary Users

JarvisLabs.ai serves a diverse range of users, from individual AI researchers and machine learning engineers to teams at small and medium-sized enterprises. Its platform is particularly valuable for deep learning developers and AI content creators who require powerful computing resources without the complexity of physical infrastructure management. Whether for personal projects or collaborative efforts within larger organizations, JarvisLabs.ai provides a scalable solution for AI development.

1.1.8 Strategic Partnerships

JarvisLabs.ai has established strategic partnerships to enhance its service offerings. A key collaboration with Nebius AI has allowed the company to provide access to state-of-the-art GPUs such as the H100. Additionally, the company's recognition by INDIAai, the Indian government's AI portal, underscores its significant contribution to the national AI ecosystem and highlights its role in advancing AI adoption across various sectors.

1.1.9 Online Presence

The company actively engages with the developer community through its **GitHub** presence (https://github.com/jarvislabsai), where it shares resources and technical insights. Its feature on **INDIAai** (https://indiaai.gov.in/) showcases its role in fostering AI innovation and supporting the growth of India's AI industry.

1.1.10 Impact and Innovation

JarvisLabs.ai has established itself as a crucial enabler in the AI ecosystem by democratizing access to high-performance computing resources. Their platform effectively bridges the gap between complex infrastructure management and practical AI development, making advanced GPU computing accessible to organizations without dedicated DevOps teams. This democratization has accelerated AI innovation and research across various sectors.

1.1.11 Future Outlook

As a growing player in the AI infrastructure space, JarvisLabs.ai continues to evolve its offerings to meet the increasing demands of AI development and deployment. The company maintains its focus on providing cost-effective, user-friendly solutions for AI computing needs while adapting to emerging technologies and market requirements. Their commitment to innovation and accessibility positions them well for continued growth in the dynamic AI infrastructure market.

1.2 SYSTEM CONFIGURATION

A brief about the hardware and software infrastructure used in the development of these projects.

1.2.1 Hardware Specification

The hardware environment in which the project is carried out has been detailed.

Processor	:	11th Gen Intel(R) Core(TM) i5-1135G7
Storage	:	477 GB SSD
RAM	:	8.00 GB

1.2.2 Software Specification

The software environment encompasses the following tools, languages, and platforms.

Operating System	:	Windows 10 Home Single Language
OS Version	:	22H2 Build 19045.5011

1.3 TOOLS AND TECHNOLOGIES USED

The following is a brief overview of the tools, languages, frameworks, databases, and libraries used in the development of these projects.

1.3.1 Python 3.11.3

Python 3.11.3 is a high-level, interpreted programming language known for its readability and versatility. Released in 2023, it features performance enhancements that make it 10-60% faster than Python 3.10. Key updates include improved error handling with clearer messages and traceback functionality, better type checking and inference, optimized memory usage, and native support for TOML configuration files. This version is ideal for various applications, including web development, data science, artificial intelligence, machine learning, and automation, due to its comprehensive standard library and extensive third-party packages.

1.3.2 JavaScript

JavaScript is a dynamic programming language essential for modern web development. It enables interactive web applications through client-side scripting and supports server-side development via Node.js. With robust asynchronous programming capabilities, it efficiently handles concurrent operations. JavaScript has a rich ecosystem of libraries and frameworks, supports both object-oriented and functional programming paradigms, and offers seamless data interchange with native JSON support, ensuring consistent performance across different platforms and devices.

1.3.3 Nest.js

Nest.js represents a progressive Node.js framework designed for building efficient, scalable server-side applications. Built with TypeScript at its core, it implements a modular architecture using decorators that enhances code organization and maintainability. The framework provides built-in support for dependency injection, making it easier to manage component dependencies and testing. Nest.js maintains compatibility with most Node.js libraries, while offering comprehensive support for both REST API and GraphQL implementations. Its integrated testing utilities facilitate thorough application testing, and the framework is backed by extensive documentation and an active community that contributes to its continuous evolution.

1.3.4 Next.js

Next.js emerges as a powerful React-based framework engineered for production-grade applications. It excels in providing server-side rendering (SSR) and static site generation (SSG) capabilities, significantly improving application performance and search engine optimization. The framework implements automatic code splitting to optimize page loads, while featuring a built-in routing system that simplifies navigation management. Next.js includes API routes for backend functionality, enabling full-stack development within a single project. Development

is streamlined through features like hot code reloading, image optimization, and font optimization. The framework's zero configuration deployment options make it particularly attractive for teams seeking rapid deployment capabilities.

1.3.5 React.js

React.js is a pioneering JavaScript library focused on building dynamic user interfaces with efficiency and flexibility. Its component-based architecture promotes reusable code and maintainable applications, while the Virtual DOM implementation ensures optimal rendering performance through efficient updates. The library implements a unidirectional data flow that makes application state management more predictable and easier to debug. React's JSX syntax combines JavaScript with HTML-like code, providing an intuitive way to describe UI components. The library benefits from a large ecosystem of third-party components, strong community support, and extensive developer tools that enhance the development experience.

1.3.6 Express

Express is a minimal yet powerful Node.js web application framework that emphasizes simplicity and flexibility. Its robust routing system enables precise handling of HTTP requests, while its middleware support allows for modular application development with easily insertable functionality. The framework excels in static file serving and seamlessly integrates with various template engines for dynamic content generation. Express provides comprehensive error handling mechanisms and database integration capabilities, making it suitable for building both simple and complex web applications. Its HTTP utility methods simplify common web development tasks, making it a preferred choice for developers building RESTful APIs and web applications.

1.3.7 FastAPI

FastAPI stands as a cutting-edge Python web framework specifically designed for building high-performance APIs. Built on Starlette and Pydantic, it combines speed with ease

of use, offering automatic API documentation generation that updates in real-time with code changes. The framework's robust asynchronous support enables efficient handling of concurrent requests, while its type hints and validation ensure reliable API operations. FastAPI's integration with OpenAPI (Swagger) provides interactive API documentation, and its WebSocket support enables real-time communication features. The framework's high-performance characteristics make it particularly suitable for production environments requiring quick response times and efficient resource utilization.

1.3.8 Tailwind CSS

Tailwind CSS revolutionizes frontend development as a utility-first CSS framework that prioritizes flexibility and efficiency. Its comprehensive set of pre-built utility classes enables rapid UI development without leaving HTML, while its responsive design system ensures consistent appearance across different screen sizes. The framework includes built-in dark mode support and offers extensive customization options through its configuration system. Tailwind's JIT (Just-In-Time) compilation generates styles on-demand, significantly reducing final bundle sizes. The integration with PurgeCSS eliminates unused styles in production, while its extensive plugin system allows for easy extension of functionality.

1.3.9 PostgreSQL

PostgreSQL stands as an advanced open-source relational database system that combines robust features with reliable performance. Its ACID compliance ensures data integrity, while its support for complex queries enables sophisticated data operations. The database excels in handling JSON data, making it suitable for applications requiring both structured and semi-structured data storage. PostgreSQL's full-text search capabilities and spatial database features expand its utility across various use cases. Its extensible type system allows for custom data types, while concurrent user support and table partitioning enable efficient scaling. The implementation of multi-version concurrency control (MVCC) ensures

consistent data views during concurrent operations.

1.3.10 MySQL

MySQL has established itself as a leading open-source relational database management system, known for its reliability and extensive feature set. Its ACID compliance ensures transaction reliability, while its replication support enables scalable database architectures. The system's partitioning capabilities allow for efficient handling of large datasets, complemented by stored procedures for complex data operations. MySQL includes robust support for triggers and views, enhancing data manipulation and presentation capabilities. Its unicode support ensures proper handling of international character sets, while SSL security features protect data transmission. The implementation of query caching optimizes performance for frequently accessed data.

1.3.11 Streamlit

Streamlit is an open-source Python library that enables developers to create interactive web applications for data science and machine learning projects with minimal effort. It allows users to build user interfaces directly from Python scripts, leveraging simple commands to display charts, tables, and input widgets. Ideal for rapidly prototyping and sharing data-driven applications, Streamlit enhances collaboration by making it easy to showcase results and insights in a user-friendly format.

1.3.12 Visual Studio Code

Visual Studio Code is a lightweight yet powerful source code editor widely used in modern software development. It features IntelliSense, an intelligent code completion system offering context-aware suggestions across multiple languages. The editor supports robust debugging, allowing developers to step through code and inspect variables in real-time. Its native Git integration streamlines version control directly from the interface, while the extensive marketplace offers numerous extensions for additional functionality. With multi-

language support, an integrated terminal, and custom snippets, VS Code serves as a versatile tool for a variety of development projects.

1.3.13 ComfyUI

ComfyUI is a user-friendly interface built to facilitate various machine learning tasks, especially in the realm of generative models. It allows users to create, manage, and run workflows visually without needing deep programming knowledge. ComfyUI typically integrates with various backends, enabling functionalities such as image generation, data processing, and more. Its intuitive design helps users efficiently utilize GPU resources and manage AI workflows.

1.3.14 GitHub

GitHub is a web-based platform for version control and collaboration, allowing developers to store, manage, and track code using Git. It provides features like repositories, pull requests, and issue tracking, facilitating teamwork on software projects. With a vibrant community and extensive integration options, GitHub is essential for open-source development and project management.

1.3.15 GitHub Desktop

GitHub Desktop serves as a streamlined graphical user interface that simplifies Git operations and GitHub interactions. The application provides a clear visual representation of commit history, making it easier to track project changes and understand code evolution. Its branch management system facilitates smooth creation, switching, and merging of branches, while the pull request handling streamlines code review and collaboration processes. The application simplifies repository cloning, allowing users to easily obtain local copies of remote repositories. Its merge conflict resolution interface provides a visual way to resolve conflicting changes, making it more accessible for developers new to version control. The file history viewing feature enables quick access to previous versions of files, and its seamless integration

with GitHub ensures synchronized remote and local repositories.

1.3.16 MySQL Workbench

MySQL Workbench functions as a unified visual database design and management tool that streamlines database operations. Its SQL development environment provides advanced features for writing and optimizing SQL queries, complete with syntax highlighting and code completion. The tool's data modelling capabilities enable visual creation and modification of database schemas through an intuitive interface. Server administration features allow for comprehensive database server management, including user administration and security configuration. The visual explain plans help optimize query performance by analysing execution paths. Schema migration tools facilitate smooth database updates and version control, while the server monitoring features enable tracking of performance metrics and resource utilization.

1.3.17 DBeaver

DBeaver stands as a versatile universal database management tool that supports a wide range of database systems. Its multi-platform support ensures consistent functionality across different operating systems, while its support for multiple database types makes it a single solution for diverse database environments. The tool's ER diagram functionality enables visual database structure representation and modification. Comprehensive data export/import capabilities facilitate data migration and backup processes. The advanced SQL editor includes features like auto-completion and syntax highlighting, making query writing more efficient. Query execution plans help optimize database performance, while metadata editing capabilities allow for detailed database structure management.

1.3.18 Swagger UI

Swagger UI functions as a sophisticated interactive API documentation interface that transforms OpenAPI specifications into user-friendly documentation. It supports comprehensive OpenAPI/Swagger specifications, enabling detailed API documentation with

interactive features. The interface includes a built-in API testing tool that allows developers to execute requests directly from the documentation. Request and response visualization helps understand API behaviour, while authentication support enables testing of secured endpoints. The tool's API endpoints grouping feature organizes complex APIs into logical sections. Schema visualization makes complex data structures more comprehensible, and the try-it-out functionality enables real-time API testing without additional tools.

1.3.19 Figma

Figma stands as a powerful collaborative interface design tool that has revolutionized the design workflow in modern software development. Its real-time collaboration features enable multiple designers to work simultaneously on the same project, facilitating team coordination and immediate feedback. The component-based design system promotes consistency and reusability across projects, while its vector editing capabilities ensure precise design control. Figma's prototyping features allow designers to create interactive mock-ups that simulate real user experiences. The platform's design system support helps maintain consistency across large-scale projects, while its extensive plugin ecosystem extends functionality for various specialized needs. The design handoff features streamline the transition from design to development by providing detailed specifications and assets.

1.3.20 Pandoc

Pandoc functions as a comprehensive universal document converter that excels in transforming between various document formats. Its support for multiple formats enables conversion between numerous file types, including markup formats, word processor formats, and presentation formats. The tool's custom templates feature allows for consistent document styling across conversions. Advanced metadata handling capabilities enable the preservation and manipulation of document properties during conversion. Built-in syntax highlighting supports code block formatting in technical documentation. The table of contents generation

feature automatically creates navigable document structures, while bibliography support enables academic and technical writing with proper citations. Pandoc's PDF output via LaTeX ensures high-quality typeset documents suitable for professional publication.

1.3.21 MinIO

MinIO operates as a high-performance object storage system designed for cloud-native applications. Its S3 API compatibility ensures seamless integration with existing applications designed for Amazon S3 storage. The system's scalable architecture enables growth from small deployments to large-scale installations while maintaining performance. MinIO's erasure coding provides data protection and high availability, while comprehensive bucket management features enable organized data storage. The built-in version control system tracks object changes over time, and identity management features ensure secure access control. Event notifications enable automated workflows and real-time data processing integrations, making it suitable for modern cloud-native applications.

1.3.22 JoditEditor

JoditEditor serves as a sophisticated WYSIWYG rich text editor that enhances content creation in web applications. Its customizable toolbar allows for tailored editing experiences suited to specific needs, while robust file upload support enables media integration within content. The editor's image handling capabilities include resizing, alignment, and caption features. Advanced table creation and editing tools facilitate complex content organization. The source code editing mode provides direct HTML manipulation when needed, while multiple language support makes it suitable for international applications. The editor's mobile-friendly interface ensures consistent functionality across different devices, making it ideal for modern web applications requiring rich text editing capabilities.

1.3.23 Prisma ORM

Prisma ORM is an open-source database toolkit that streamlines database access for Node.js and TypeScript applications. It provides an intuitive data modelling interface, type-safe queries, and a powerful migration system, enhancing developer productivity and code quality. With support for various databases like PostgreSQL, MySQL, and SQLite, Prisma simplifies the process of interacting with databases while ensuring type safety and ease of use.

1.3.24 Zod

Zod is a TypeScript- first schema validation library designed for building type-safe validation logic in applications. It provides a straightforward API for defining and validating complex data structures, ensuring that inputs conform to specified types and formats. With Zod, developers can easily integrate robust validation into their workflows while leveraging TypeScript's static typing capabilities.

CHAPTER 2
EVENT MANAGEMENT SYSTEM

The Event Management System (EMS) have developed to streamline the management of events at PSG College of Technology through a dedicated web platform. The system addresses the challenging of organizing, recording, and displaying various college events, ranging from major conferences to minor departmental activities. Traditionally, event management can be fragmented, leading to inefficiencies in communication, planning, and tracking. This EMS centralizes the process, enabling administrators and organizers to create, update, and mange events efficiently.

2.1 INTRODUCTION

The Event Management System (EMS) offers functionalities to record event details before they occur and allows the upload of event-related images after the events have concluded. All events are displayed on the frontend page, categorized as upcoming or past, with filter capabilities to allow users to quickly access relevant event information.

This EMS is tailored specifically for PSG College of Technology to improve coordination and streamline event-related processes, fostering better collaboration across various departments and ensuring that all event information is easily accessible and up to date.

2.2 SYSTEM FLOW

The EMS follows a structured workflow that ensures smooth event management from creation to post-event updates. Each step is designed to centralize event data and improve accessibility for both organizers and users.

- **Event Creation:** Organizers input details such as event name, date, time, venue, description and other details through the system.
- **Database Management:** All event information stored in a centralized database before the event, with post-event uploads for images or files.

- **Event Display:** Events are showcased on the website's frontend, categorized into upcoming and past events.
- **Post-Event Updates:** After an event, organizers can upload event images and other relevant materials.
- **Filtering and Viewing:** Users can filter events by criteria such as date and category to quickly find specific event information.

2.3 OBJECTIVES

The main goal of the Event Management System (EMS) are as follows,

- Automating event management tasks to reduce manual effort and improve efficiency.
- Providing a user-friendly interface that allows easy creation and management of event details.
- Enabling file uploads using Multer and secure storage with MinIO for event-related media.
- Facilitating admin control over event data to ensure proper coordination and accuracy.
- Improving accessibility by displaying event information in an organized and easily filterable format.
- Enhancing collaboration between organizers and administrators for seamless event planning.

2.4 SYSTEM DESIGN

The Event Management System (EMS) follows a three-tier architecture, with clear separation between the frontend, backend, and database.

- **UI/UX Design:** The UI was designed using Figma, focusing on user roles such as admin and organizer. The goal was to provide a clear interface for managing event

details, with an intuitive event dashboard, event list filtering, and file uploads and storage.

- **Frontend:** The frontend is started building using React, which communicates with the backend through REST APIs, displaying event data and allowing interaction via a user-friendly interface.
- **Backend:** Developed with Nest.js, it handles business logic, routes requests, authentication logic, and manages event data using PostgreSQL via Prisma ORM. The backend processes event CRUD operations, file storage, and user authentication.
- **File Storage:** MinIO is used as the object storage system for event-related files, which are uploaded using Multer and linked via database records.

2.5 FILE HANDLING AND STORAGE

The files related to EMS are handled in the following way:

- **MinIO:** MinIO (Object Storage) is used to store multimedia content, such as images or documents uploaded post-event. After an event concludes, organizers can upload photos, flyers, or reports, which are stored securely in MinIO's object storage system. It provides a high-performance, scalable object storage solution that integrates well with the EMS, allowing for secure and efficient storage and retrieval of large media files. It uploads and pre-signed URL is generated for the file uploaded which will be later used to fetch the file from cloud storage.
- **Multer:** Multer is used in the backend to handle file uploads. Organizers can upload event-related images and documents, which are then processed and stored in MinIO. It simplifies file handling by efficiently processing multipart form data (used for file uploads), ensuring that uploaded files are correctly handled and passed to the object storage.

2.6 DATABASE DESIGN

The Event Management System (EMS) database is designed to manage users, events, and file storage. Key tables include:

- **Users Table:** Manages user roles (admin, organizer) and authentication.
- **Profile Table:** Stores profile information for each user.
- **Events Table:** Stores core event information.
- **Event Details Table:** Holds extended event-specific details.
- **Media Files Table:** Stores media related to the event.
- **Event Host Table:** Stores hosts of the event.
- **Registration Table:** Handles event registrations.
- **Competitions Table:** Manages information about competitions related to events.
- **Prizes Table:** Stores prize information for competitions.
- **Guest Speakers Table:** Stores guest or speaker details.
- **Sponsors Table:** Stores sponsorship details.
- **Event Files Table:** Handles additional media files like documents, images, or videos.

Database relationships:

- One-to-One:
    - ✓ Events and Event Details
    - ✓ Events and Registration
- One-to-Many:
    - ✓ Users and Profile
    - ✓ Users and Host
    - ✓ Events and Host
    - ✓ Events and Media Files

- ✓ Events and Competitions
- ✓ Events and Guests/Speakers
- ✓ Events and Sponsors
- ✓ Events and Event Files
- ✓ Competitions and Prizes

- Many-to-One:
  - ✓ Event Details and Events (each detail belongs to one event)
  - ✓ Media Files and Events (each media file belong to one event)
  - ✓ Registration and Events (each registration belongs to one event)
  - ✓ Competitions and Events (each competition belongs to one event)
  - ✓ Guests/Speakers and Events (each guest/speaker belongs to one event)
  - ✓ Sponsors and Events (each sponsor belongs to one event)
  - ✓ Event Files and Events (each file belongs to one event)
  - ✓ Prizes and Competitions (each prize belongs to one competition)

This structure clarifies how entities in the database interact with one another.

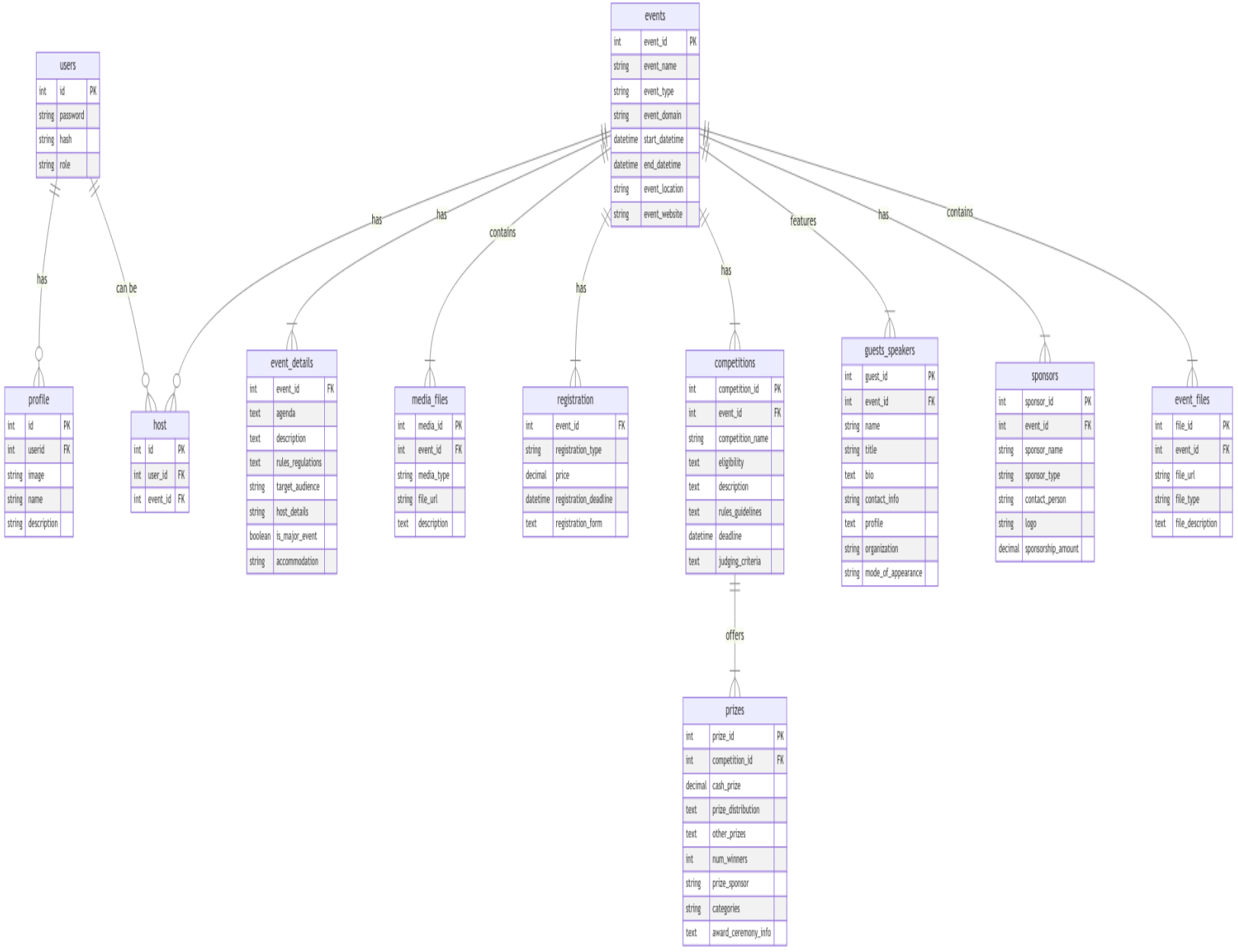


Figure 2.1 EMS

2.7 IMPLEMENTATION

In the implementation of the Event Management System (EMS), the primary contribution involved developing the authentication and authorization mechanisms using NestJS and TypeScript. A **role-based access control (RBAC)** system was implemented, a security approach in which permissions are assigned based on user roles rather than individual user accounts. This mechanism ensures that users are granted access only to resources and actions appropriate for their assigned roles.

In the EMS, two primary roles were defined: **admin** and **organizer**. The admin role was designed to allow the creation of users with the organizer role, setting of user roles, deletion of users if necessary, and listing of all users. The organizer role was given limited access, allowing only the addition of event data. This structured approach enhances overall security by ensuring that only authorized users can interact with the system's sensitive operations.

Several key authentication and authorization routes were developed:

- **/login:** This endpoint handles user login by verifying the user's credentials. If successful, a JWT (JSON Web Token) is issued, which is used to manage session storage and authentication for subsequent requests.
- **/changePassword:** This endpoint allows authenticated users to change their password. It ensures that users can securely update their credentials while logged in.
- **/forgotPassword:** This route handles password recovery. When a user submits their email, the system generates a JWT reset token and sends it via email with a reset link. I used **Nodemailer** to handle the sending of these emails, ensuring smooth and reliable email delivery.

- **/resetPassword:** This endpoint is accessed via the reset link sent in the forgotPassword email. It verifies the JWT token from the link, allowing the user to securely set a new password.

For session management, JWT was used to store session information, providing a secure and scalable solution for user authentication across the EMS. This approach ensures that session data is easily validated, supporting efficient and secure user access control.

After the backend was completed, focus shifted to the frontend, designed in React with user interfaces crafted in Figma. User interfaces and a multi-step event creation form were developed in Figma to streamline the event management process for organizers. The frontend interacts with backend APIs to display upcoming and past events, allowing users to filter and access detailed event information. This cohesive approach delivers a robust and user-friendly Event Management System for PSG College of Technology.

CHAPTER 3
GOVERNING COUNCIL DATA COLLECTION SYSTEM

The Governing Council Data Collection System is designed to streamline the collection and management of institutional reports from various departments and bodies within the college. This system plays a pivotal role in enhancing transparency, accountability, and communication across the institution by enabling structured data collection and efficient report generation for two half-yearly periods. This chapter discusses the architecture, functionality, and implementation of the system, which supports multiple roles and ensures efficient approval workflows.

3.1 INTRODUCTION

The Governing Council Data Collection System is a centralized platform for managing institutional reporting across various college departments and bodies. It facilitates the collection, review, and consolidation of semi-annual reports, streamlining approvals and actions while promoting efficient decision-making and transparency across all college bodies. This system represents a significant step toward digital transformation in institutional management. The scalable architecture and comprehensive feature set ensure that the system can evolve with the institution’s growing needs while maintaining data integrity and security.

3.2 SYSTEM OVERVIEW

The system collects data from different bodies such as Academic, Faculties, Representatives, Autonomous, Placement, NCC, NSS, PED, Personal Assistant, COE, and others. Each user, based on their role, can log in and contribute to their respective reports. Once reports are completed, they are forwarded to the HOD for approval or further action. The system ensures that all reports are submitted in a timely and structured format, which can then be consolidated and downloaded.

3.3 FEATURES

The GC Data Collection System comprises of following features:

- **Role-based Access:** Different user roles have specific access to sections of the system.
- **Data Entry and Progress Tracking:** Users can view, edit, and track the progress of their reports through an intuitive interface.
- **Report Submission and Approval Workflow:** Upon report completion, the HOD can approve or decline the report, with comments. If declined, the user can edit and resubmit.
- **Two Half-yearly Periods:** The system handles reports for two half-year periods, allowing users to edit and submit data for each.
- **Report Generation and Download:** The final consolidated report can be downloaded in a structured .docx format using a Pandoc Server.

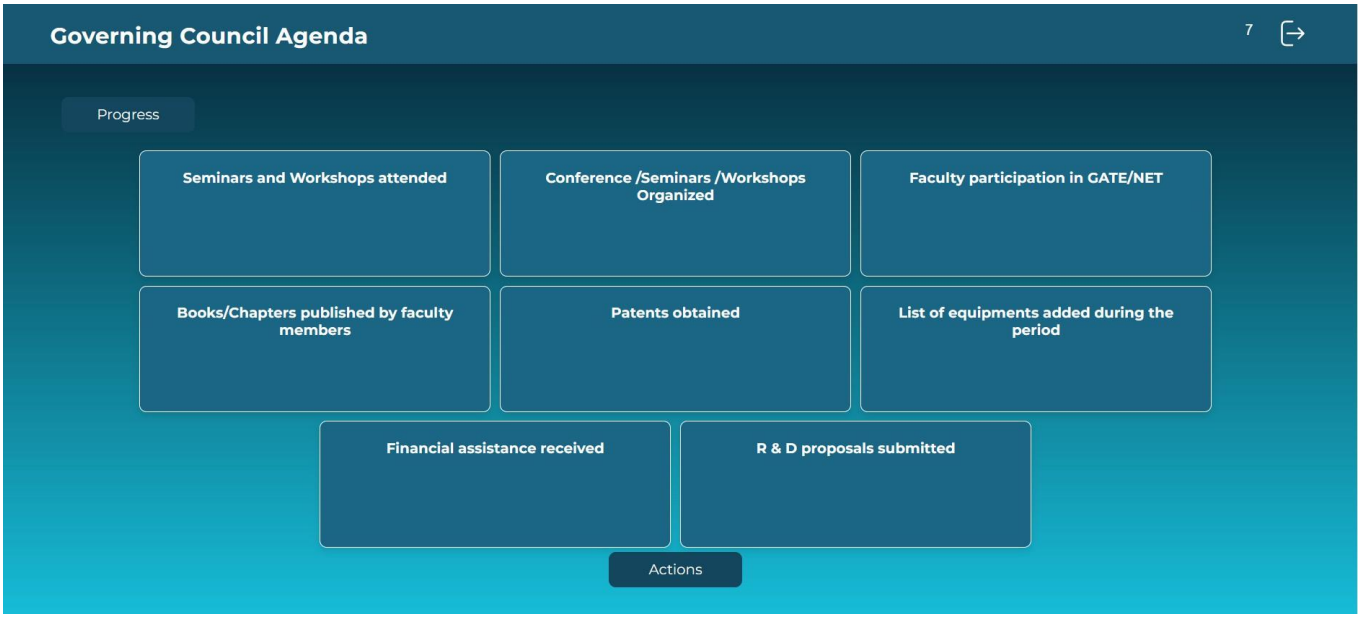


Figure 3.1 Governing Council Agenda

3.4 SYSTEM ARCHITECTURE

The Governing Council Data Collection System uses three tier architecture divided into frontend, backend and database components.

- **Frontend:** The frontend is built using React, ensuring a dynamic and responsive user experience. Each user is redirected to their respective dashboard upon login, where they can mange their reports. The frontend provides interactive cards representing the two half-yearly periods, with options to edit or view reports.
- **Backend:** The backend is powered by Node.js using the Express framework. It handles routing, authentication, and data processing. The backend validates all inputs using the Zod library and securely manages file uploads through Multer, which are stored in MinIO for object storage.
- **Database:** The system uses PostgreSQL as its database, managed with Prisma ORM for efficient querying and data management. The database tracks user roles, report progress, report content, and workflow status (whether a report is pending approval or has been approved).

3.5 FILE HANDLING AND STORAGE

The report and its related files are handled and stored in the following way,

- **Multer and MinIO:** The system uses Multer for handling file uploads, which are stored in MinIO as an object storage solution. This ensures that all media files uploaded within the reports are securely stored and easily retrievable.
- **Pandoc Server:** The Pandoc Server is used for .docx generation. Once reports are completed and approved, they are downloaded using Pandoc. The Pandoc server converts HTML content from JoditEditor – WYSIWYG editor that allows users to see and edit content in a form that appears as it would when displayed on an interface,

webpage, or printed document into .docx format, ensuring consistency in report formatting.

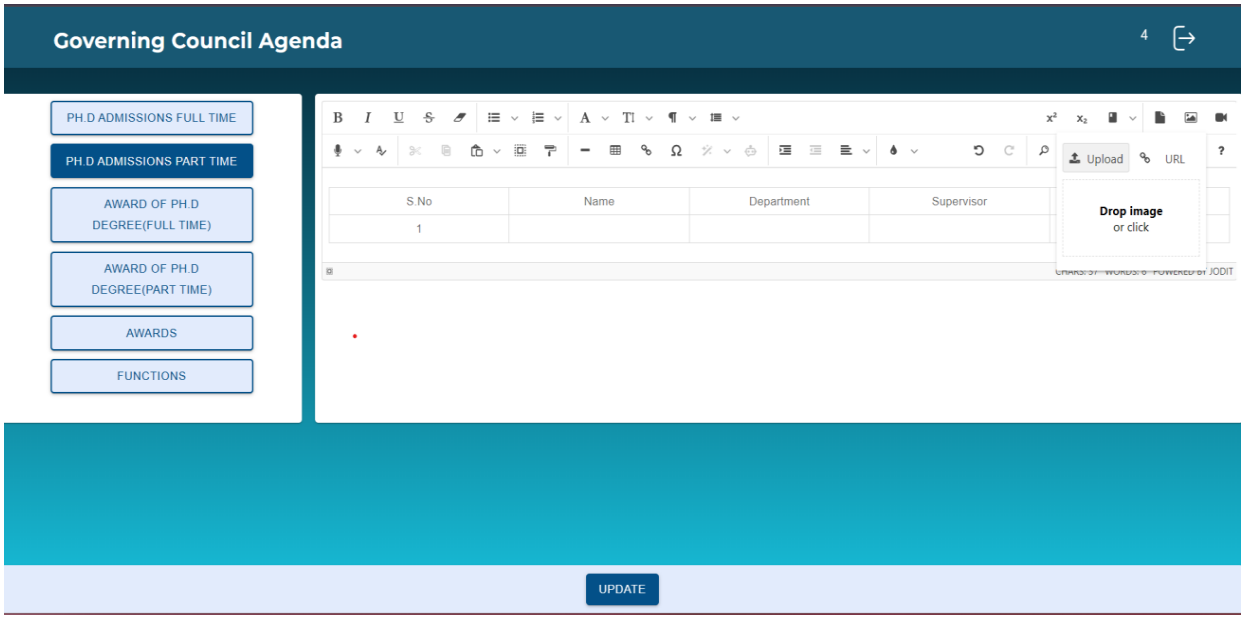


Figure 3.2 JoditEditor

3.6 DATABASE DESIGN

The database design for the system is structured to effectively manage and organize data related to departments, users, reports, and their associated elements. The database employs a relational model to maintain structured data integrity and facilitate efficient querying. Key entities include:

- **Department Table:** Represents various departments within the organization, each responsible for specific functions and activities.
- **User Table:** Represents individuals working within departments who have access to the system.
- **Report Table:** Represents reports generated by departments, detailing their activities and progress.
- **Object Table:** Represents associated objects or attachments linked to reports.
- **ProgressStep Table:** Represents individual steps in the progress tracking of a report.

- **_ReportToUser Table:** Join table that establishes a many-to-many relationships between reports and users.

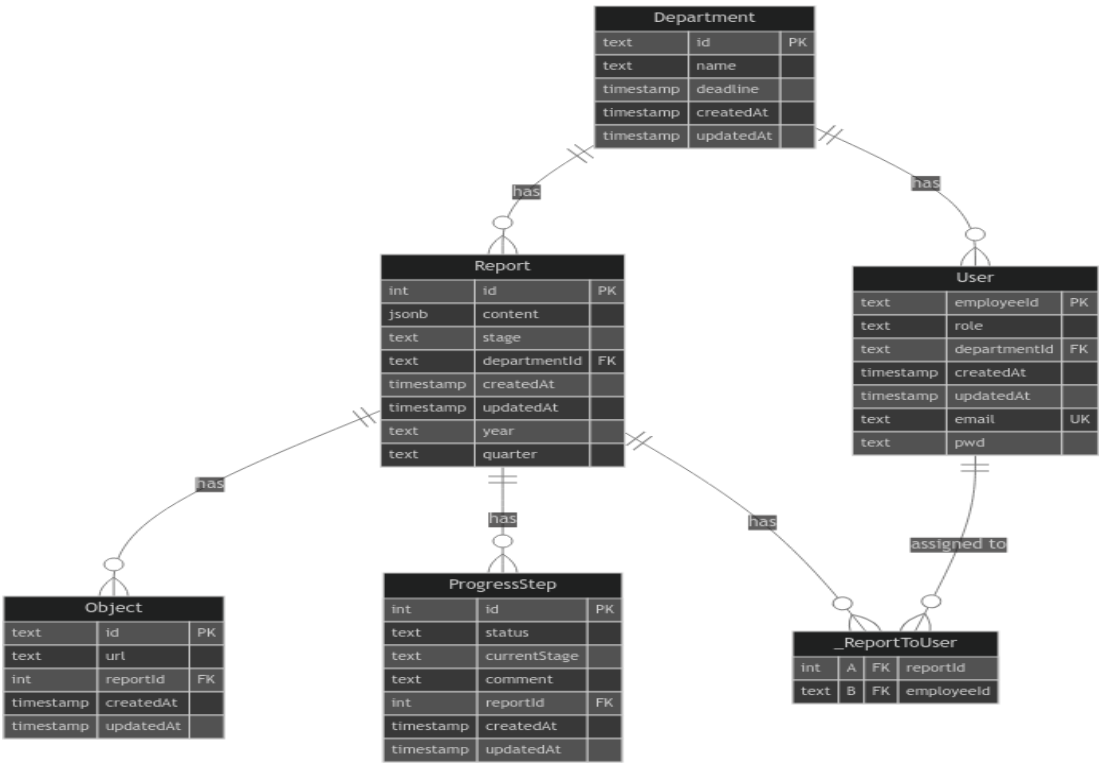


Figure 3.3 GC Data Collection System Schema

Database relationships:

- **One-to-Many:**
    - ✓ Department to Report – A department can have multiple reports.
    - ✓ Department to User – A department can have multiple users.
    - ✓ Report to Object – A report can have multiple associated objects.
    - ✓ Report to ProgressStep – A report can have multiple progress steps.

- **Many-to-Many:**
    - ✓ Report to User (_ReportToUser Table) – A report can be assigned to multiple users, and a user can be assigned to multiple reports through the _ReportToUser join table.

This structure clarifies how entities in the database interact with one another.

3.7 USER WORKFLOW

The overall system workflow will be described here,

- **Login and Role-based Redirection:** Upon login, users are redirected to their respective roles' pages. Each role sees fields relevant to them and can contribute to the report for each half-year.

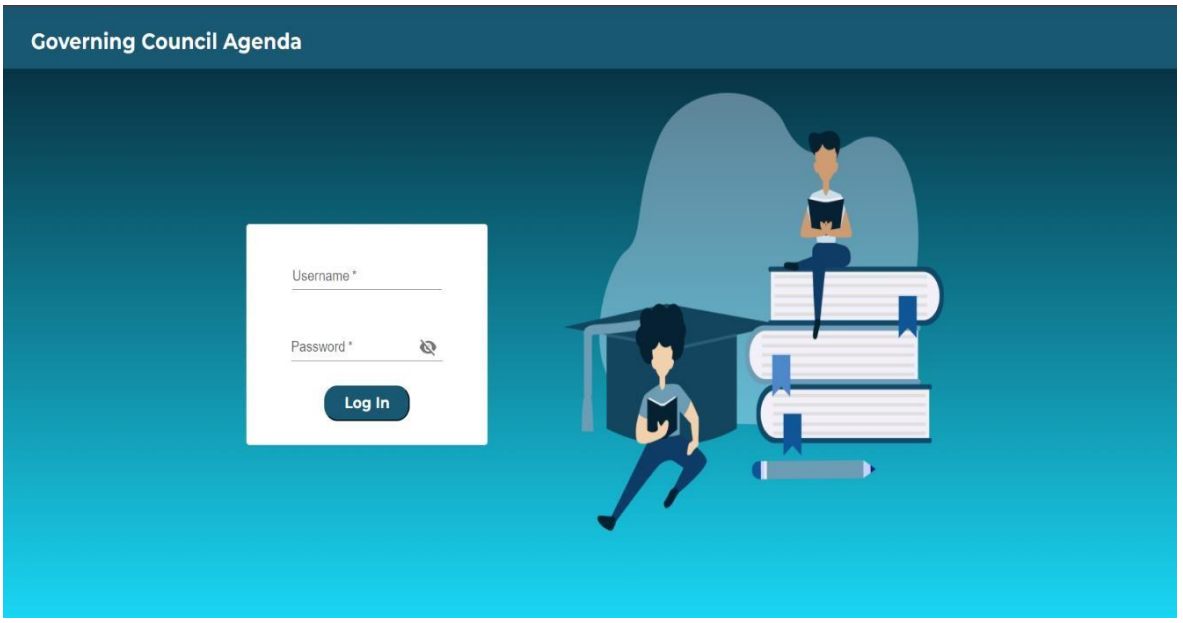


Figure 3.4 Authentication Page

- **Progress Tracking and Report Entry:** Users can see the progress of their reports. If the half-year period is active, users can edit or delete their reports. The JoditEditor interface allows them to add content and updates.

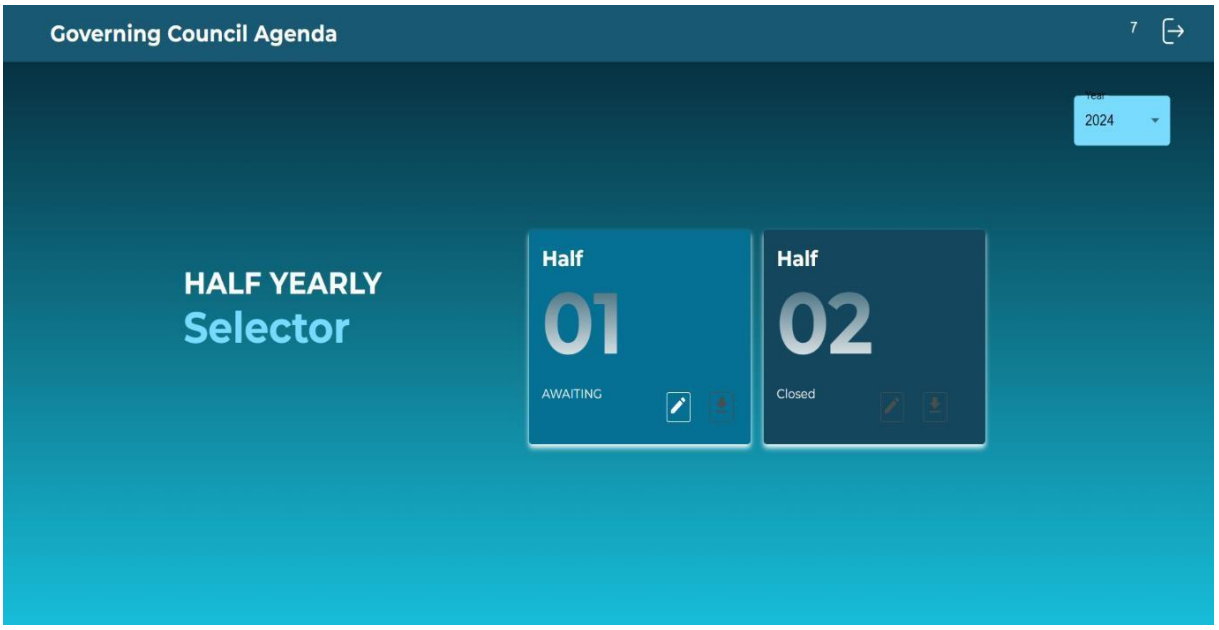


Figure 3.5 Biannual Cards

- **Submission and Approval:** After all required fields are filled, users submit their reports. The status changes to "Action from HOD," and the editing option is disabled. The HOD can then review, comment, and approve or decline the report. Declined reports can be edited and resubmitted.

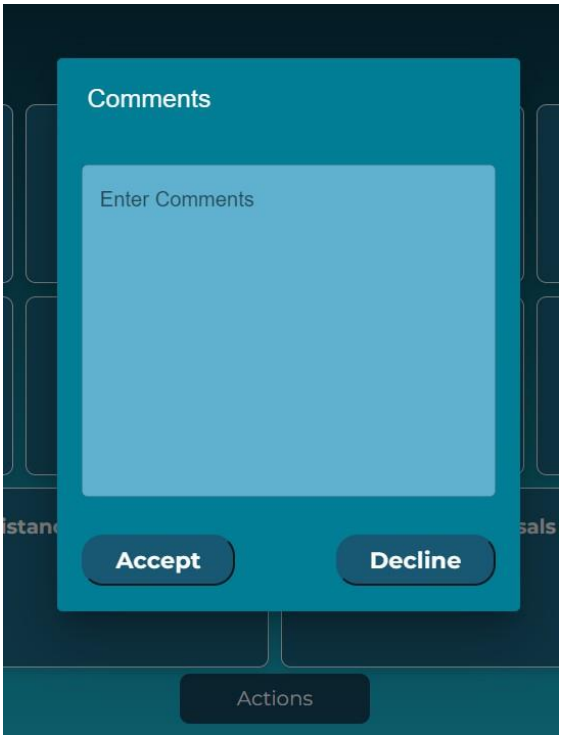


Figure 3.6 Action Page

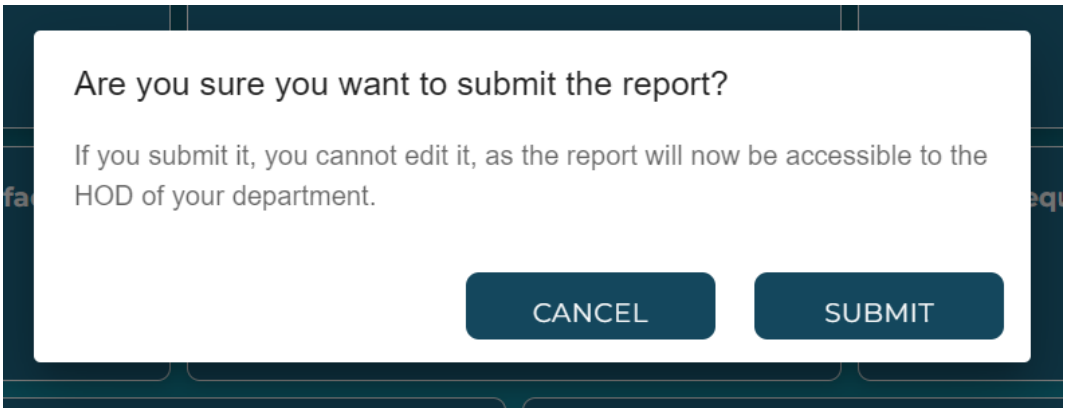


Figure 3.7 Confirmation Page

- **Report Consolidation and Download:** At the end of the half-year period, the system consolidates all submitted reports and allows for the download of a zip file containing the reports in .docx format. The Pandoc server ensures that each report is formatted uniformly and includes all media and data entered by users.

This implementation exemplifies how the system leverages modern web technologies to ensure a structured, transparent, and efficient reporting process from data entry to final report generation.

CHAPTER 4
PRODUCT OPTIMIZATION AND ENHANCEMENT

The project work at JarvisLabs.ai encompassed several phases, including the development of AI workflows with ComfyUI and the optimization of GPU usage through the creation of the product ComfyUI Deploy. Features such as AnimateDiff and ControlNet were explored for creative content generation. Additionally, data analysis was conducted, dashboards were created, and improvements were made to the company's website by implementing API endpoints and enhancing the user experience with the addition of invoice generation and a File System UI.

4.1 INTRODUCTION

During the internship period at JarvisLabs.ai, exposure was provided to various innovative tools and technologies aimed at optimizing GPU resource usage and facilitating creative AI workflows. The main objective of the company, providing GPU as a Service (GaaS) on a cloud platform, was central to the project work, which introduced various aspects of AI development, from image generation to cloud infrastructure optimization.

4.2 COMPANY PRODUCT

JarvisLabs.ai is a cloud platform offering high-performance GPU instances (A5000, A6000, RTX series) and pre-configured AI/ML environments with frameworks like PyTorch and FastAI. It simplifies infrastructure management, enabling developers and researchers to train models and run experiments without complex setups. With a pay-as-you-go pricing model, it's accessible to both individual developers and teams.

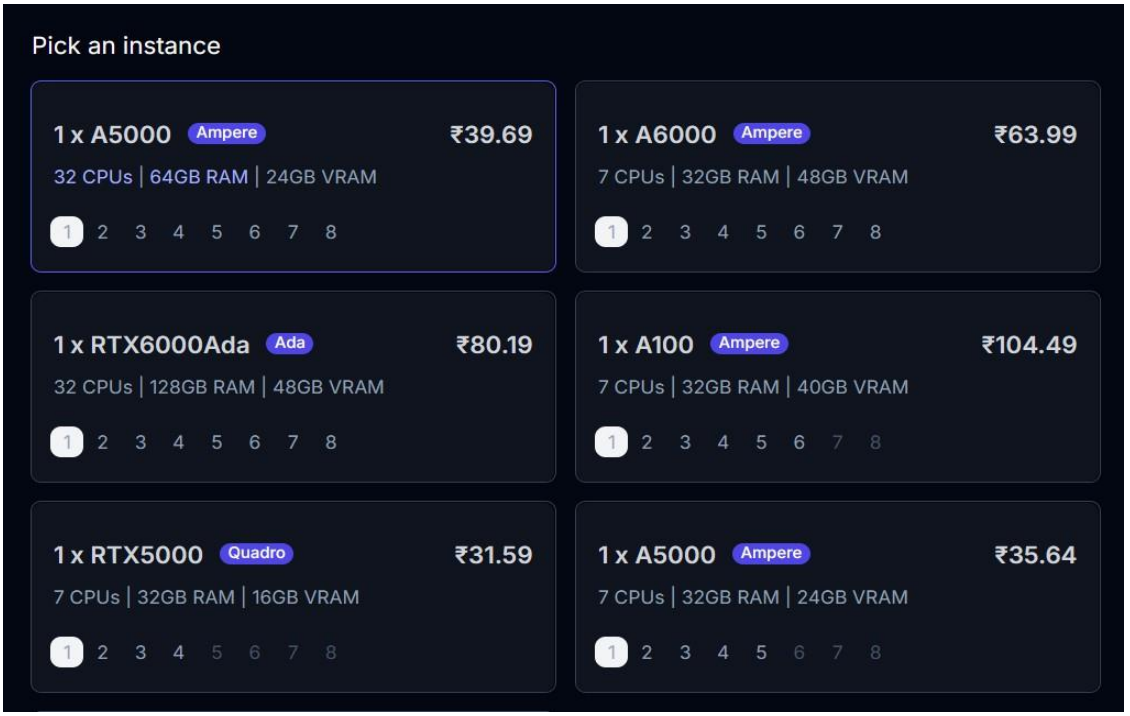


Figure 4.1 GPU Instances

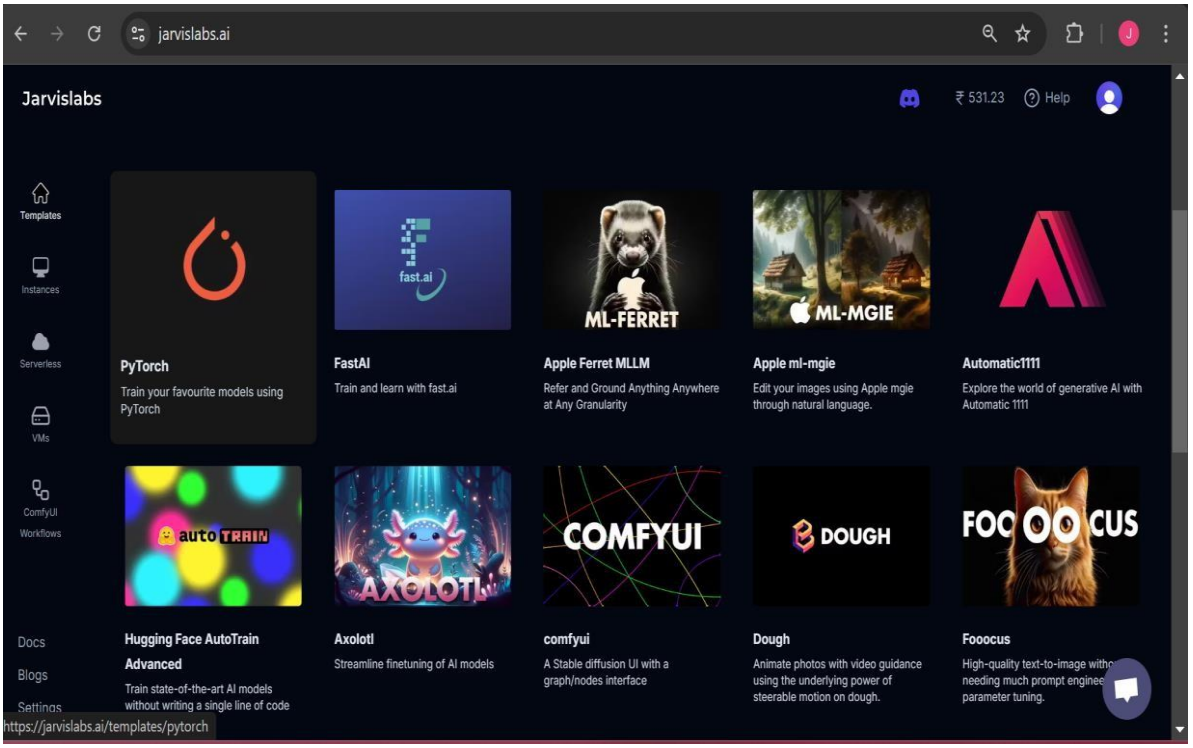


Figure 4.2 Frameworks

4.3 COMFYUI AND AI IMAGE GENERATION

The journey began with an exploration of ComfyUI, a sophisticated stable diffusion tool for AI-driven image and video generation. ComfyUI was examined for its capabilities in simplifying the use of stable diffusion models, such as SDXL, SD1.5, and SD2.x, for text-to-image or video generation. Through this interface, users are enabled to connect nodes in a drag-and-drop manner, facilitating the creation of complex image generation pipelines. Its distinctive feature—a node-based graphical interface—allows for the construction of intricate image generation workflows.

Familiarization with the platform was achieved, and several features were explored, including AnimateDiff, used to create smooth animations from image sequences, and ControlNet, which enables precise control over image generation using reference points. These explorations resulted in creative outputs, such as video-to-video generation. Additionally, knowledge was gained about Lora models, which are trained on top of existing models like SD1.5 or SDXL, and various Lora models available on platforms like Civitai, Hugging Face, and GitHub were experimented with.

Figure 4.3 Workflow Outputs

4.4 CUSTOM NODES

Installation of custom nodes, which are not available by default but serve specific purposes, was undertaken. These nodes were installed using ComfyUI Manager or through terminal commands from platforms like GitHub, Hugging Face, and Civitai. Skills were further honed by participating in a company hackathon focused on creative content generation. Contributions included the development of workflows such as video creation based on prompts and image-to-video workflows. Additionally, issues related to custom nodes and workflows were troubleshot and resolved.

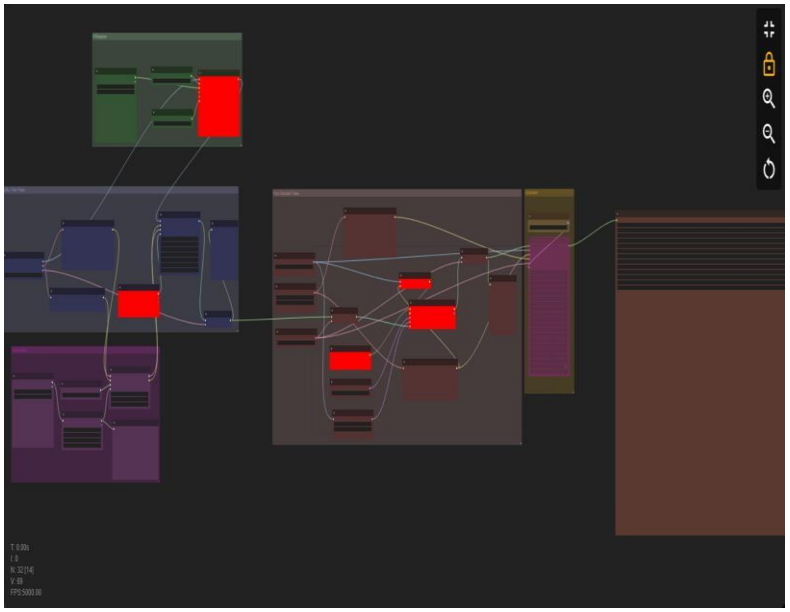


Figure 4.4 Custom Nodes

Figure 4.5 ComfyUI Manager

4.5 PRODUCT DEVELOPMENT: COMFYUI DEPLOY

As part of a team, contributions were made to the development of ComfyUI Deploy, a product aimed at reducing GPU consumption. Traditionally, GPUs were allocated and charged even during workflow setup. The goal was to modify the existing code so that GPUs would be utilized only when necessary—specifically, when the “Run” button was clicked. This was achieved by integrating the ComfyUI interface into an iframe using Next.js and establishing WebSocket connections. This significant improvement helped reduce GPU consumption for users while maintaining a smooth workflow experience.

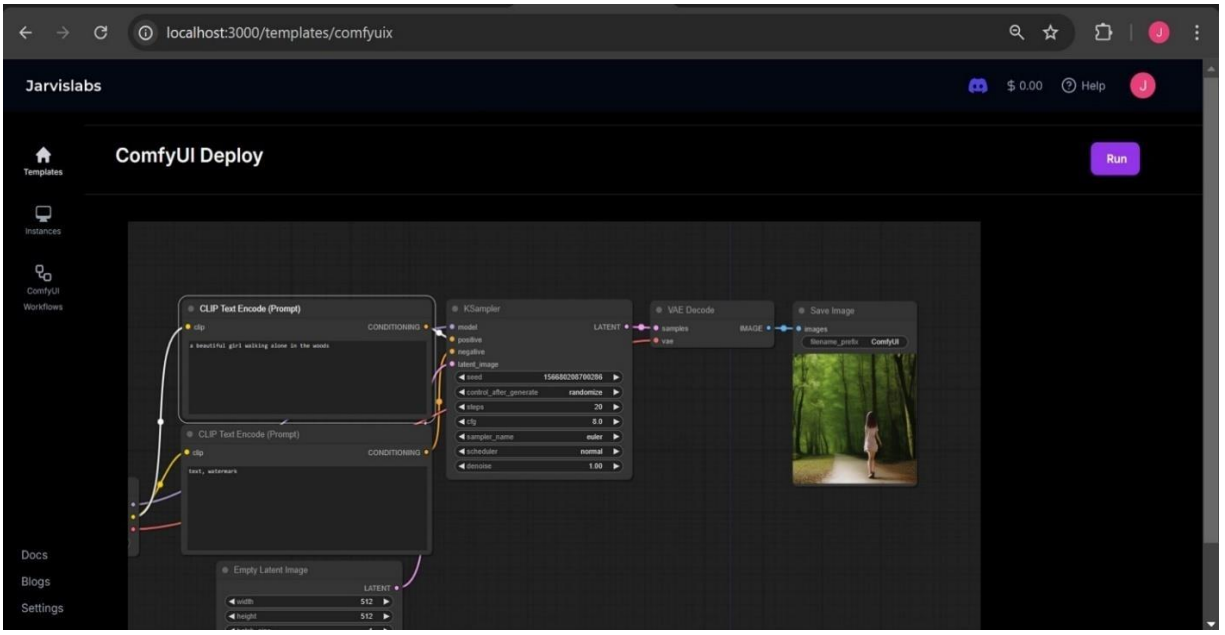


Figure 4.6 ComfyUI Deploy

4.6 DATA ANALYSIS AND VISUALIZATION

The skill set was expanded by working on data analysis and visualization tasks using the Streamlit tool. Initially, dummy data was used, but later, the focus shifted to analyzing real company data, which included user transactions, GPU usage, and more. Detailed dashboards were created to visualize trends across different time periods (daily, weekly, monthly, quarterly, yearly), providing the company with valuable insights.

Based on GPU Utilization

- **GPU Utilization:** Visualizes the total GPU usage in hours across different time periods, helping track resource consumption.
- **GPU Allocation:** Shows the allocation of GPU resources across users or tasks, providing insights into how resources are distributed within the platform.
- **Most Used GPU Type:** Displays the most commonly used GPU type, helping understand user preferences and optimize resource offerings.

Based on Users

- **Number of New Users:** Tracks the total number of new users registered over a specified period, providing insights into user growth and platform adoption.
- **Number of New Users Recharged:** Monitors the count of new users who have made their first recharge, indicating initial engagement and the effectiveness of onboarding strategies.
- **Active Members Over Time:** Visualizes the trend of active members over time, highlighting user retention and engagement levels on the platform.
- **Churn Rate:** Measures the percentage of users who stop using the service over a given time frame, helping to assess user satisfaction and identify potential areas for improvement.

Based on Recharge

- **Total Amount Recharged:** Represents the cumulative amount of money recharged by all users over a specified period, providing insights into overall platform revenue and user investment.
- **Total Amount Recharged by new Users:** Tracks the total recharge amount contributed by new users, indicating their initial commitment and engagement with the platform shortly after registration.

- **Total Amount Recharged by existing Users:** Measures the total recharge amount from existing users, reflecting ongoing user loyalty and continued usage of the service.
- **Number of new users who Recharged within 1 month of creating their account:** Counts the number of new users who completed a recharge within the first month of account creation, highlighting the effectiveness of the onboarding process and user conversion strategies.

4.7 DEVELOPMENT IN WEBSITE

Responsibilities then shifted toward the frontend and backend development of the company's website, utilizing Next.js and FastAPI. New features such as invoice generation and downloading, as well as the user interface for the file system, were developed.

In the invoice generation feature, the jsPDF library was utilized to create PDF documents efficiently. For Indian customers, the invoice defaults to applying GST, ensuring compliance with local tax regulations. For foreign customers, the applicable tax rate is specified, allowing for accurate billing based on their location.

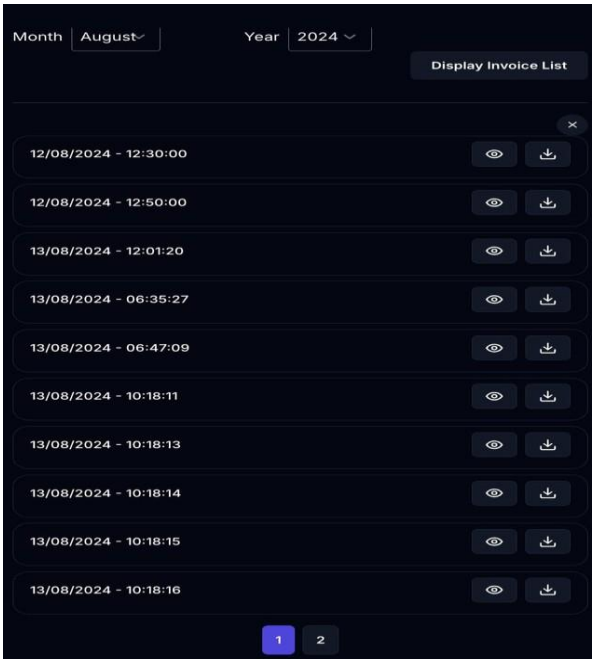


Figure 4.7 Invoice Display

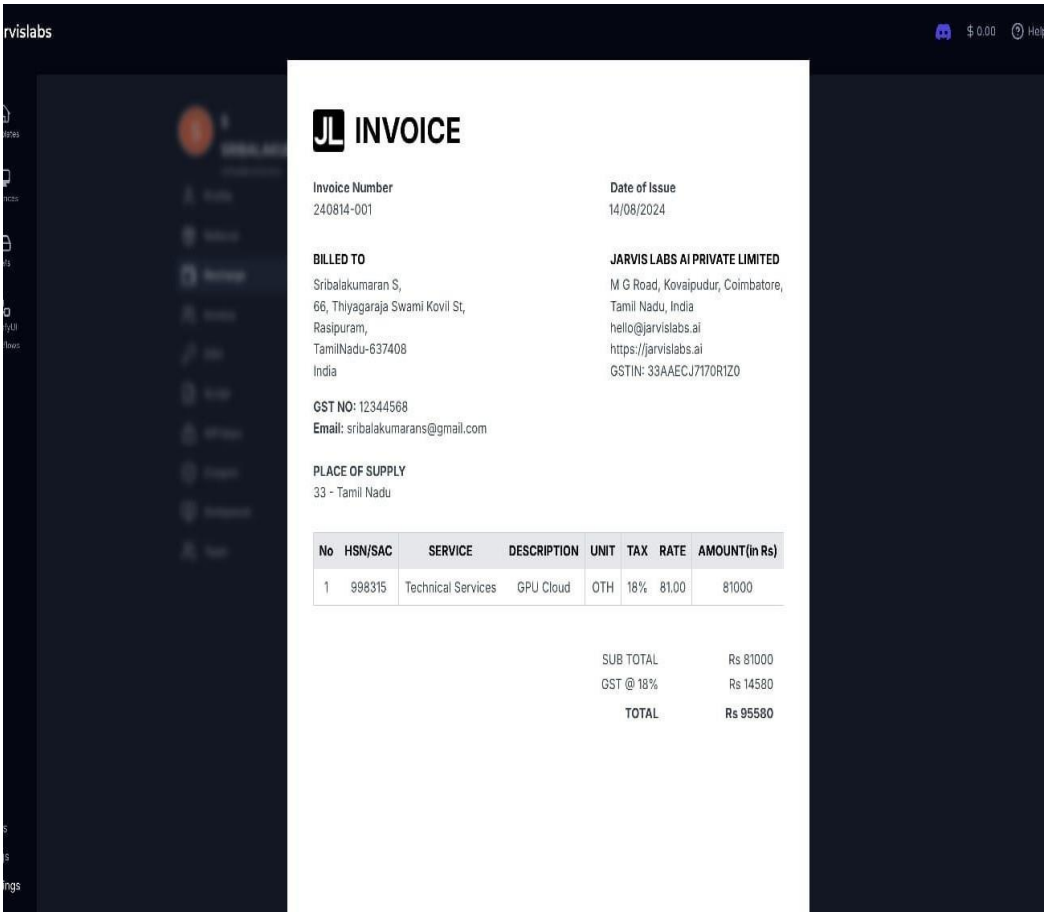


Figure 4.8 Invoice View

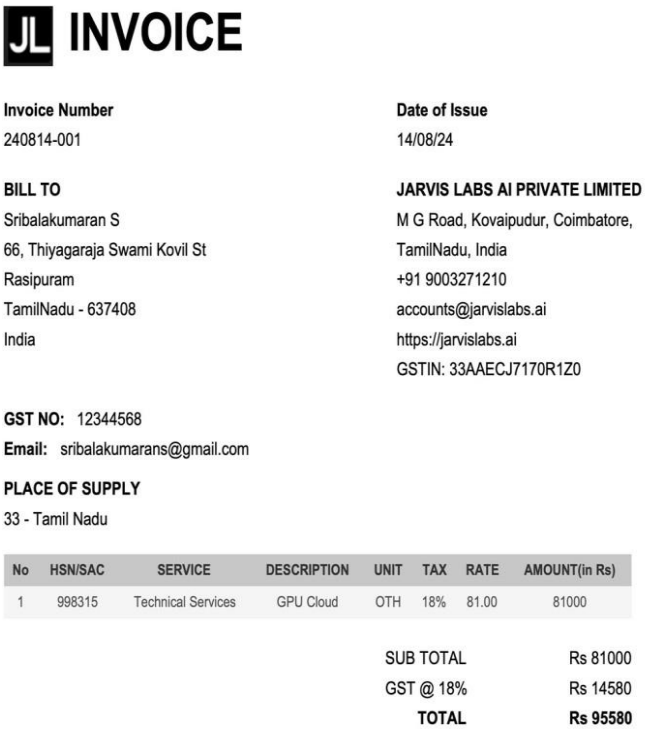


Figure 4.9 Invoice PDF

A file system storage is a service that enables users to store, manage, and access data specifically optimized for GPU workloads over the internet. It offers features like high-speed data retrieval, scalability for large datasets, and support for collaboration on AI/ML projects, allowing users to efficiently work with massive files across multiple devices and locations. Additionally, GPU cloud storage typically includes robust data redundancy and security measures to ensure data integrity and protection against unauthorized access.

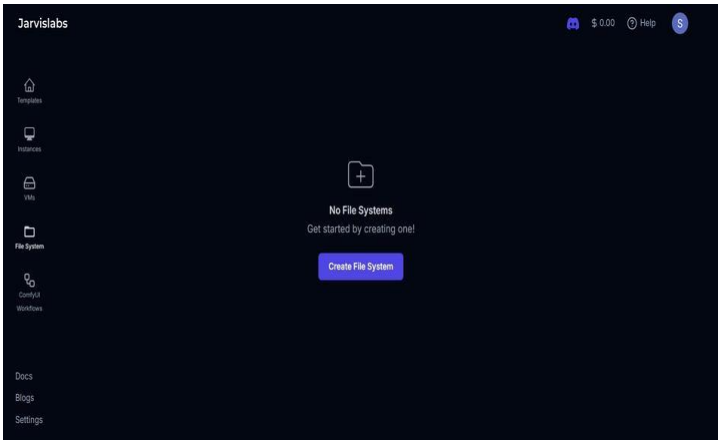


Figure 4.10 File System (i)

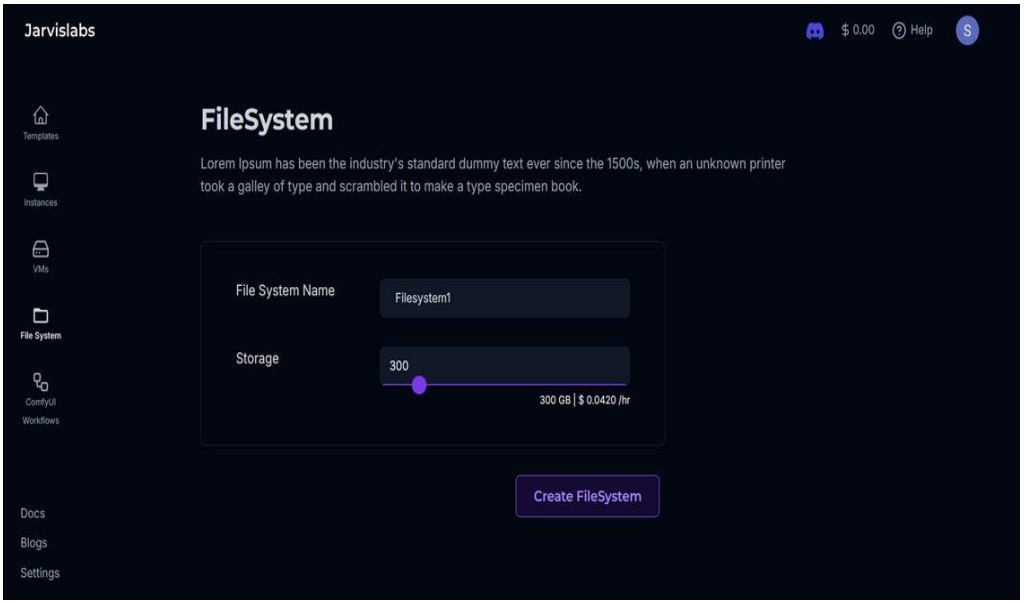


Figure 4.11 File System (ii)

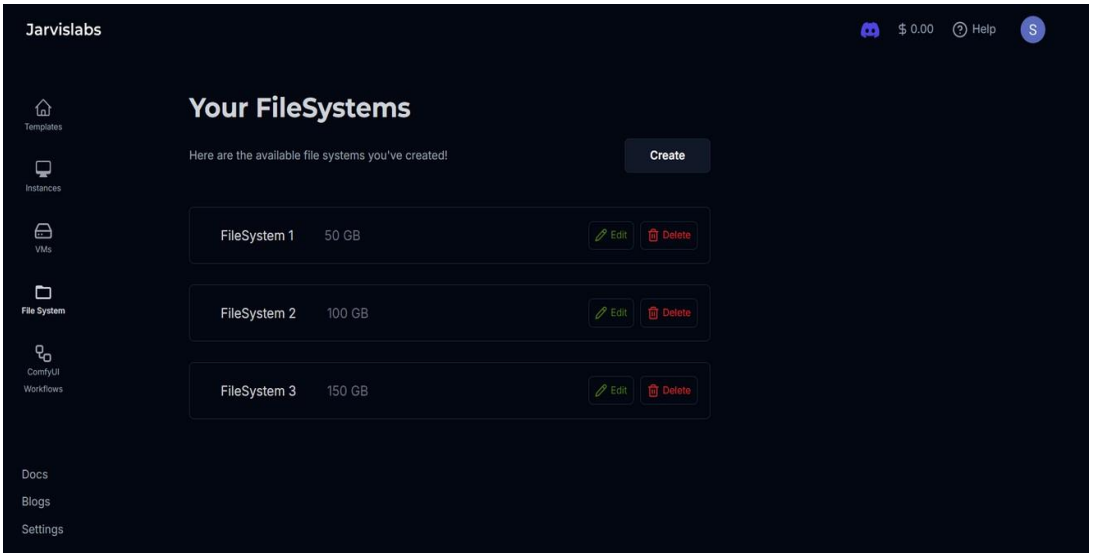


Figure 4.12 File System (iii)

CHAPTER 5
CONCLUSION

In conclusion, a unique opportunity was provided by my internship to contribute to two distinguished organizations with distinct goals.

At PSG College of Technology, contributions were made to two key systems: the Event Management System (EMS) and the Governing Council Data Collection System. The EMS streamlined event organization across departments, while the data collection system automated biannual reporting, improving transparency and enabling data-driven decisions.

At JarvisLabs.ai, a cloud computing platform specializing in GPU services for AI/ML applications, innovative ComfyUI workflows for AI art generation using stable diffusion models were developed, optimizing GPU resource utilization and improving output quality through ComfyUI Deploy. Additionally, an invoice feature was added to the company's website, and the file system UI was developed independently.

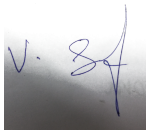
Overall, these projects reflect a commitment to driving resource optimization, enhancing workflow efficiencies, and advancing digital transformation initiatives within both organizations. The skills and knowledge acquired during this internship will undoubtedly serve as a strong foundation for future endeavors in technology and development.

BIBLIOGRAPHY

- https://min.io/docs/minio/kubernetes/upstream/index.html?ref=docs-redirect
- https://medium.com/@ggluopeihai/nestjs-uploading-pictures-8f25f84ad31e
- https://pandoc.org/
- https://zod.dev/
- https://docs.nestjs.com/
- https://developers.google.com/community/gdsc
- https://xdsoft.net/jodit/
- https://gc.psgtech.ac.in/
- https://jarvislabs.ai/
- https://github.com/comfyanonymous/ComfyUI
- https://stability.ai/
- https://www.thinkdiffusion.com/
- https://huggingface.co/
- https://civitai.com/
- https://stable-diffusion-art.com/

Audit Trail

DigiSigner Document ID: 412d6cbd-fe00-4aef-abf6-f6e310b99526

Signer	Signature
Email: kmn.amcs@psgtech.ac.in IP Address: 14.139.180.67	
Email: hod.amcs@psgtech.ac.in IP Address: 14.139.180.67	
Email: kmn.amcs@psgtech.ac.in IP Address: 14.139.180.67	
Email: murugan.senthil@mahindra.com IP Address: 171.79.54.96	

Event	User	Time	IP Address
Upload document	kmn.amcs@psgtech.ac.in	11/12/24 1:53:43 AM EST	14.139.180.67
Open document	kmn.amcs@psgtech.ac.in	11/12/24 1:53:46 AM EST	14.139.180.67
Sign document	kmn.amcs@psgtech.ac.in	11/12/24 1:54:10 AM EST	14.139.180.67
Close document	kmn.amcs@psgtech.ac.in	11/12/24 1:54:10 AM EST	14.139.180.67
Send for signing	kmn.amcs@psgtech.ac.in	11/12/24 1:54:26 AM EST	14.139.180.67
Open document	hod.amcs@psgtech.ac.in	11/12/24 3:55:48 AM EST	14.139.180.67
Sign document	hod.amcs@psgtech.ac.in	11/12/24 3:56:02 AM EST	14.139.180.67
Close document	hod.amcs@psgtech.ac.in	11/12/24 3:56:02 AM EST	14.139.180.67
Open document	kmn.amcs@psgtech.ac.in	11/16/24 12:48:24 AM EST	14.139.180.67
Sign document	kmn.amcs@psgtech.ac.in	11/16/24 12:48:54 AM EST	14.139.180.67
Close document	kmn.amcs@psgtech.ac.in	11/16/24 12:48:54 AM EST	14.139.180.67
Send for signing	kmn.amcs@psgtech.ac.in	11/16/24 12:49:01 AM EST	14.139.180.67
Resend for signing	kmn.amcs@psgtech.ac.in	11/19/24 12:49:48 AM EST	
Resend for signing	kmn.amcs@psgtech.ac.in	11/23/24 12:49:22 AM EST	
Open document	murugan.senthil@mahindra.com	11/24/24 10:53:47 PM EST	171.79.54.96
Open document	murugan.senthil@mahindra.com	11/24/24 10:54:02 PM EST	20.235.158.127
Sign document	murugan.senthil@mahindra.com	11/24/24 10:59:08 PM EST	171.79.54.96
Close document	murugan.senthil@mahindra.com	11/24/24 10:59:08 PM EST	171.79.54.96