TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Effective management of departmental events is vital for the success and growth of any academic institution. In universities, where both students and faculty members engage in numerous academic and extracurricular activities, it is essential to have a structured and efficient system to manage, document, and communicate these events. However, university departments often face challenges in coordinating such activities, which can lead to inefficiencies, communication gaps, and missed opportunities for growth and recognition.

This Department Event Management System is designed to address these challenges by providing a comprehensive platform that streamlines the post-event processes, ensuring that all stakeholders—students, faculty, and administrators—are well-informed and organized. The system serves as a centralized hub for managing event-related activities, including the submission of event details, documentation of participation.

This system not only enhances operational efficiency but also fosters a culture of recognition and documentation within university departments. By enabling students and faculty to upload certificates and photos from the events they participate in, the system helps maintain a documented history of involvement, which can be valuable for academic and career advancement. Administrators can easily filter and download these records based on specific categories, such as workshops, conferences, and hackathons, ensuring that every achievement is accurately recorded and acknowledged.

In summary, Vitastha aims to transform how university departments manage and document their events, ultimately contributing to a more organized, transparent, and growth-oriented academic environment.

CHAPTER 2
OBJECTIVES

1. To access a centralized platform for all event information.
2. To reduce time spent on communicating event details to students and faculty.
3. To provide instant data availability for the admin.
4. To provide filtering option to categorize the events by the admin.
5. To provide downloading of event certificates and photos to the admin.
6. To provide remote access to everyone.

CHAPTER 3
LITERATURE SURVEY

In paper [1], EVENT MANAGEMENT HUB

Abstract: Event management is a rapidly growing field that lacks standardization. Although often viewed as a subset of project management, event management has its unique concepts and challenges, requiring the development of specialized methodologies and tools. To address issues such as record-keeping, event delays, and communication gaps between clients and service providers, we developed an event management software to assist service providers in organizing and supervising events. The system allows registered users to log in and new users to sign up. This web application provides the essential features for various event types, such as parties, weddings, and corporate events, enabling users to select their specific requirements.

Introduction: The "Event Management Hub" web application was developed to address the challenges of managing events manually. This system is designed to meet the needs of clients, enabling them to execute their events smoothly and efficiently. The software is user-friendly, requiring no specialized knowledge from the user. Event management is closely related to project management, particularly in the planning and development of both large-scale and small-scale events, such as weddings, birthday parties, and corporate events. Event planning involves the preparation and coordination of an event, including budgeting, scheduling, venue selection, obtaining permits, arranging speakers or performers, and organizing decor, catering, and other logistical aspects.

In paper [2], EVENT MANAGEMENT SYSTEM

Abstract: This paper is aimed at developing an application for event management system. The event management system is an online event management system software project that serves the functionality of an event manager. The system allow registered user login and new user are allowed to register on the application. The system helps in the management of events, users and the aspects related to them. This proposed to be a web application. User needs to Login at the initial phase, set his/her profile details including location, choices, email-id, etc. User can modify or
change his/her profile at any stage. All the data is logged in the database. The data is then sent to the administrator and they may interact with the user as per his requirement.

Introduction: Event management is the application to manage and development of festivals, events and conferences. Proposed work Involves study of identifying the target of budget, cost, and analysis. This is an online event management system, software project that serves the functionality of an event manager. The project provides most of the basic functionality required for an event. It allows the user to select from list of event types. Event Management System is very helpful for events.

This application being as a platform to know the events, to apply for the events. Event organizer is an application under project management for managing festivals or social events like gathering, colleges, events, conferences etc. by this application user can register the students, after registering, user can login, after login, event details including name, contact, address, venue of the event, date, event conducting time, cost of events etc.

In paper [3], Evolving Trends in College Event Management: A Comprehensive Survey and Implementation Analysis

Abstract: The landscape of college event management has undergone a significant evolution with the
advent of modern technological solutions. This survey paper delves deep into the intricacies of event
management within academic institutions, with a focused exploration on the transition from traditional methods to digital platforms. Moreover, the implementation analysis meticulously examines the integration of NextJs, ReactJs, and MongoDB in developing a robust College Event Management System (CEMS). By amalgamating user insights, case studies, and in-depth analysis, this paper illuminates the transformative potential of digital event management tools in enhancing efficiency, communication, and collaboration within college environments. Through the lens of real-world implementation, the findings underscore the pivotal role of modernized event management systems in fostering student engagement and optimizing event coordination processes, thus contributing to the enhancement of the overall college experience.

Introduction: College life is characterized by a dynamic tapestry of events, ranging from academic conferences to cultural festivals, which contribute significantly to the vibrancy of campus communities. However, the orchestration of these events efficiently poses significant challenges to academic institutions. Traditional approaches often entail cumbersome manual processes, leading to inefficiencies and communication gaps. Recognizing the imperative for innovation, academic institutions are increasingly turning to digital event management systems to streamline operations and enhance the overall event experience. This section provides an extensive overview of the transition from conventional event management approaches to modern digital solutions, setting a robust foundation for a comprehensive survey and implementation analysis. In recent years, the paradigm of college event management has shifted significantly, driven by the pervasive influence of technology. Traditional methods, reliant on manual processes and paper-based communication, have proven to be increasingly inadequate in meeting the diverse needs of modern campus communities. The advent of digital event management systems represents a seminal evolution in the field, offering institutions a comprehensive solution to streamline event planning, execution, and communication processes. By
leveraging cutting-edge technologies such as NextJs, ReactJs, and MongoDB, colleges can create intuitive and efficient participant experience. This transition towards digitalization not only improves operational efficiency but also fosters a culture of innovation and collaboration within academic institutions.

CHAPTER 4

PROBLEM STATEMENT

To design and develop a web-based application for Department Event Management System which makes it easy to plan academic events like workshops, technical events, extra curriculum activities for students and conferences, workshops for faculty.

CHAPTER 5
METHODOLOGY

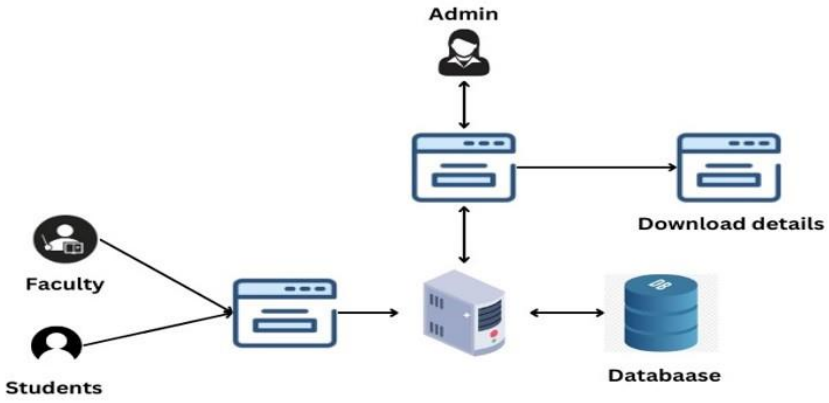

Figure 5.1 Block diagram representing methodology

5.1 User Interaction With The System

The system facilitates the interaction between different user roles (Faculty, Students, and Admin) and a centralized database.

5.1.1 Faculty and Students Interface

Input Data: Faculty and Students access the system through a web interface. The interface allows them to log in, submit their details, and upload certificates or photographs related to events or meetings.

Data Storage: The submitted information is stored in a centralized database. The database stores various records, including event details, certificates, and other relevant information.

Functionality:

- Faculty and students log in to the web interface.
- They input their data regarding attended events, meetings, or other activities.
- They upload supporting documents such as certificates and photographs.

5.1.2 Admin Interface

Admin Dashboard: The admin accesses a specialized interface that provides oversight and management capabilities. The admin can view detailed reports and download necessary data.

Data Management: The admin's interface is connected to the database, allowing them to access, manage, and download records related to faculty and students.

Functionality:

- The admin can view detailed logs of student and faculty submissions.
- The admin can download reports and manage the stored data.
- The system allows the admin to maintain the integrity and accessibility of data within the database.

5.2 Data Flow And System Integration

The system integrates the front-end interfaces with the backend database, ensuring seamless data flow and secure access:

5.2.1 Data Storage and Retrieval:

Database: All user inputs from faculty and students are stored in a structured database. This database is designed to handle queries from both the faculty/students' web interface and the admin's management interface.

Functionality:

- Data submitted through the web interface is automatically stored in the database.
- The database supports various queries and transactions to retrieve or update data as needed.

5.2.2 Data Access and Security:

Access Control: The system employs role-based access control, ensuring that only authorized users can access specific data or perform certain actions.

Security Measures: Security protocols are implemented to protect the integrity of the database and the confidentiality of user information.

Functionality:

- Role-based access control differentiates between faculty, students, and admin users.
- Security protocols ensure data is securely stored and accessed.

5.3 Downloading And Reporting

The system provides capabilities for the admin to download detailed reports, facilitating efficient data management and oversight.

Report Generation

- **Download Details**: The admin interface allows for the generation and download of detailed reports regarding student and faculty activities, submissions, and other relevant data.

Functionality:

- The admin can generate reports based on the data stored in the database.
- Reports can be downloaded for further analysis or record-keeping.

5.4 System Integration

The system is fully integrated, combining the front-end (user interfaces) and back-end (database) components, ensuring that all parts of the system communicate effectively.

5.4.1 Integration with the Database:

- **Backend Processing**: The system backend, possibly built on frameworks like Flask, handles data processing, authentication, and the interaction between the web interfaces and the database.

Functionality:

- Integration ensures real-time data flow between the user interfaces and the database.
- Backend processing includes user authentication, data validation, and secure data transmission.

CHAPTER 6
REQUIREMENTS

6.1 Functional Requirements

User Authentication and Authorization

- The system shall allow users (students, faculty, and admin) to sign up and log in using unique credentials.
- The system shall enforce role-based access control, where students and faculty can submit forms, and admins have the ability to filter and download event details.

Event Form Submission:

- The system shall allow students and faculty to submit forms detailing the events they have attended, including the type of event (workshop, conference, hackathon) and other relevant details.
- The system shall allow users to upload certificates and photos associated with the events.

Form Data Management:

- The system shall store submitted forms in a database, associating them with the respective user.
- The system shall categorize event details based on the type of event.

Admin Filtering and Downloading:

- The system shall provide admins with the ability to filter event data based on the type (workshop, conference, hackathon) or other criteria.
- The system shall allow admins to download filtered event details in a report format.

6.2 Non-Functional Requirements

Performance

- The system shall be able to handle simultaneous submissions from multiple users without significant delays.
-

- The system shall provide fast response times for filtering and downloading large sets of event data.

Security

- The system shall ensure secure user authentication and data access, preventing unauthorized access to sensitive information.
- The system shall encrypt sensitive data, such as user passwords, to protect against data breaches.

Usability

- The system shall have a user-friendly interface that is easy to navigate for all users, including those with minimal technical skills.
- The system shall provide clear instructions and feedback to users during the submission and data retrieval processes.

Reliability

- The system shall ensure high availability, with minimal downtime, to allow users to access the platform at any time.

Scalability

- The system shall be scalable to handle an increasing number of users and event submissions as the university grows.

CHAPTER 7

DESIGN

The Design consists of a use case diagram, sequence diagram, data flow diagram, and many more as follows.

7.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows Within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

The symbols depict the four components of the data flow diagram:

External entity:

An outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system, or a business system. They are also known as terminators, sources and sinks, or actors. They are typically drawn on the edges of the diagram.

Process:

The process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules.

Data store:

The files or repositories that hold information for later use, such as a database table or a membership form.

Data flow:

The route that data takes between the external entities, processes, and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like “Billing details”. Levels in DFD are numbered 0, 1, 2 or beyond.

Level-0 DFD:

It is also known as a context diagram. It's designed to be an abstract view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

Figure 7.1 Level-0 Data Flow Diagram

7.2 Sequence Diagram

Sequence models, which show the sequence of object interactions. These are represented using a UML sequence or a collaboration diagram. Sequence models are dynamic models. Sequence models are dynamic models that describe, for each mode of interaction, the sequence of object interactions that take place. When documenting a design, you should produce a sequence model for each significant interaction. If you have developed a use case model then there should be a sequence model for each use case that you have identified.

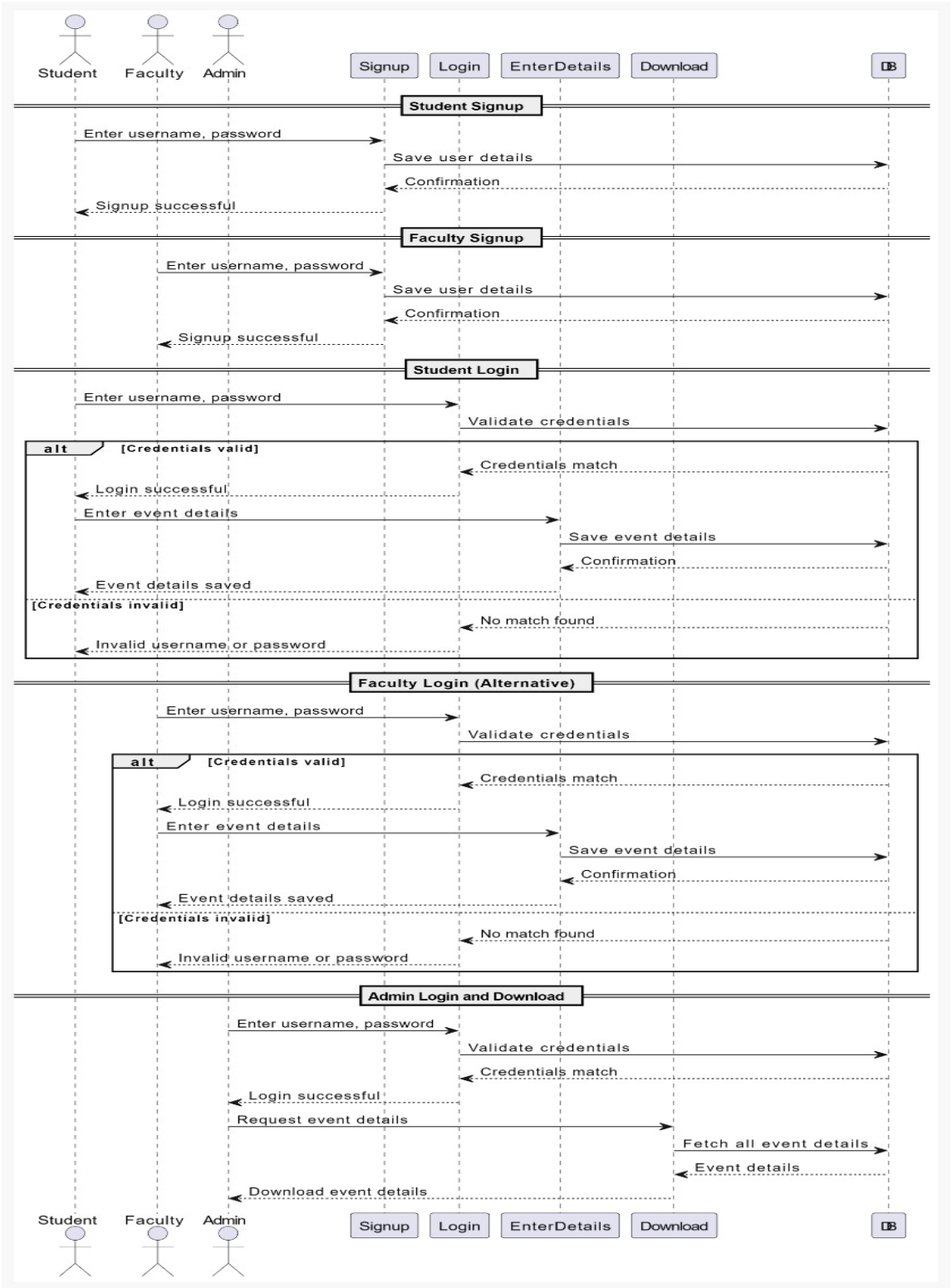

Figure 7.2 Sequence Diagram

The sequence diagram outlines the interactions among students, faculty, and admin within a system. It details the process flow for signup, login, event details entry, and event details download. Here's a breakdown of the different scenarios illustrated in the diagram:

1. Student Signup:

Action: The student enters a username and password.

Process: The system saves the user details.

Outcome: A confirmation is sent back to the student indicating that the signup was successful.

Both students and faculty can sign up with a username and password, which the system saves, and they receive confirmation.

2. Faculty Signup:

Action: Similar to students, the faculty member enters a username and password.

Process: The system saves the faculty's details.

Outcome: A confirmation is sent back, indicating that the signup was successful.

3. Student Login:

Action: The student enters their username and password to log in.

Validation:

If the credentials are valid, the system confirms a match and logs the student in successfully.

If the credentials are invalid, the system sends an error message indicating "Invalid username or password."

Event Entry:

- Upon successful login, the student can enter event details.
- The system saves these details and sends back a confirmation.
- Validation occurs for each login attempt.
- Upon successful validation, students and faculty can enter event details, while admins can request and download event details.
- Error handling is in place for invalid credentials.

4. Faculty Login (Alternative):

Action: The faculty member enters their username and password.

Validation:

If the credentials are valid, the system confirms the match and logs the faculty in.

If the credentials are invalid, an error message is returned.

Event Entry:

Similar to students, the faculty member can enter event details upon successful login.

5. Admin Login and Download:

Action: The admin enters their username and password.

Validation:

The system validates the credentials, and upon a successful match, logs the admin in.

Event Details Request:

The admin can request event details, which are fetched by the system.

Download:

The event details are then sent to the admin for download.

This diagram effectively shows the roles and how each interacts with the system through various processes.

7.3 Use Case Diagram

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system.

An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve the scope of the system

Common components include:

- **Actors:** The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. They must be external objects that produce or consume data.
- **System:** A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.
- **Goals:** The end result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

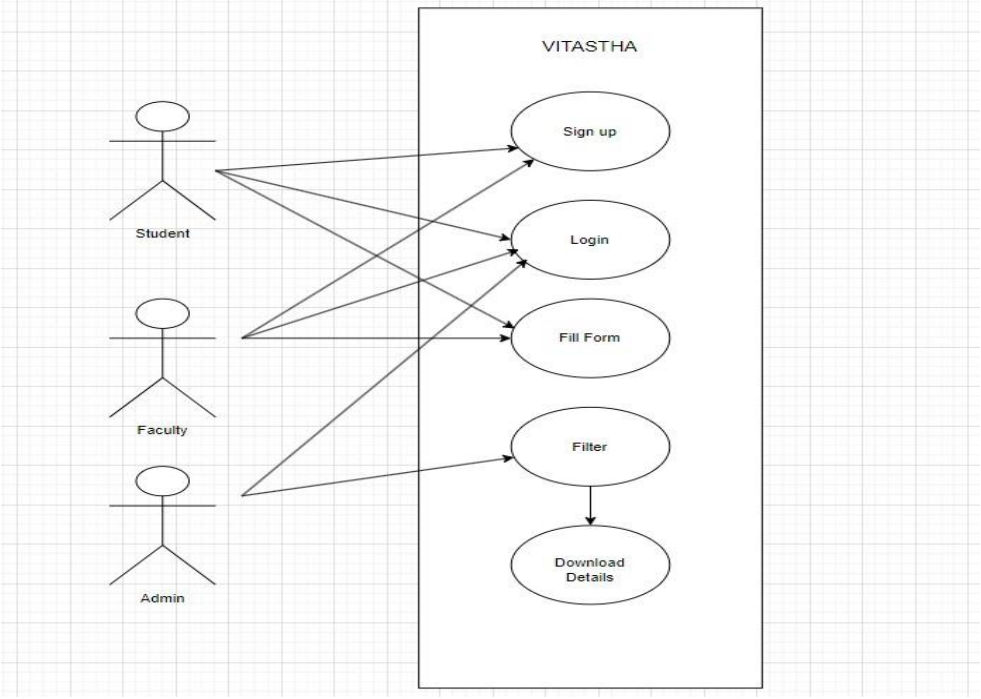

Figure 7.3 Use Case Diagram

The use case diagram represents the interactions between different user roles and the system. The diagram shows how students, faculty, and admins interact with various functionalities within the system.

Actors:

1. **Student**:
    - Represents the users who are students interacting with the system.
2. **Faculty**:
    - Represents the faculty members who also interact with the system.
3. **Admin**:
    - Represents the system administrator with special permissions and access to additional functionalities.

Use Cases (System Functions):

1. **Sign Up**:
    - **Interaction**: Students, faculty, and admin all have the ability to sign up in the system. This involves creating an account by providing necessary details like username, password, etc.

2. **Login**:
    - **Interaction**: All actors (students, faculty, and admin) can log into the system using their credentials after signing up. The login process would validate the credentials and grant access to the system functionalities based on their role.
3. **Fill Form**:
    - **Interaction**: This function allows users (students and faculty) to fill out forms, likely related to event details, academic submissions, or other data entry tasks.
4. **Filter**:
    - **Interaction**: This function might be used by users (students and faculty) to filter data within the system. It could be filtering events, information, or submissions relevant to their needs.
5. **Download Details**:
    - **Interaction**: This function is primarily available to the admin. It allows the admin to download details from the system, possibly related to student activities, event logs, or other relevant data stored in the system.

CHAPTER 8
IMPLEMENTATION

8.1 MODULES

- Student Interface Design
- Faculty Interface Design
- Admin Interface Design
- Database Management
- API Development
- Integration and Testing

8.2 Module Description

8.2.1 Student Interface Design

This module focuses on creating an intuitive form for students to input details about the events they attend, including certificates and photographs. The form should be user-friendly and ensure that students can easily submit their information.

Functionality

- Design a user-friendly interface for students to enter event details.
- Implement form fields for certificates and photographs.
- Ensure validation and error handling for user inputs.

8.2.2 Faculty Interface Design

This module involves designing a form for faculty members to submit details about their events and meetings, along with certificates and photographs. The form should be designed to be straightforward and efficient for faculty use.

Functionality

- Create an easy-to-navigate form for faculty event and meeting submissions.
- Include options for uploading certificates and photographs.
- Validate and handle errors in the submission process.

8.2.3 Admin Interface Design

The admin interface module includes creating a dashboard page that allows admins to view and manage student and faculty information. This includes the ability to download and review data.

Functionality:

- Design a comprehensive dashboard to display student and faculty data.
- Implement features for data download and viewing.

8.2.4 Database Management

This module involves setting up a database to store information about students, faculty, and admins. It includes the design and implementation of the database schema.

Functionality:

- Design a database schema to accommodate student, faculty, and admin data.
- Implement data storage and retrieval mechanisms.
- Ensure data integrity and security.

8.2.5 API Development

This module includes the development of APIs to handle authentication for students, faculty, and admins, as well as form submissions and data retrieval.

Functionality:

- Develop APIs for user authentication (student, faculty, admin).
- Create endpoints for form submissions and data retrieval.
- Ensure API security and efficiency.

8.2.6 Integration And Testing

This module involves integrating the frontend and backend using the Flask framework and conducting thorough testing to ensure everything functions correctly.

Functionality:

- Integrate frontend interfaces with backend APIs using Flask.
- Conduct integration testing to ensure proper functionality.
- Perform end-to-end testing to validate the complete system.

8.3 Tools Used

8.3.1 Programming Languages

Python:

- **Description**: Python is a high-level, interpreted programming language known for its simplicity and readability. It's widely used in web development, data analysis, machine learning, and more.
- **Usage**: In this project, Python is employed for backend development, data processing, and integrating with frameworks and libraries such as Flask. Its extensive standard library and supportive ecosystem make it suitable for handling various tasks, including form submission, authentication, and database management.

HTML:

- **Description**: HTML (HyperText Markup Language) is the standard markup language used for creating web pages and applications. It structures the content on the web.
- **Usage**: HTML is used to build the structure of the web application's frontend. It defines the elements of the user interface, such as forms, buttons, and layout, enabling students, faculty, and admins to interact with the application.

CSS:

- **Description**: CSS (Cascading Style Sheets) is used to control the presentation, formatting, and layout of HTML elements. It enhances the visual appeal of web pages.
- **Usage**: CSS styles the HTML elements to make the web application visually appealing and user-friendly. It controls aspects such as fonts, colors, spacing, and positioning, ensuring a cohesive and attractive design.

JavaScript:

- **Description**: JavaScript is a dynamic, high-level programming language that adds interactivity to web pages. It's essential for client-side scripting and dynamic content updates.
- **Usage**: JavaScript enhances user interactions within the web application. It handles form validations, dynamic content updates, and interactive features like displaying and interacting with maps.

8.3.2 Technologies, Libraries, And Frameworks

Flask:

- **Description**: Flask is a lightweight web framework for Python. It's designed for building web applications quickly with minimal overhead.
- **Usage**: Flask is used to develop the web server for the application. It handles routing, templates, and session management, facilitating communication between the frontend and backend components. Flask is chosen for its simplicity and flexibility in developing RESTful APIs and integrating with other tools.

SQLAlchemy:

- **Description**: SQLAlchemy is a SQL toolkit and Object-Relational Mapping (ORM) library for Python. It simplifies database interactions by allowing developers to work with database tables as Python objects.
- **Usage**: SQLAlchemy is used to manage interactions with the database. It provides a high-level API for querying and manipulating data, defining database schema, and handling relationships between entities. This abstraction simplifies database management and ensures efficient data handling.

Jinja2:

- **Description**: Jinja2 is a templating engine for Python. It allows for dynamic content generation by embedding Python code in HTML templates.
- **Usage**: Jinja2 is integrated with Flask to render HTML templates dynamically. It enables the generation of HTML pages with variable data, such as user profiles or event details, based on backend data.

8.3.3 IDE

VS Code:

- **Description**: Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft. It supports a wide range of programming languages and comes with built-in debugging, Git integration, and a robust extension ecosystem.
- **Usage**: VS Code is used for writing, managing, and debugging the codebase of the project. Its features, including an integrated terminal, code linting, and version control, make it a powerful tool for development.

8.3.4 Database

SQLAlchemy:

- **Description**: As mentioned earlier, SQLAlchemy is a comprehensive toolkit for SQL and ORM in Python. It supports various database backends and provides a Pythonic way to interact with databases.
- **Usage**: In this project, SQLAlchemy is used to define and interact with the database schema, manage data relationships, and execute queries. It ensures that the backend can efficiently store and retrieve information related to students, faculty, and admin users.

CHAPTER 9

TEST CASE

System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is the process of executing the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied.

9.1 Unit Testing

This involves testing individual components or modules of a system in isolation to ensure they work correctly. In your descriptions, some tests could be considered unit tests if they focus on the smallest parts of the algorithms (like testing specific functions or methods).

9.1.1 For Student Signup

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Valid Signup	Valid Student USN	Student registered successfully	Pass
2	Invalid Student USN	Existing USN	Student already exist	Pass

Table 9.1 Unit Testing for Student Signup

9.1.2 For Student Login

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Valid Login	Valid Student USN and password	Student successful Login	Pass
2	Invalid Login	Incorrect details	Invalid USN or Password	Pass

Table 9.2 Unit Testing for Student Login

9.1.3 For Student Form

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Invalid Name	Numbers or Special Characters	Name should contain only letters	Pass
2	Invalid date	Future Date	Can't be selected	Pass
3	Invalid Place	Numbers or Special Characters	Place should contain only letters	Pass
4	Invalid File upload	More than 3 files file size greater than 16MB	You can upload maximum of 3 files	Pass
5.	Valid Form	Valid Credentials	Form Submitted Successfully	Pass

Table 9.3 Unit Testing for Student Form

9.1.4 For Faculty Signup

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Valid Signup	Valid Faculty Name, Email and Password.	Faculty registered successfully	Pass
2	Invalid Faculty Email	Existing Email	Faculty already exist	Pass

Table 9.4 Unit Testing for Faculty Signup

9.1.5 For Faculty Login

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Valid Login	Valid Email and password	Faculty login successful	Pass
2	Invalid Login	Incorrect details	Invalid Email or Password	Pass

Table 9.5 Unit Testing for Faculty Login

9.1.6 For Faculty Form

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Invalid Name	Number or special character	Name should contain only letters	Pass
2	Invalid Date	Future Date	Can’t be selected	Pass
3	Invalid Place	Number or special character	Place should contain only letters	Pass
4	Invalid files upload	More than 3 files or file size greater than 16MB	You can upload maximum of 3 files	Pass
5.	Valid Form	Valid Credentials	Form Submitted Successfully	Pass

Table 9.6 Unit Testing for Faculty Form

9.1.7 For Admin Login

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Valid Login	Valid Admin Email and password	Admin Login successful	Pass
2	Invalid Login	Incorrect details	Invalid Email or Password	Pass

Table 9.7 Unit Testing for Admin Login

9.1.8 For Admin Download Details

Test Case Id	Test Scenario	Input	Expected Output	Pass/Fail
1	Extracting Data	From date, to date and category	No result found	Pass

Table 9.8 Unit Testing for Admin Download Details

CHAPTER 10
RESULTS AND DISCUSSION

10.1 Event Management System Results

Implementing the Department Event Management System has yielded significant improvements in the management and coordination of academic and extracurricular activities within the department. The system was designed to address specific challenges such as document misplacement and the inefficiencies of manual paperwork, leading to optimized post-event handling and enhanced overall event management.

Below are the key findings:

- Streamlined Documentation Process: The system's transition from manual paperwork to digital automation has drastically reduced the chances of document misplacement. Event-related documents are now stored securely and can be accessed easily, improving the efficiency and reliability of the event management process.
- Enhanced Coordination: The automation and centralized management of event-related tasks have significantly improved coordination. This has led to smoother execution of departmental events, minimizing delays and ensuring that all necessary preparations are completed on time.
- Adaptive and Scalable Framework: The system's design allows it to evolve continuously through regular monitoring and feedback. As the needs of university event management change, the system can adapt, ensuring that it remains effective and relevant in supporting both academic and extracurricular activities.

10.2 System Visualization And Performance

- Centralized Dashboard: The system features a centralized dashboard that provides real-time insights on task completion status. This allows event organizers and department heads to monitor progress and make informed decisions quickly.

- Document and Task Tracking: The system provides visual tracking of document submissions and task completions, ensuring transparency and accountability.

Figure 9.1 Home Page

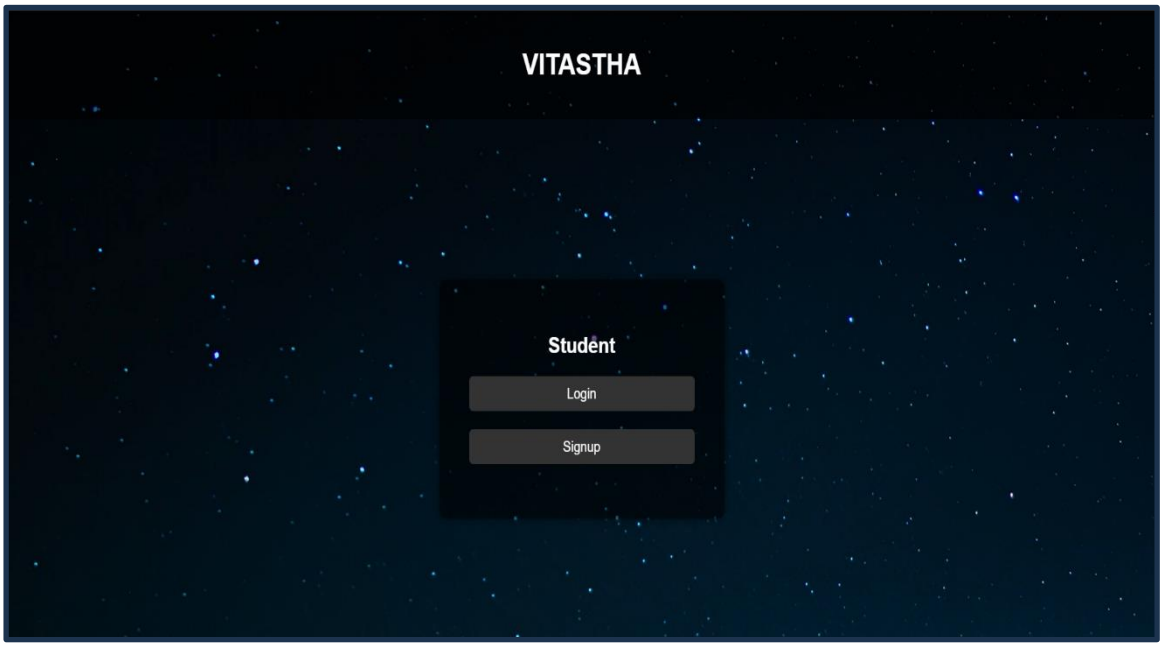

Figure 9.2 Student Login/Signup Page

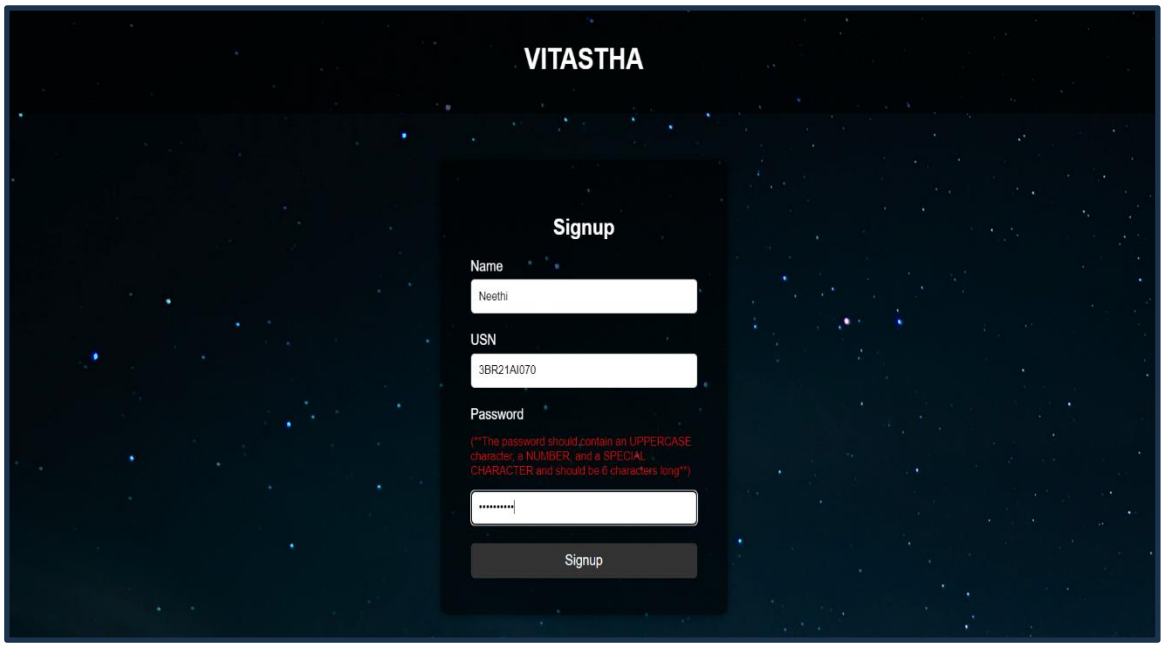

Figure 9.3 Student Signup Page

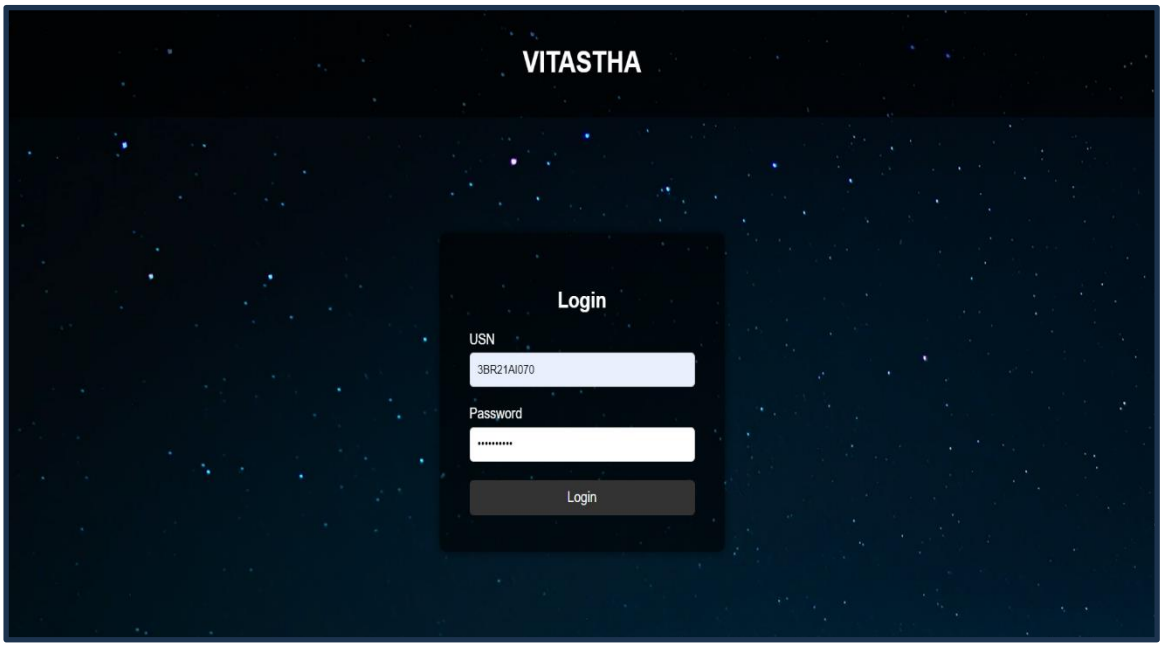

Figure 9.4 Student Login Page

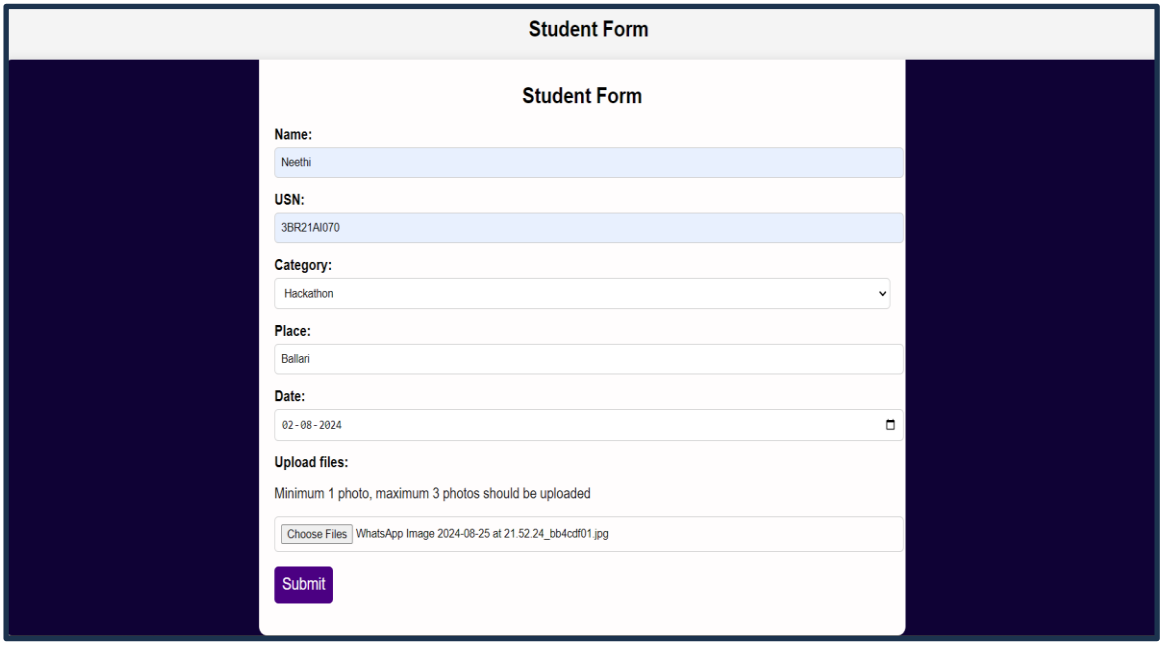

Figure 9.5 Student Form Page

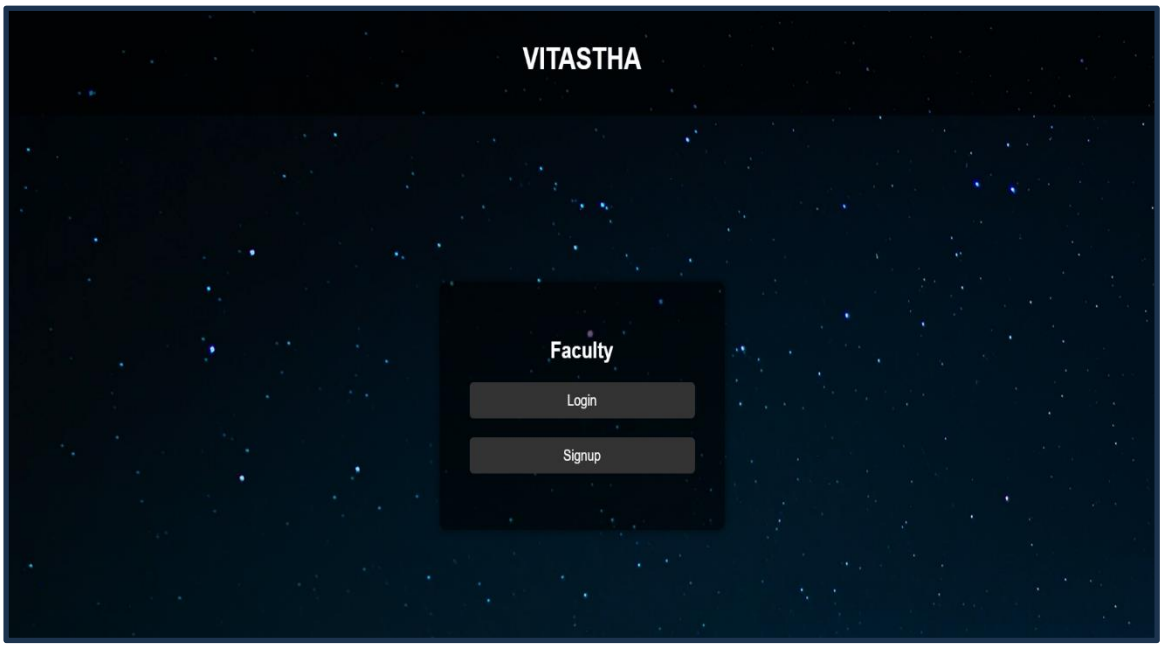

Figure 9.6 Faculty Login/Signup Page

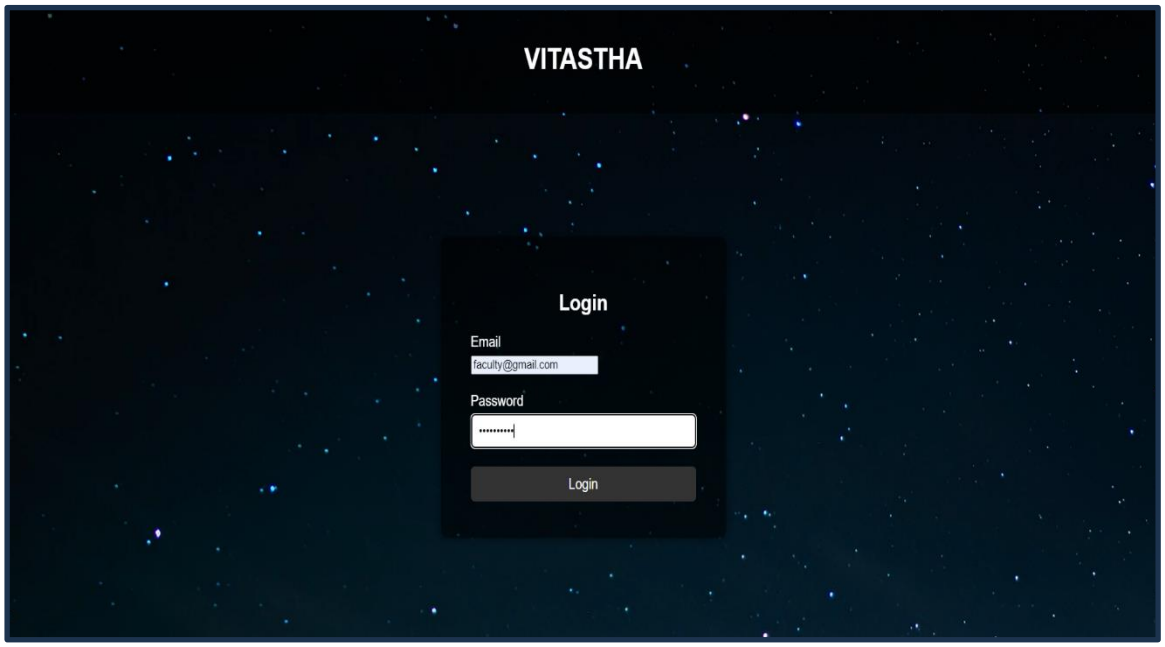

Figure 9.7 Faculty Login Page

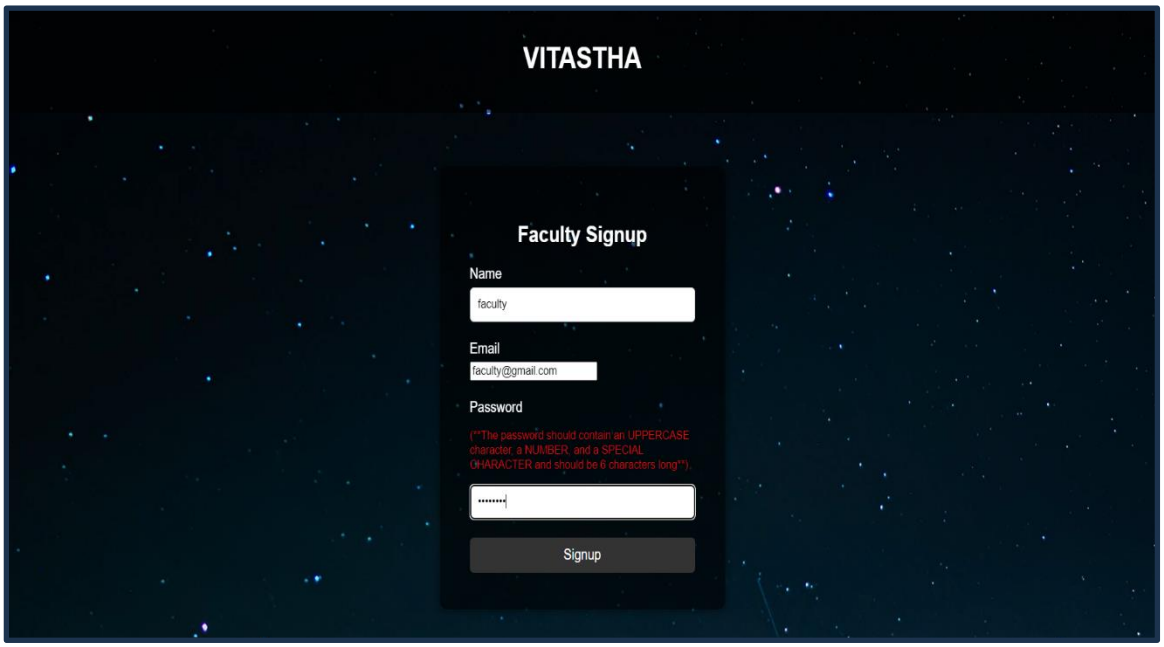

Figure 9.8 FacultySignup Page

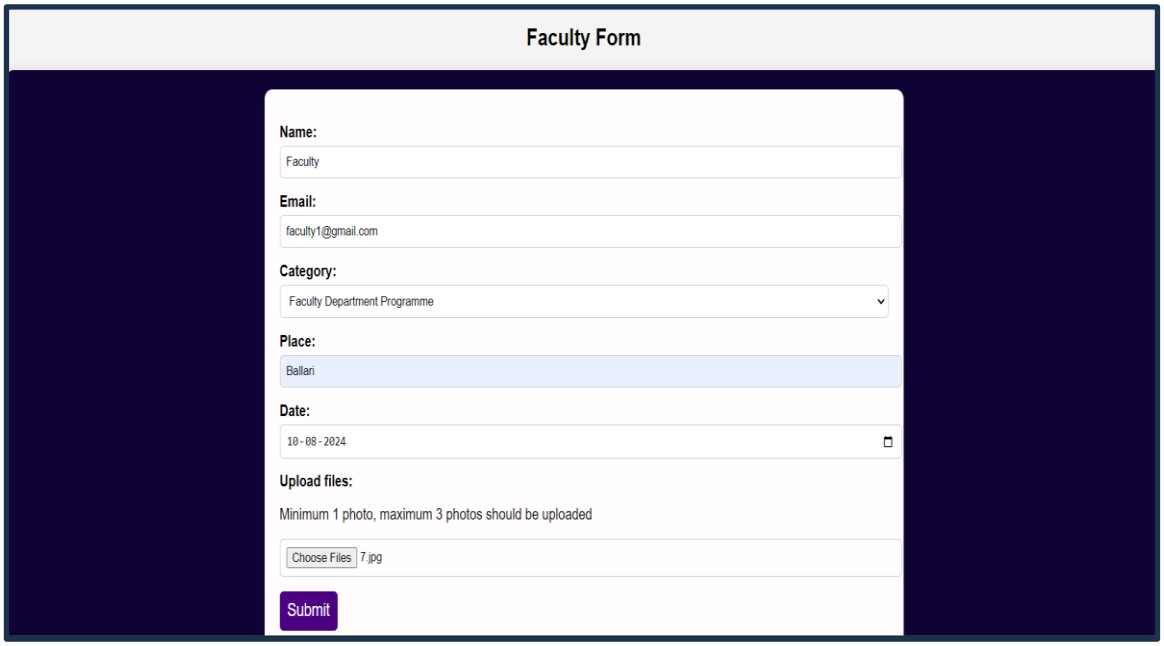

Figure 9.9 Faculty Form Page

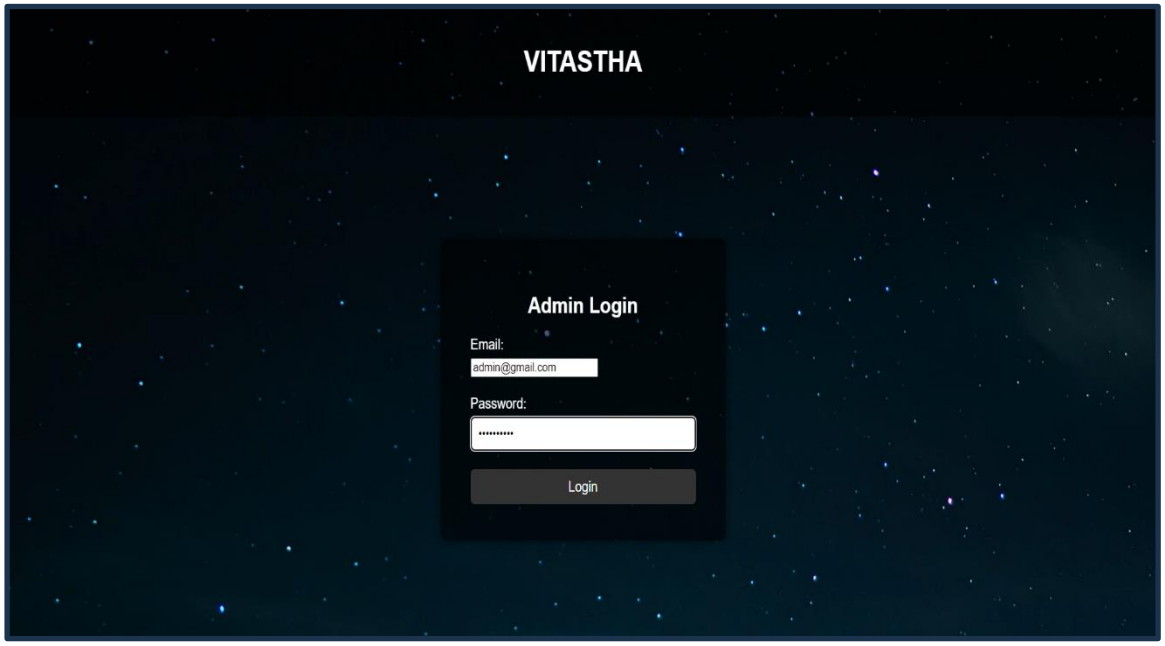

Figure 9.10 Admin Login Page

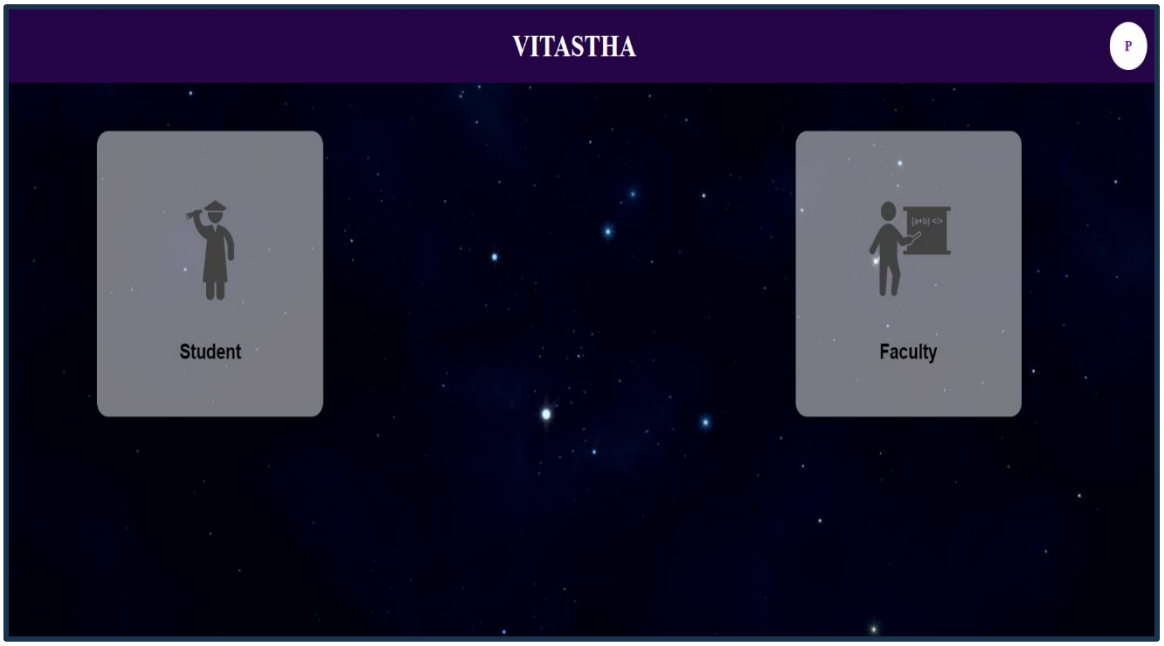

Figure 9.11 Admin Home Page

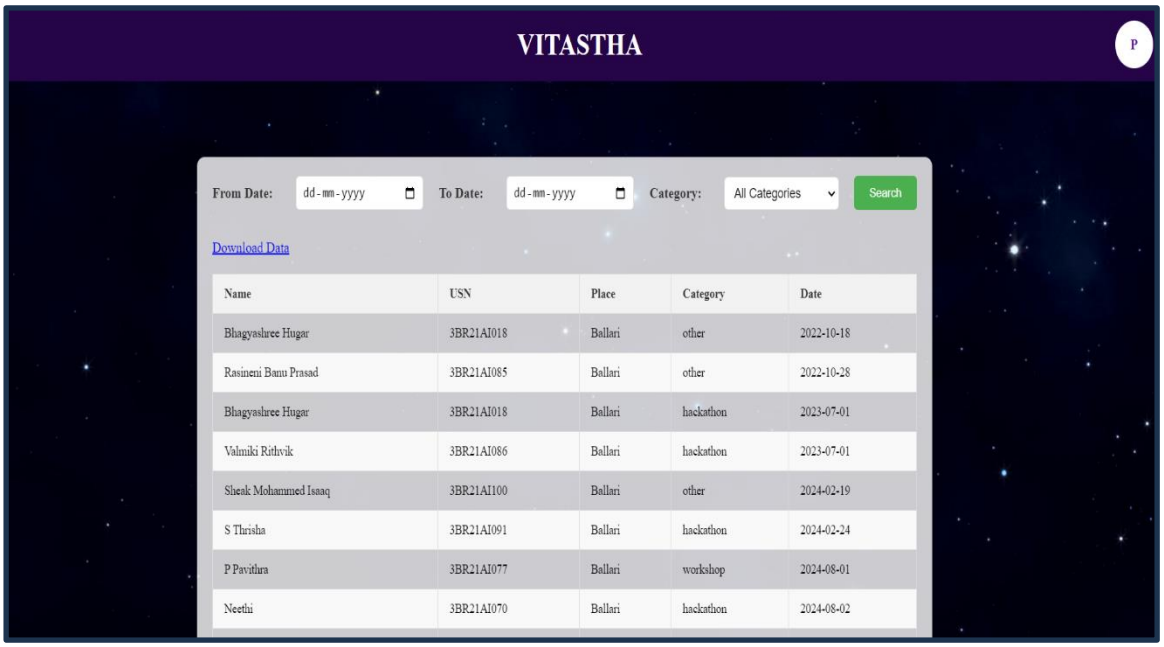

Figure 9.12 Student Details Page

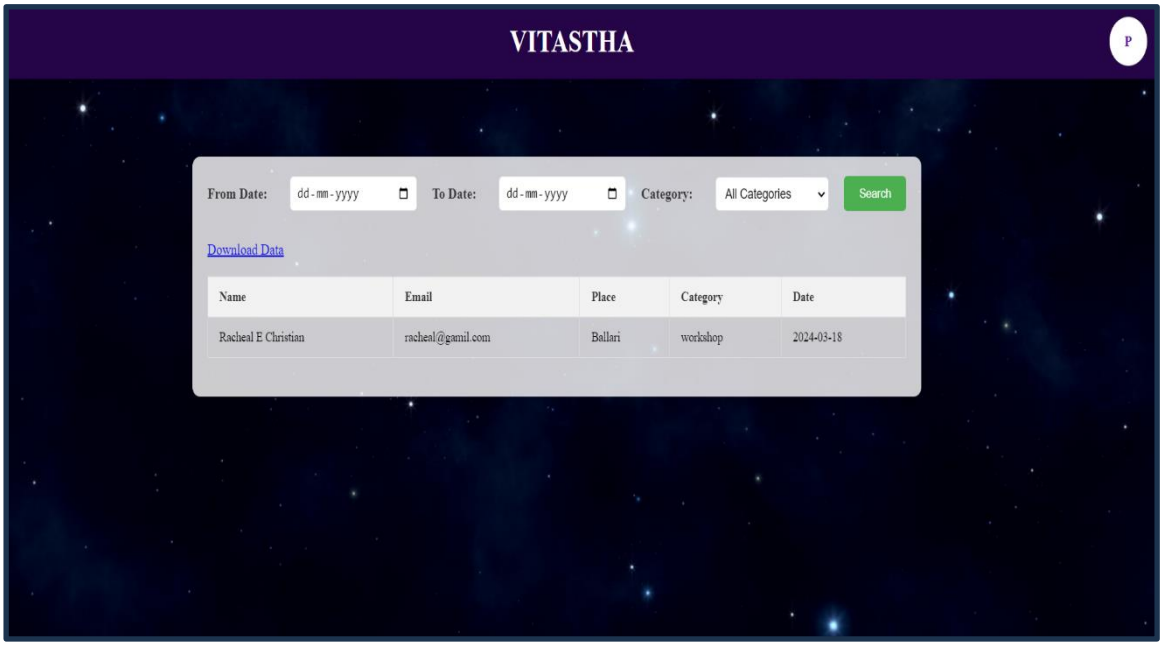

Figure 9.13 Faculty Details Page

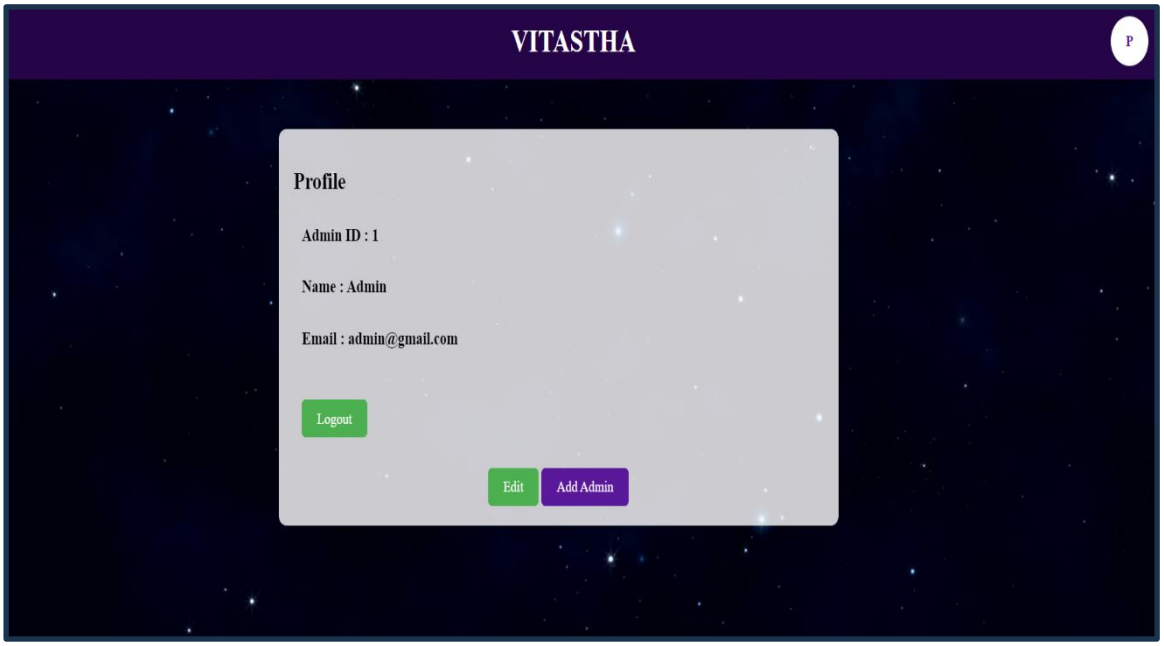

Figure 9.14 Admin Profile Page

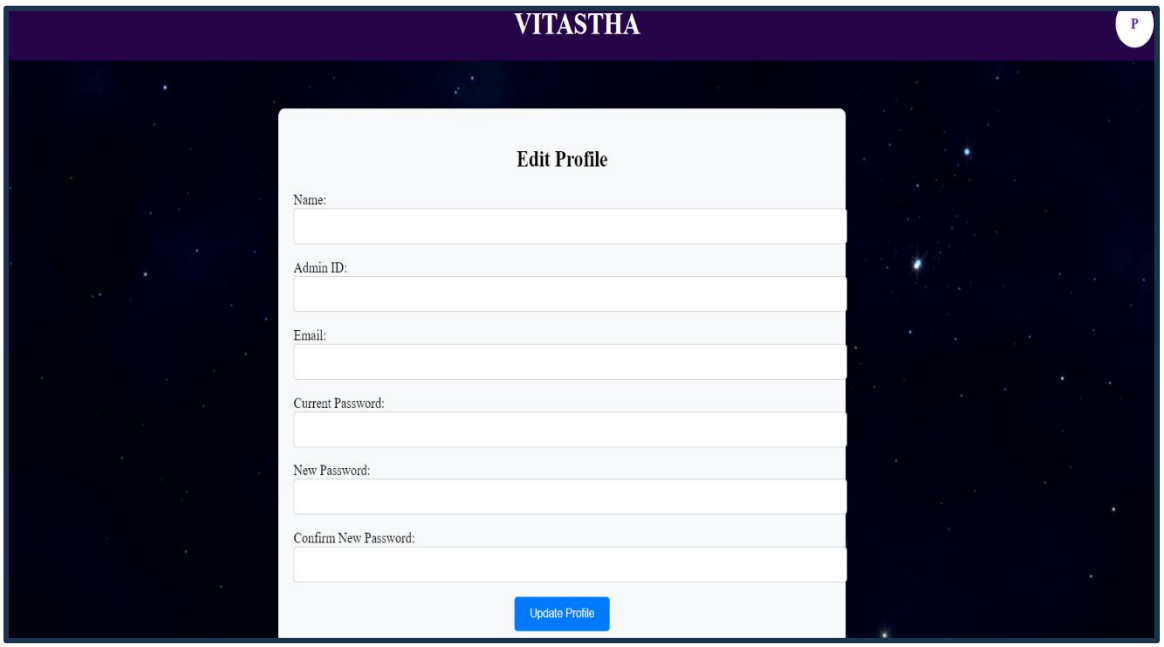

Figure 9.15 Admin Edit Profile Page

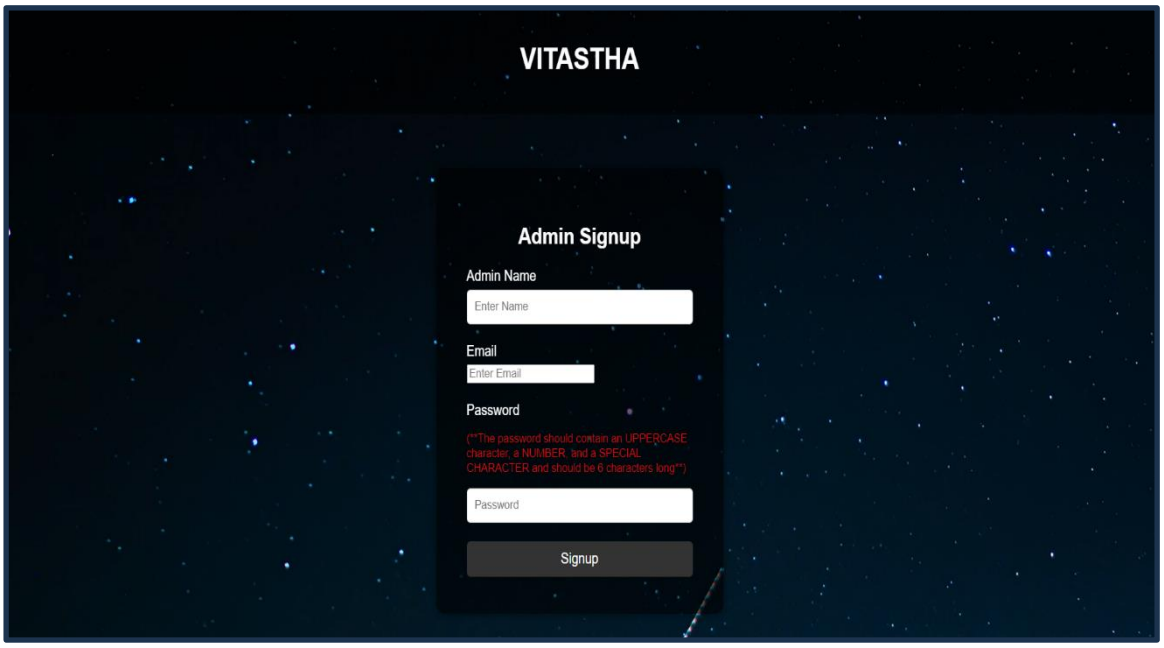

Figure 9.16 Admin Signup Page

CHAPTER 11

CONCLUSION

This Department Event Management System streamlines academic event by moving from manual processes to a centralized website. It improves data management and saves time. Key features include an integrated calendar, participant details, event categories, and options to download certificates and photos. The system is user-friendly and accessible remotely, adding value to event management and engagement within academic departments.

REFERENCES

[1] Nikita M. Bawankar, Ankita T. Tembhurne, Gayatri R. Chatap, Mansi M. Bandbuche, Tejaswini S. Dhote, Prof. Vaishali R. Surjuse, "Event Management System", International Journal of Advance Research and Innovative Ideas in Education (IJARIIE) ISSN(O): 2395-4396, volume 7 Issue 2, pg no: 128-131, 2021, https://ijariie.com/AdminUploadPdf/EVENT_MANAGEMENT_SYSTEM_ijariie13806.pdf

[2] Siddharth Katrapwar, Krupal Vora, Dr. Bhavesh Patel, Archana Chaugule, "Event Management Hub", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 p-ISSN: 2395-0072, Volume 9 Issue 1, pg no: 511-513, 2022, https://www.irjet.net/archives/V9/i1/IRJET-V9I190.pdf

[3] Gaurav Thombare, Pavan Jadhav, Sachin Dhere, Prasad Jadhav, Priyanka Raikar, "Evolving Trends in College Event Management", International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) ISSN (Online): 2581-942, volume 4 Issue 2, pg no: 342-346, 2024, https://ijarsct.co.in/Paper17045.pdf