CASE STUDY

ON

FEEDBACK MANAGEMENT SYSTEM

INTRODUCTION

1. The Feedback Management System is a **web-based application** developed to efficiently collect, manage, and analyse feedback from users in a structured and organized manner.

2. The system aims to **replace traditional paper-based and manual feedback collection methods**, thereby reducing human errors and improving data accuracy and reliability.

3. Administrators are provided with the ability to **create, manage, and publish feedback forms** for various events conducted by the organization.

4. Users can easily **select the published events and submit their feedback** through a simple and user-friendly interface.

5. The administrator can **view all submitted feedback, search feedback records, filter them based on ratings, and delete feedback when necessary**.

6. The project ensures **role-based access control, secure data handling, and ease of use**, making the system reliable and efficient for both administrators and users

ABSTRACT

The **Feedback Management System (FMS)** is a web-based application designed to simplify and automate the process of collecting feedback for events conducted by an organization or institution. The system provides a structured platform where administrators can create and publish event-specific feedback forms, ensuring that feedback is collected in an organized and consistent manner. Users can securely submit their feedback for selected events through an easy-to-use interface, eliminating the need for traditional paper-based feedback methods. The primary objective of this project is to offer an efficient and reliable mechanism for feedback collection, management, and analysis. By automating the entire feedback process, the system reduces manual data handling, minimizes errors, and improves data accuracy. The application includes advanced features such as searching feedback records, filtering feedback based on ratings, and viewing feedback in an event-wise manner, enabling administrators to gain valuable insights and make informed decisions. The Feedback Management System is developed using modern technologies such as **Spring Boot** for backend development, **Spring Security** for authentication and role-based access control, **JPA (Hibernate)** for database interaction, and **MySQL** for data storage. The frontend is designed using **Thymeleaf, HTML, and CSS**, providing a responsive and user-friendly interface. The system follows a **layered architecture**, which enhances scalability, maintainability, and security, making it suitable for real-world applications.

OBJECTIVES OF THE PROJECT

1. **To automate the feedback collection, process**
   The system aims to replace manual and paper-based feedback collection methods with a fully automated digital solution. This helps in saving time, reducing human errors, and ensuring faster and more accurate feedback processing.

2. **To allow administrators to create and publish event-based feedback forms**
   The application enables administrators to create feedback forms for different events and publish them as required. This ensures that feedback is collected separately for each event, making evaluation and analysis more structured and meaningful.

3. **To allow users to submit feedback for selected events**
   Users can easily select an event from the available list and submit their feedback through a simple and user-friendly interface.

4. **To implement role-based authentication and authorization**
   The system uses role-based access control to differentiate between administrators and users. This ensures that only authorized users can access specific features, improve system security and prevent unauthorized access to sensitive data.

5. **To enable administrators to search and filter feedback based on ratings**
   The application provides powerful search and filter options that allow administrators to quickly locate feedback records and analyse them based on ratings.

CLIENT REQUIREMENTS

The client requires a system with the following features:

- **Admin** should be able to:
    - Login securely
    - Create events
    - Publish/unpublish feedback forms
    - View all feedback
    - Filter feedback by rating
    - Search feedback by name or email
    - Delete feedback records

- **User** should be able to:
    - Register and login
    - Select an event
    - Submit feedback
    - Logout securely

TECHNOLOGIES USED

Framework: Spring Boot Java Framework.

• Java 8+

• Maven

• STS

• Apache Tomcat

• Spring core, Spring security(jwt), Spring data JPA(Hibernate) etc

• MySQL Database.

• HTML and CSS

SYSTEM REQUIREMENTS

- **Backend:** Spring Boot, Spring Security, JPA (Hibernate)
- **Database:** MySQL 8.0
- **IDE:** Spring Tool Suite (STS)
- **Web Server:** Apache Tomcat
- **Frontend:** HTML, CSS, Thymeleaf
- **Browser:** Google Chrome
- **Testing:** JUnit

PROJECT MODULES

1.Admin Module

- Secure login
- Event creation
- Publish feedback forms
- View all feedback
- Search feedback
- Filter feedback by rating
- Delete feedback

2.User Module

- User registration
- Secure login
- Select event
- Submit feedback
- Logout

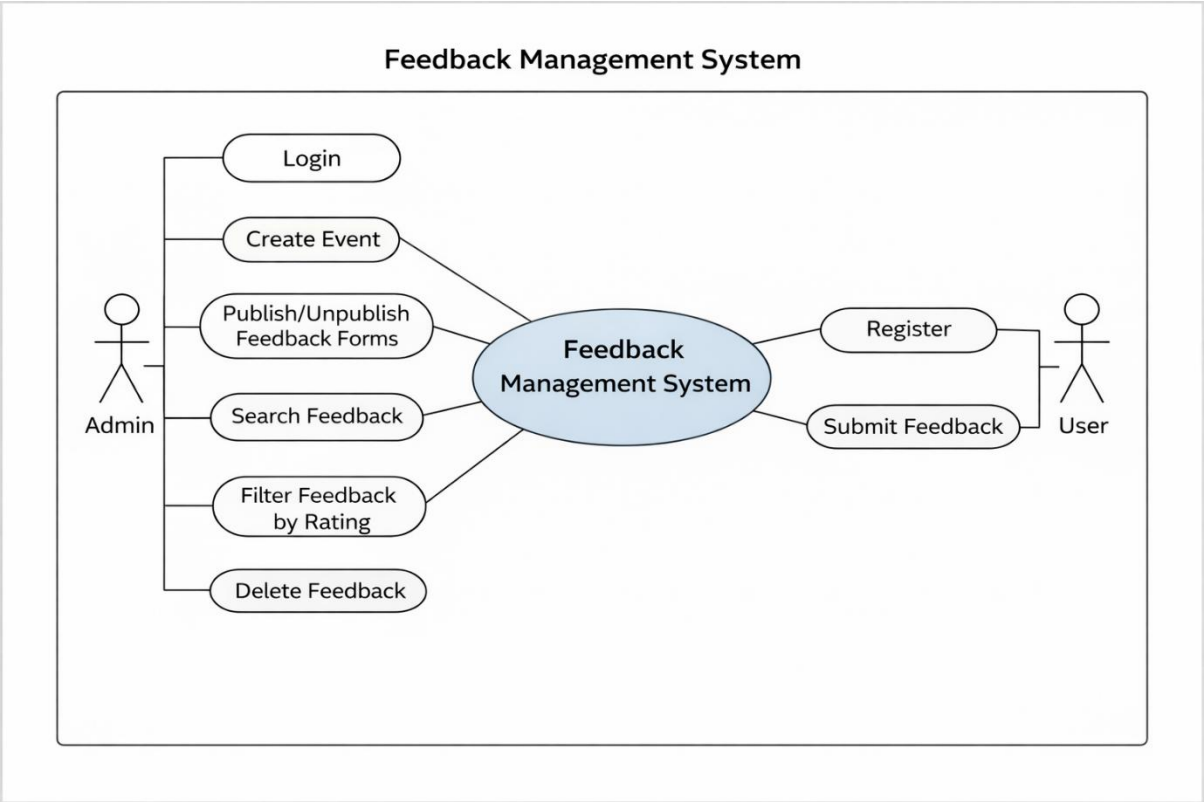
3.Event Module

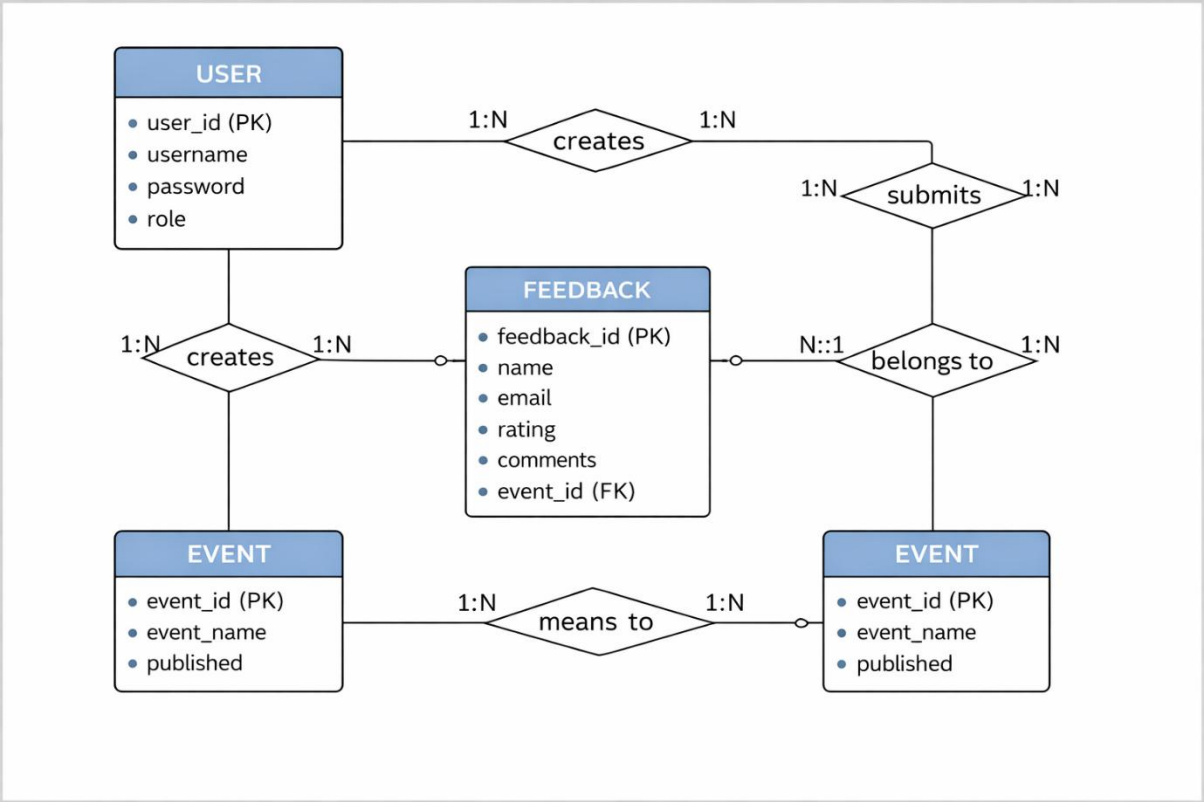
- Event creation
- Event publishing
- Event listing

4.Feedback Module

- Feedback submission
- Feedback storage
- Feedback retrieval
- Feedback deletion

ER DIAGRAM

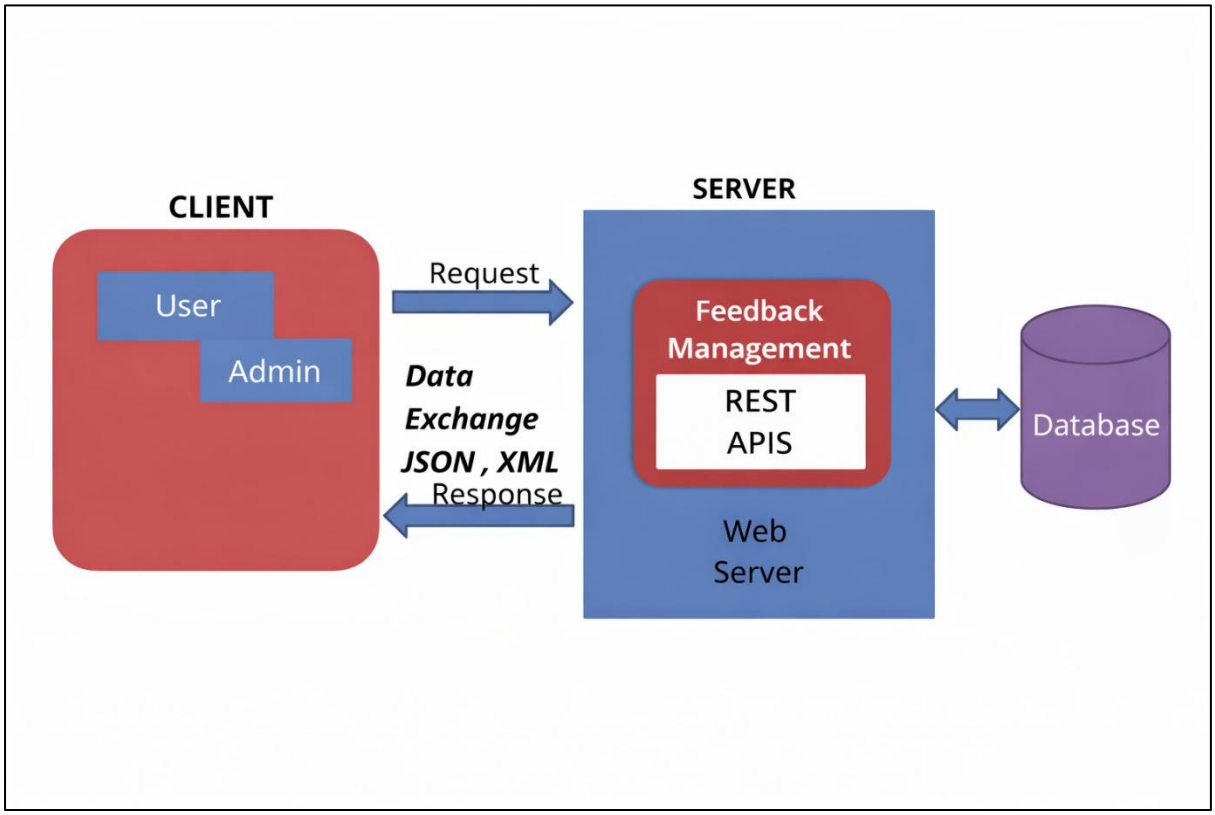

Feedback Management System
Login
Create Event
Publish/Unpublish Feedback Forms
Search Feedback
Filter Feedback by Rating
Delete Feedback
Admin
Feedback Management System
Register
Submit Feedback
User


USER
user_id (PK)
username
password
role
1:N
creates
1:N
1:N
submits
1:N
FEEDBACK
feedback_id (PK)
name
email
rating
comments
event_id (FK)
1:N
creates
1:N
N::1
belongs to
1:N
EVENT
event_id (PK)
event_name
published
1:N
means to
1:N
EVENT
event_id (PK)
event_name
published

CILENT-SERVER ARCHITECTURE

The Feedback Management System is based on a **client–server architecture**, where the client and server have clearly defined roles. The client side consists of a web-based interface developed using **HTML, CSS, and Thymeleaf**, which is accessed through a web browser by users and administrators. All user actions such as login, event selection, feedback submission, searching, and filtering generate HTTP requests that are sent to the server.

The server side is implemented using **Spring Boot**, which handles request processing through controllers, business logic through services, and data access through repositories. The server communicates with a **MySQL database** to store and retrieve user, event, and feedback information. After processing the request, the server sends an appropriate response back to the client, ensuring secure, efficient, and reliable data exchange.



MODULES

1. Admin Module

1. Provides secure login access to administrators using role-based authentication.
2. Allows admins to create, manage, and publish feedback forms for different events.
3. Enables viewing, searching, filtering, and deleting of user feedback records.
4. Helps administrators analyse feedback effectively to improve event quality.

2. User Module

1. Allows users to register and log in securely to the system.
2. Enables users to select published events and submit their feedback.
3. Provides a simple and user-friendly interface for feedback submission.
4. Ensures secure session handling and proper logout functionality.

3. Event Module

1. Allows administrators to create events for which feedback can be collected.
2. Supports publishing and unpublishing of events based on requirements.
3. Displays only published events to users for feedback submission.
4. Helps organize feedback in an event-wise manner for better analysis.

4. Feedback Module

1. Collects user feedback including name, email, rating, and comments.
2. Stores feedback securely in the database using JPA and MySQL.
3. Supports searching and filtering feedback based on ratings and keywords.
4. Allows administrators to manage feedback records efficiently.

HTTP REQUEST METHODS

HTTP defines a set of request methods that indicate the action to be performed on feedback, events, and users in the system.

HTTP Method	URL	Description	Access Role
GET	http://localhost:8080/events	Retrieves the list of all published events	User/ Admin
GET	http://localhost:8080/feedback	Retrieves all submitted feedback records	Admin
GET	http://localhost:8080/feedback/event/5	Retrieves feedback for the event with ID 5	Admin
POST	http://localhost:8080/submit-feedback	Submits feedback for a selected event	User
PUT	http://localhost:8080/admin/event/10	Updates details of the event with ID 10	Admin
DELETE	http://localhost:8080/admin/feedback/7	Deletes the feedback record with ID 7	Admin

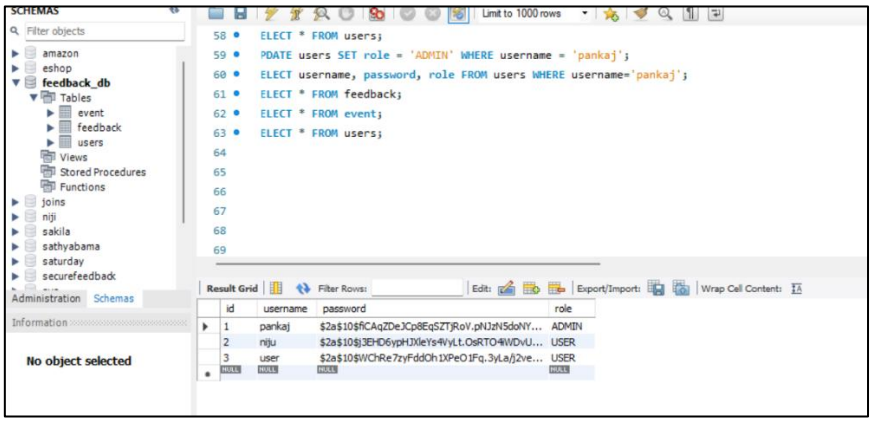
URI-(UNIFORM RESOURCE IDENTIFIER)

A URI is the address used to identify a resource on the internet or a web application. It tells what resource you want and where it is located

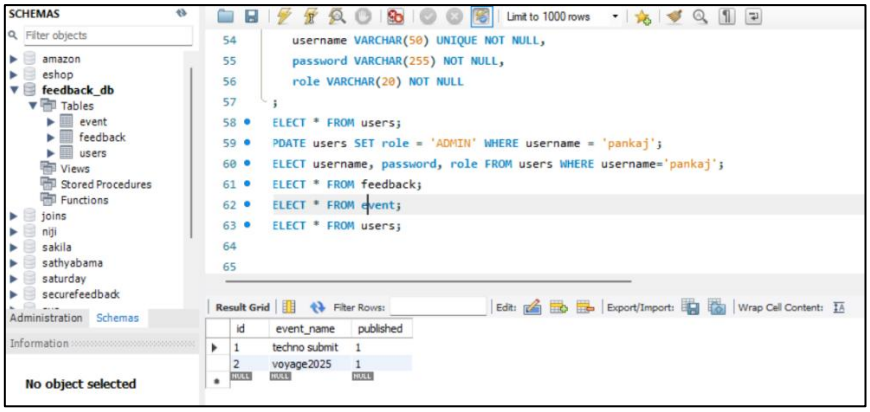
HTTP Method	**URI**	**Description**
GET	http://localhost:8080/events/	Returns the list of all published events
GET	http://localhost:8080/feedback/15	Returns the feedback details of feedback ID 15
POST	http://localhost:8080/feedback/	Submits new feedback for an event
PUT	http://localhost:8080/admin/events/8	Updates the event details of event ID 8
DELETE	http://localhost:8080/admin/feedback/5	Deletes the feedback of feedback ID 5

DATABASES

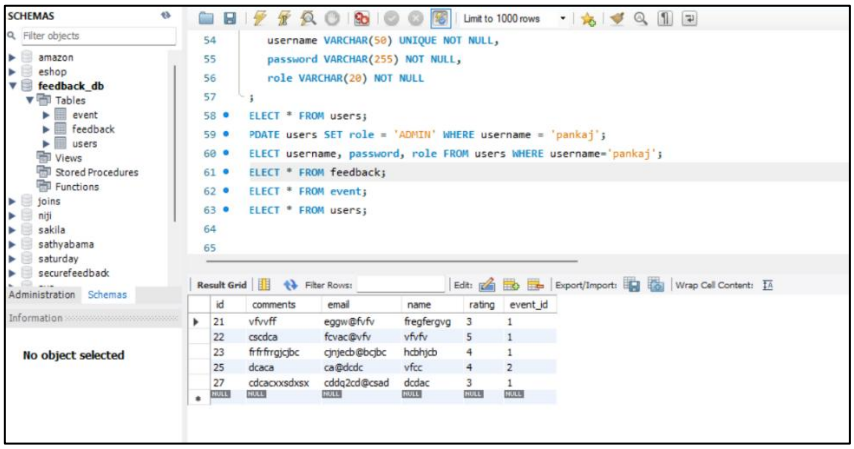
USER DETAILS DATABASE



EVENT LIST DATABASE

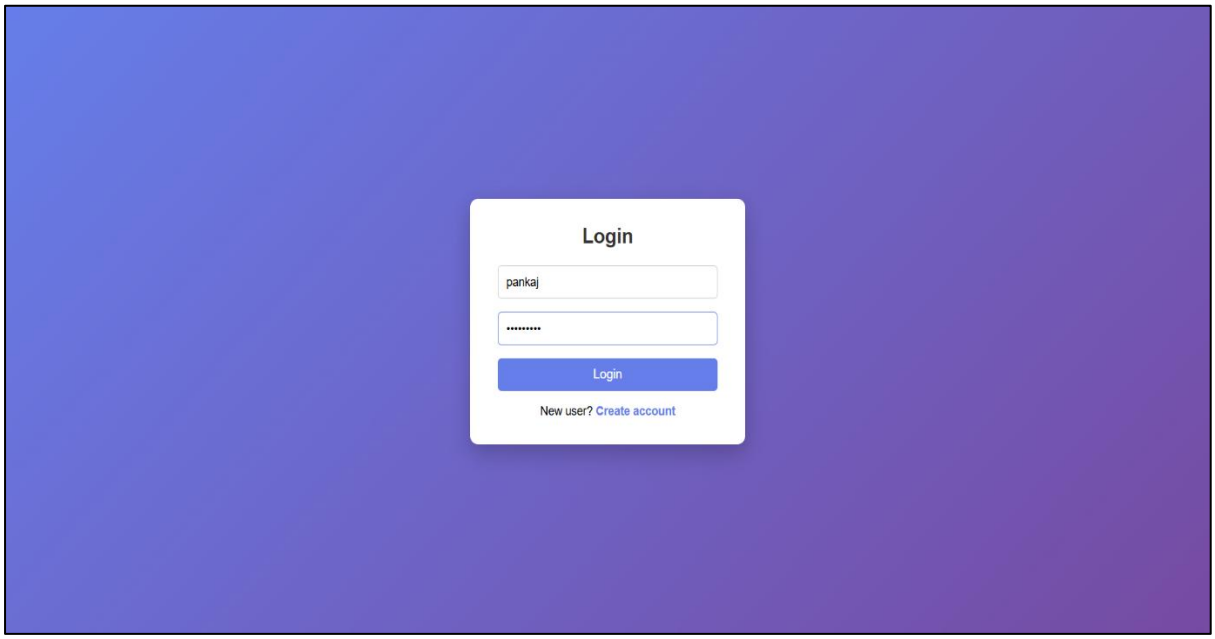


FEEDBACK DATABASE

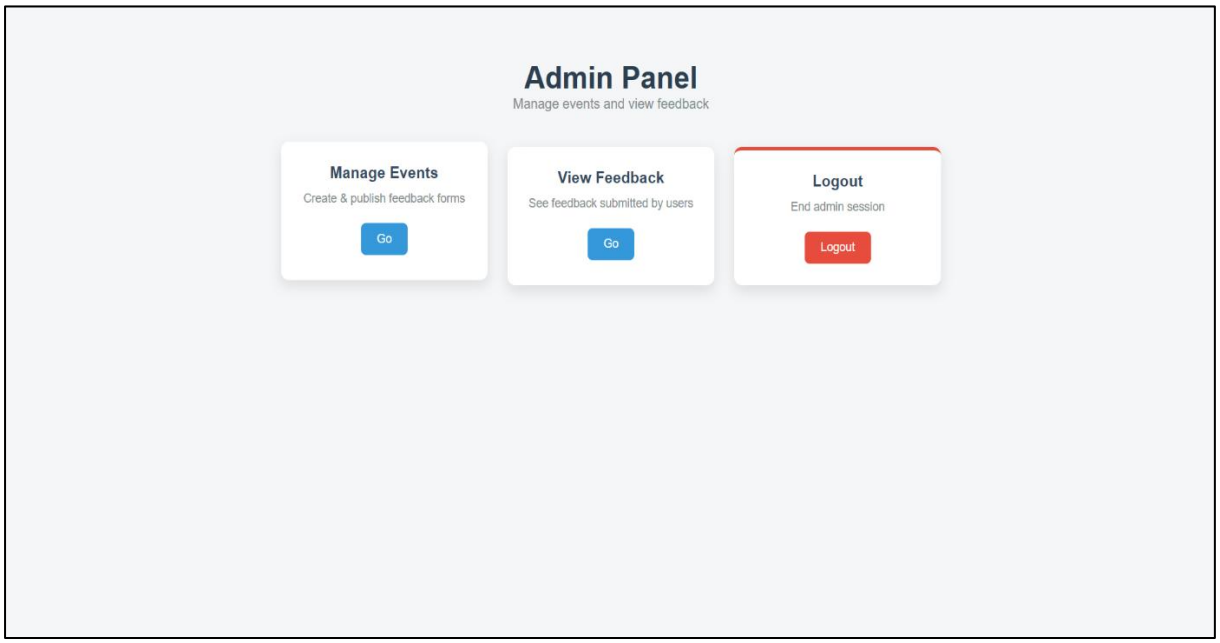


ADMIN OUTPUTS

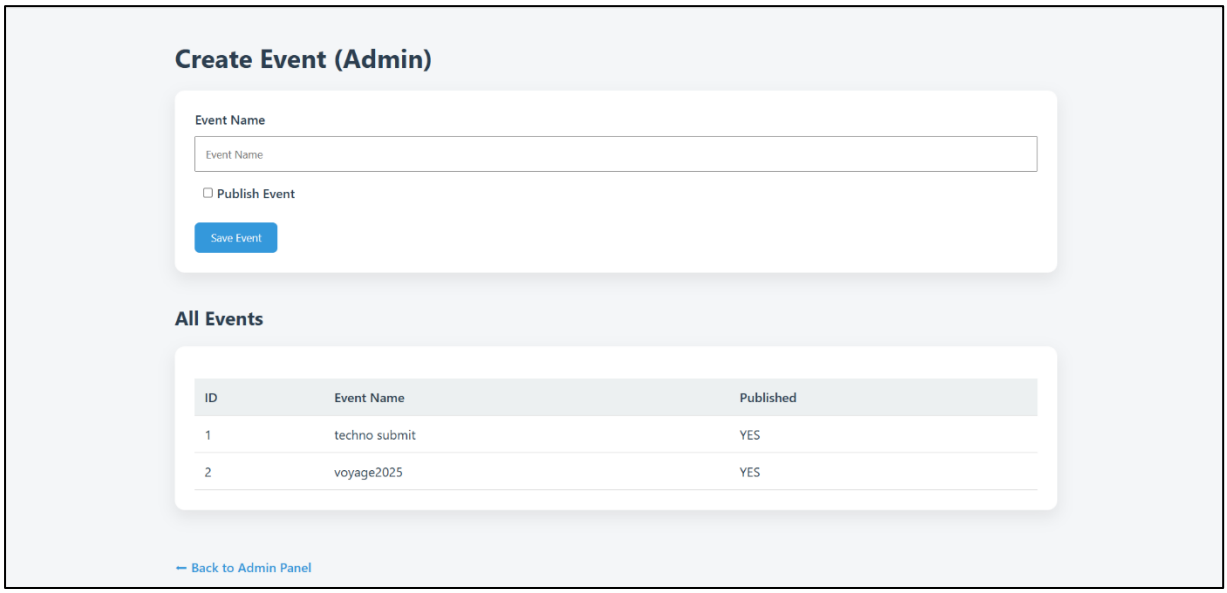
ADMIN LOGIN PAGE



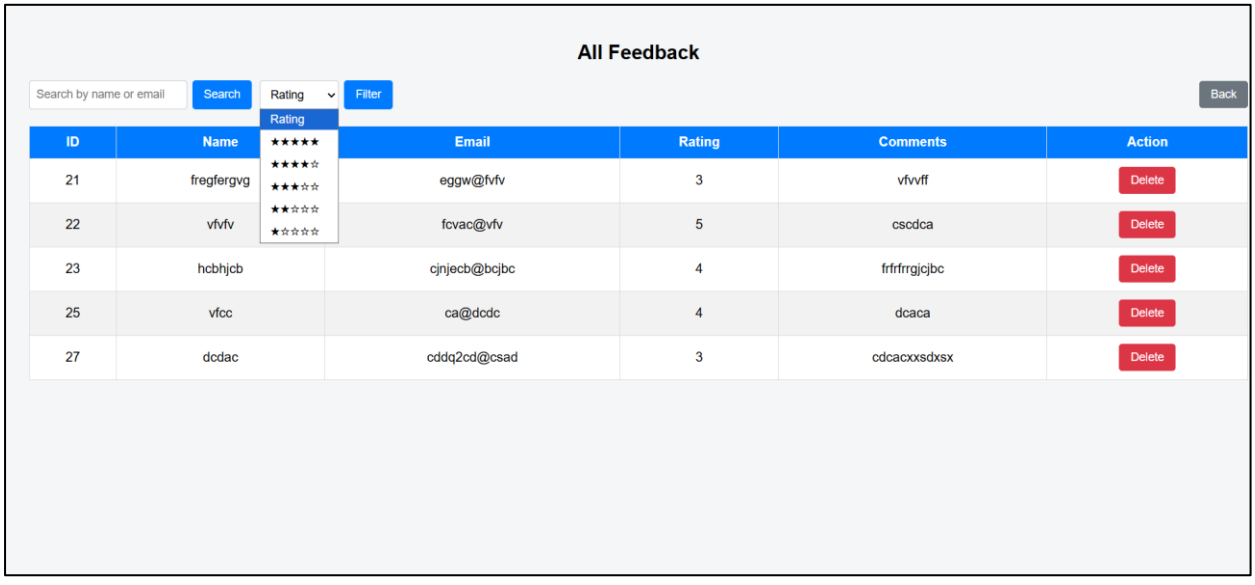
ADMIN PANEL



EVENT CREATION BY ADMIN

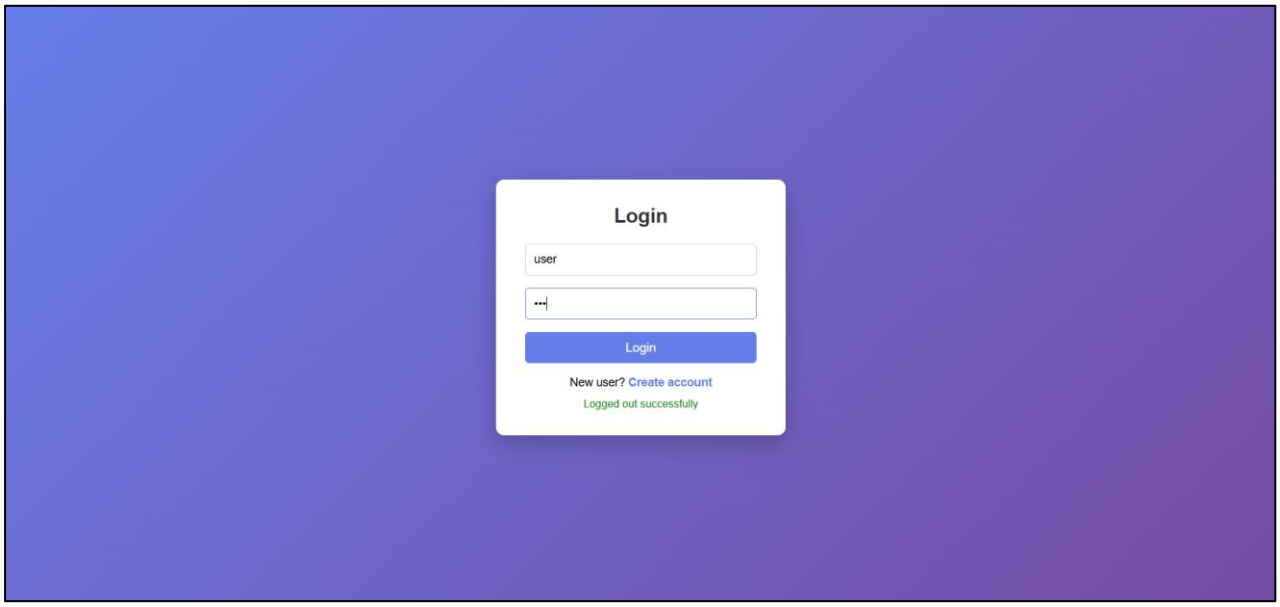


VIEW ALL FEEDBACK

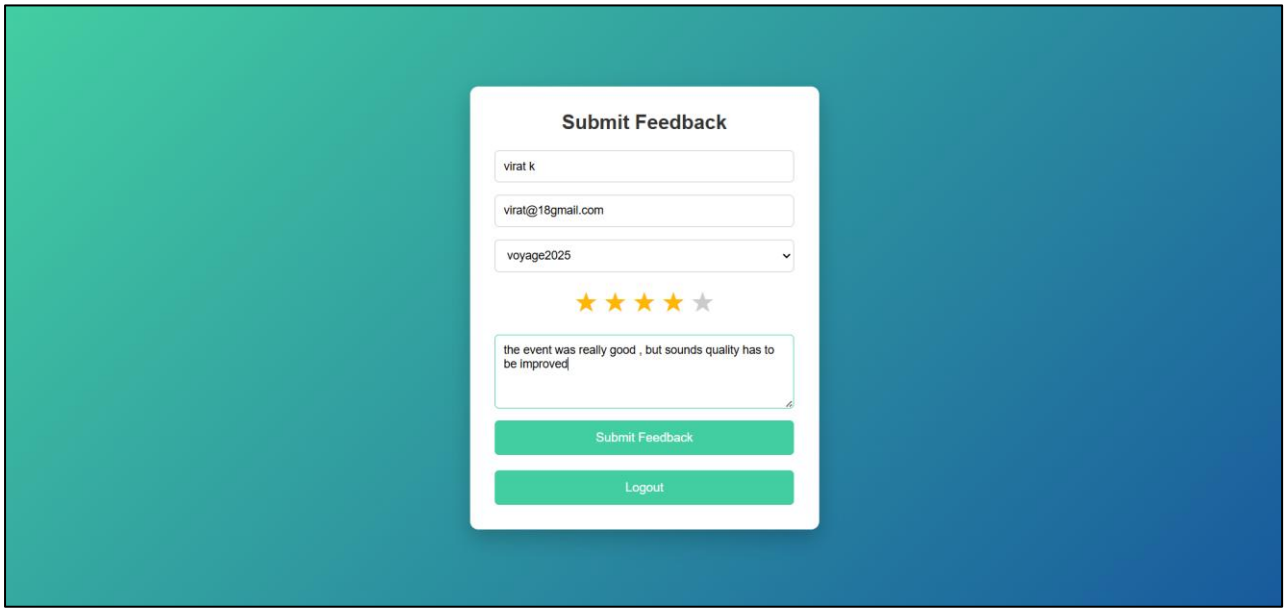


USER OUTPUTS

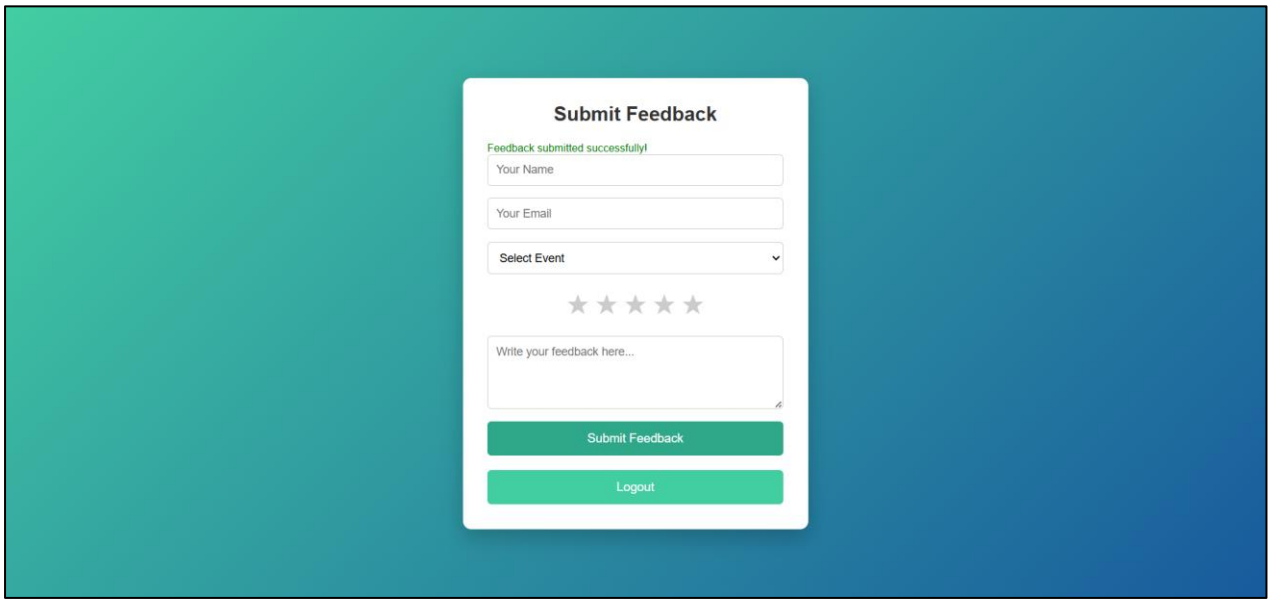
USER LOGIN PAGE



FEEDBACK ENTER INTERFACE



FEEDBACK SUBMITTED SUCCESSFULLY



NEW REGISTRATION INTERFACE