**SmartEventPro – Event Management with AI:**

*A Project Based Learning Report Submitted in partial fulfilment of the requirements for the award of the degree*

*of*

**Bachelor of Technology**

**in The Department of CSE**

**23CS2103E Advanced Object Oriented Programming**

Submitted by

**Roll.no: 2310030090 ,Name: Bhuvan**

**Roll.no: 2310030087,Name: Anas**

**Roll.no: 2310030177,Name:Mahboob**

**Roll.no: 2310030030,Name:Manish**

Under the guidance of

**Anuradha Nandula**



Department of Computer Science Engineering

Koneru Lakshmaiah Education Foundation, Aziz Nagar

Aziz Nagar – 500075

**ABSTRACT:**

This project presents the development of a cutting-edge Event Management System that leverages Artificial Intelligence to streamline and enhance the planning, coordination, and execution of events. Designed for event organizers, venues, and attendees, the platform provides a centralized digital space to manage events efficiently, minimize logistical challenges, and improve user engagement through intelligent features.

The platform addresses common issues in event planning such as resource mismanagement, scheduling conflicts, and poor audience targeting. By integrating AI technologies, the system offers smart recommendations for venue selection, optimal event timing based on historical trends, and personalized content suggestions for targeted user outreach. AI-driven analytics also help organizers understand attendee behaviour, predict attendance rates, and adjust marketing strategies accordingly.

Users can register as Admins, Organizers, or Attendees, each with distinct roles and permissions. Organizers can create and manage events, including uploading event details such as descriptions, dates, images, ticketing options, and category tags. Attendees can browse, search, and filter events based on location, interests, ratings, or popularity. The admin panel supports platform moderation, analytics oversight, and user management to ensure smooth operations.

A robust scheduling assistant uses AI to avoid date clashes and recommends event slots with maximum availability. The system also features a recommendation engine that personalizes event suggestions for attendees based on their browsing history and previous attendance. Fraud detection is employed to monitor suspicious user behaviour, such as abnormal booking patterns or bot-driven registrations.

The frontend is built using React.js, delivering a responsive and dynamic user interface that adapts seamlessly to desktop and mobile environments. The backend is developed using Spring Boot, providing a scalable, modular framework for creating secure RESTful APIs. MongoDB, a flexible NoSQL database, is used for storing unstructured data like user profiles, event details, feedback, and logs. Its document-based model supports fast querying and allows for efficient handling of evolving data structures. Security is maintained using JWT (JSON Web Tokens), ensuring secure, stateless authentication across the platform. This enables reliable role-based access control, allowing each user to perform operations relevant to their role while safeguarding sensitive data.

In summary, this AI-enabled Event Management System not only simplifies event operations but also enhances strategic decision-making through data-driven insights. It empowers organizers to host successful, data-informed events while enriching the experience for attendees. This platform serves as a step toward the intelligent automation of event planning and management in the digital era.

**TABLE OF CONTENTS:**

|  |  |
| --- | --- |
| 1. | Cover page |
| 2. | Abstract |
| 3. | Introduction |
| 4. | Methodology |
| 5. | Experiments |
| 6. | Results |
| 7. | Conclusion and Future Work |
| 8. | References |

# **1.Introduction :**

Event planning and management often involve complex coordination, resource allocation, and audience engagement—challenges that are especially difficult for small to mid-sized organizers without access to advanced tools. To address these inefficiencies, we have developed a web-based AI-powered Event Management Platform that simplifies the event lifecycle from planning to execution while enhancing user experiences through intelligent automation.

This platform enables event organizers to effectively plan, promote, and manage their events while leveraging AI to automate decision-making, improve targeting, and optimize logistics. Attendees benefit from a streamlined event discovery and booking experience, while administrators maintain control over platform integrity and operations.

The core features of the platform include:

* **Event Creation and Listing**: Organizers can upload event details including titles, images, categories, descriptions, locations, ticket types, and scheduling.
* **User Management**: Attendees can register, track their bookings, and receive personalized event recommendations. Organizers and admins manage their respective domains via intuitive dashboards.
* **AI-Powered Scheduling & Recommendations**: Intelligent algorithms help avoid event overlap, suggest optimal timing, and recommend events to users based on their interests and behavior patterns.
* **Secure Booking System**: All transactions are encrypted, with seamless integration to payment gateways ensuring a secure and smooth checkout experience.
* **Fraud Detection Mechanisms**: Monitors for anomalies in user behavior, fake bookings, or suspicious organizer activity, helping maintain a trustworthy environment.
* The platform implements a role-based access control model, supporting three primary roles:
* **Organizers** can create and manage events, track attendee registrations, and access event performance analytics.
* **Attendees** can browse, book, and review events, as well as receive personalized event suggestions.
* **Admins** supervise platform-wide activity, resolve conflicts, and manage flagged users or events.
* Security is enforced through **JWT (JSON Web Token)**-based authentication, ensuring that only authenticated users access their designated features. Combined with granular role-based permissions, this ensures both platform security and data protection.
* From a technical perspective, the architecture consists of:
* **React.js** for the frontend, delivering a responsive and interactive user interface compatible with all major devices.
* **Spring Boot** on the backend, offering a reliable and scalable REST API layer that handles business logic and security.
* **MongoDB**, a schema-less NoSQL database, which is ideal for managing dynamic, unstructured data such as user profiles, event metadata, and interaction logs. Its document-oriented model supports nested documents and high-performance querying, which is crucial for storing reviews, booking histories, and AI-generated analytics.

# **2.METHODOLOGY:**

**a) Selecting the Technology Stack**

To develop a scalable, intelligent, and responsive event management platform, a modern technology stack was selected to ensure fast development, robust performance, and easy maintenance:

* **Frontend: React.js**  
  Enables the creation of dynamic, reusable UI components for an interactive, real-time user experience across various devices.
* **Backend: Spring Boot (Java)**  
  Used to build secure and scalable RESTful APIs with support for modular architecture, authentication, and service orchestration.
* **Database: MongoDB**  
  A document-oriented NoSQL database ideal for managing unstructured and nested data like event details, user activity, and AI insights. It supports fast queries, schema flexibility, and horizontal scalability.
* **Hosting: AWS (Amazon Web Services)**  
  The platform is deployed using AWS services such as EC2 and S3, providing high availability, elasticity, and global accessibility.

**b) System Design**

The application follows a **layered architecture** based on the MVC (Model-View-Controller) pattern to separate concerns and promote scalability:

* **Controller Layer**: Manages incoming HTTP requests and maps them to the appropriate service logic.
* **Service Layer**: Encapsulates business logic including event scheduling, recommendations, and booking workflows.
* **Repository Layer**: Interfaces with MongoDB using Spring Data MongoDB for data persistence and query abstraction.
* **Database Layer**: Stores data in collections with a mix of embedded documents and references to enable efficient querying.
  + **Attendee**: Registers, books events, receives AI-driven recommendations, and submits reviews.

**c) Database Design (MongoDB-Oriented)**

MongoDB stores event and user data in flexible, JSON-like documents that align with the dynamic nature of event management workflows:

* **Collections**:
  + users
  + events
  + bookings
  + reviews
  + ai\_logs (for storing prediction outputs and recommendation data)
* **Relationships**:
  + **One-to-Many (via ObjectId references or embedded documents)**:
    - A user may have multiple bookings referenced in the bookings collection.
    - An event may contain embedded reviews or references to external review documents.
    - AI-generated insights can be linked to both users and events through dedicated logs.
* **AI & Fraud Detection**:
  + Suspicious patterns (e.g., repeated fake bookings or spamming reviews) are identified using **MongoDB aggregation pipelines** and anomaly detection algorithms.

# **3.EXPERIMENTS:**

To validate the core functionalities and reliability of our AI-powered Event Management System, the following experimental implementations and testing phases were conducted:

**a) API Development**

A comprehensive set of RESTful APIs was developed using Spring Boot to handle event-related operations and user management:

* + **GET /events**: Retrieves a list of available events with filters for category, location, and date.
  + **POST /bookings**: Allows attendees to book tickets for selected events.
  + **POST /events**: Enables organizers to create new event listings.
  + **POST /register**: Registers new users with securely hashed credentials.
  + **POST /auth**: Authenticates users and issues JWT tokens for secure session management.  
    These APIs interact with **MongoDB** via **Spring Data MongoDB**, enabling seamless CRUD operations without the complexity of SQL joins.

**b) MongoDB CRUD Operations**

CRUD functionalities were implemented through MongoRepository interfaces, offering high-level abstraction and efficient data access:

* + **Create**: New documents (e.g., events, users, bookings) are stored using save().
  + **Read**: Data retrieval is handled using method-named queries or custom @Query annotations to fetch complex, nested structures like event bookings or user preferences.
  + **Update**: Partial updates and field-level modifications are supported using Update objects.
  + **Delete**: Documents are deleted using IDs or condition-based filters.  
    MongoDB’s schema-less structure allowed for embedding reviews, user history, and AI-generated recommendations within related documents, improving performance and reducing query overhead.

**c) Frontend-Backend Integration**

* + Utilized **Axios** in **React.js** to perform asynchronous HTTP requests to backend APIs.
  + JSON responses from Spring Boot were parsed and rendered dynamically in reusable React components (e.g., event cards, booking forms, dashboards).
  + State management was handled using React hooks and context to ensure seamless UI updates in real time.

**d) Testing**

**Backend Testing**:

* + **Unit tests** were written using **JUnit** for core service logic and utility classes.
  + **Integration tests** employed **embedded MongoDB** to simulate real data operations in a controlled test environment.
  + **API Testing**:
  + Conducted using **Postman**, validating endpoint responses, authentication workflows, input validation, and error handling.
  + **Frontend Testing**:
  + Manual testing was performed for key workflows including user registration, login, event browsing, booking, and organizer/admin features.
  + UI responsiveness and data synchronization were verified across multiple device types and browsers.

# **4.RESULTS:**

The developed AI-powered Event Management Platform successfully achieved its primary goals—delivering a secure, intelligent, and user-centric environment for event organizers, attendees, and administrators.

🚨 **AI-Driven Fraud Detection & Anomaly Monitoring**

* AI models and **MongoDB aggregation pipelines** were used to monitor suspicious user behavior, such as bulk fake bookings or review spamming.
* Events showing abnormal activity patterns (e.g., excessive last-minute cancellations or consistently poor feedback) were automatically flagged for admin intervention, enhancing platform trust and reliability.

🔐 **Secure Authentication & Role-Based Access**

* Implemented **JWT-based authentication** for secure, stateless user sessions.
* All protected routes required token verification, enforcing strict access control across admin, organizer, and attendee roles to prevent unauthorized actions.

✅ **Deployment Success**

* The system was successfully deployed using **Amazon Web Services (AWS)**:
  + **React.js frontend** was hosted on **Amazon S3 or EC2** for responsive access across regions.
  + **Spring Boot backend** was deployed on **EC2**, connected to **MongoDB Atlas** for secure and scalable data storage.

🚀 **Performance Optimizations**

* Applied **MongoDB indexing** on frequently queried fields such as event.\_id, user.\_id, and booking.userId to boost query performance.
* Enabled **pagination** using MongoDB’s skip and limit functions to efficiently handle event listings and browsing.
* Static assets like images and documents were served via a **CDN (Content Delivery Network)** to ensure fast loading times globally.

**👥 User Experience Highlights**

**Attendees Were Able To:**

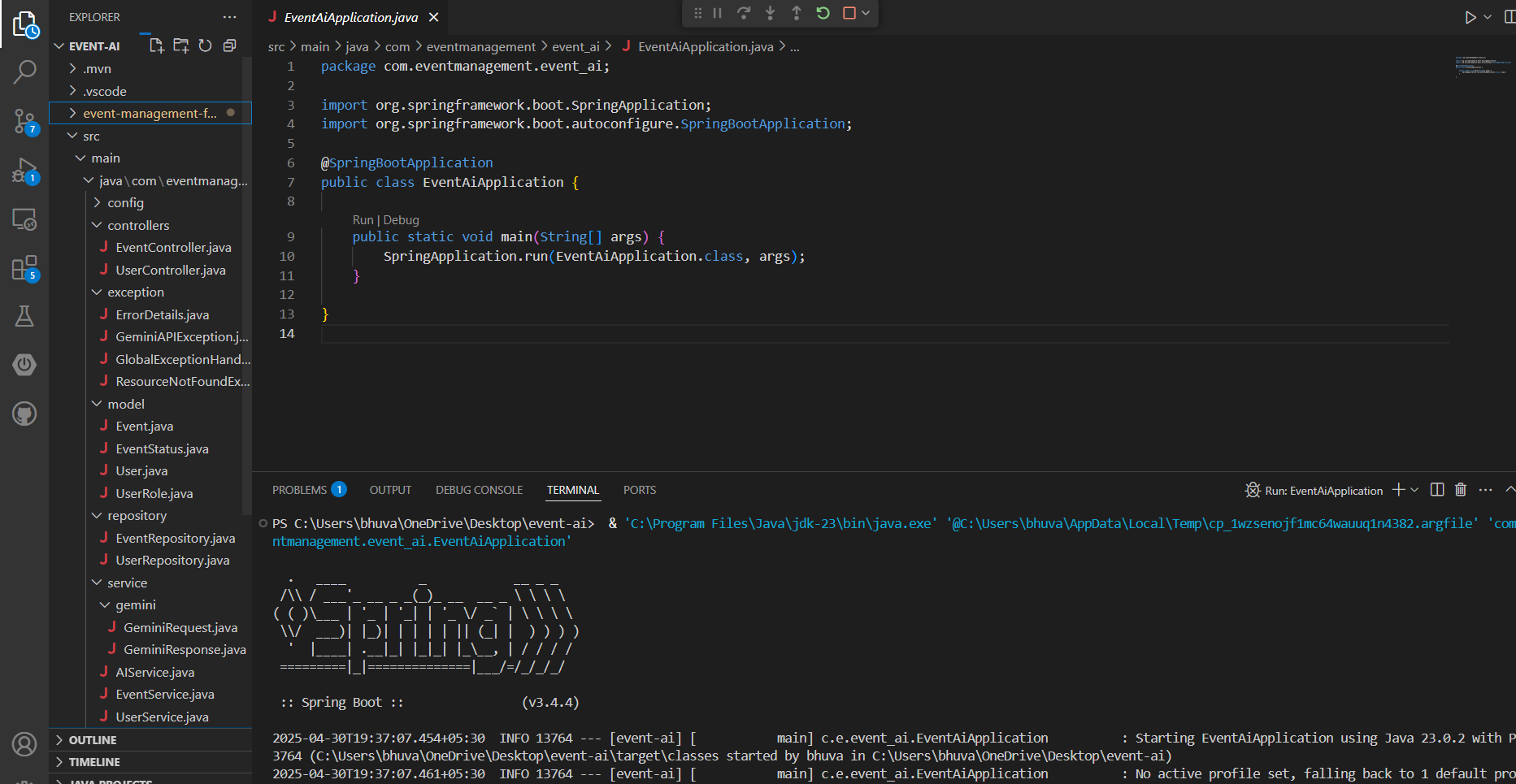
* Browse and filter available events by category, location, or date.
* Register, log in, and receive personalized event recommendations using AI.
* Book tickets and view their booking history through a user-friendly interface.

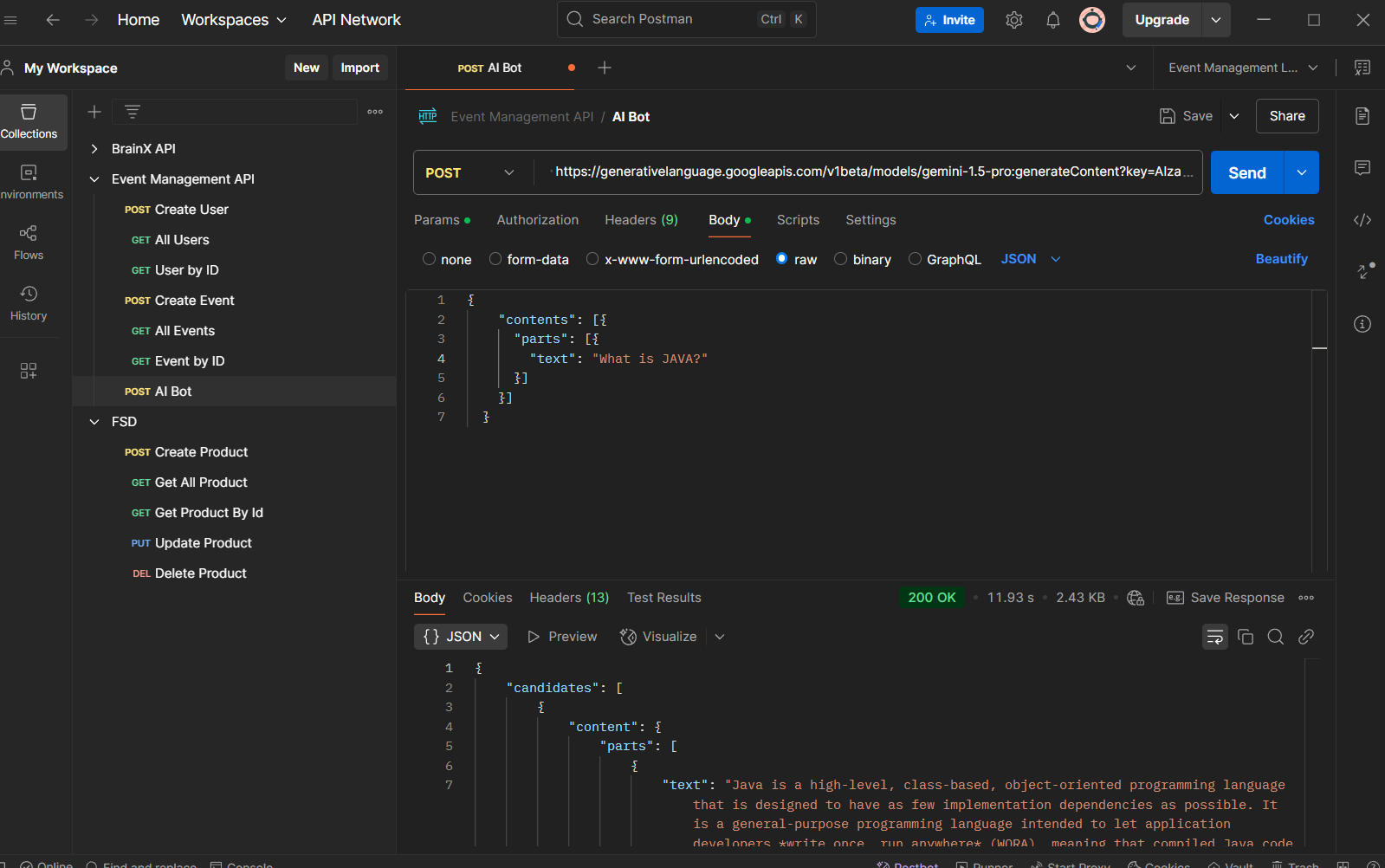
**Organizers Could:**

* Create, edit, and publish event listings with rich content and media.
* Track bookings, monitor event performance, and receive feedback analytics.

**Admins Were Empowered To:**

* Oversee flagged events or accounts through fraud detection alerts.
* Manage user roles, access permissions, and system-level configurations.
* Maintain platform health by monitoring activity logs and usage metrics.





# **5.CONCLUSION :**

In this project, we successfully developed a scalable and secure **AI-powered Event Management Platform** designed to streamline the event planning process for organizers and enhance the experience for attendees. The platform was built using a modern technology stack: **React.js** for the frontend, **Spring Boot** for backend service orchestration, and **MongoDB** as the core database solution.

The shift from traditional relational databases to **MongoDB** offered several key advantages. MongoDB’s flexible, schema-less, document-oriented data model allowed for the efficient storage of event details, attendee data, and user-generated content such as reviews, bookings, and AI-driven recommendations. This flexibility accelerated development, facilitated data scaling, and supported the complex, unstructured data involved in event management.

Security was a top priority in the platform's design. **JWT-based authentication** was implemented to ensure safe, stateless access control, with role-specific permissions for **Admins**, **Organizers**, and **Attendees**. This role-based access control system contributed to both the security and the integrity of user data. MongoDB’s integration with **Spring Data MongoDB** further simplified data interactions, removing the need for complex SQL queries and joins.

Beyond the technical features, the project has a significant social impact, empowering event organizers and attendees through a seamless, digital experience, and providing greater access to live events, both locally and globally.

**Future Work:**

To further enhance the platform's functionality and expand its reach, the following improvements are proposed:

* **Mobile Application Development**: Building a cross-platform app using **React Native** to offer mobile-first accessibility and a smoother user experience.
* **AI-Driven Event Recommendations**: Implementing machine learning algorithms to analyze user behavior and offer personalized event suggestions, increasing attendee engagement and event conversions.
* **Integration with Third-Party Ticketing & Payment APIs**: Adding support for external ticketing platforms and payment processors for seamless ticket purchases and event management.
* **Advanced Admin Dashboard**: Creating an enhanced dashboard with real-time analytics, fraud monitoring, and AI-driven insights using **MongoDB aggregation pipelines** and advanced visualization tools.

##### **6.References:**

 React.js Documentation: <https://react.dev/>

 Spring Boot Documentation: <https://spring.io/projects/spring-boot>

 MySQL Documentation: <https://dev.mysql.com/doc/>

 JWT Authentication: <https://jwt.io/introduction>

* https://www.deepseek.com/en