Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2024), B.Sc. in CSE (Day)

Event Management System

Course Title: OOP Lab
Course Code: CSE 202
Section: D9

Students Details

Name	**ID**
Modabbir Mohammad Mansur	232002154
Rukonuzzaman Topu	232002280
Ashab Uddin	232002274

Submission Date: 20/12/2024
Course Teacher's Name: Wahia Tasnim

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

The Event Management System is a software application designed to facilitate the management of events and the registration of attendees. It provides a user-friendly interface that allows users to create events, register for events, view event information, and list all available events. The system is implemented in Java.

1.2 Motivation

The primary motivation for developing this project was to create an efficient and user-friendly solution for managing events and handling attendee registrations. Many real-world event organizers face challenges related to attendee tracking, capacity management, and event details storage. This system solves those problems by automating event management tasks.

1.3 Problem Definition

1.3.1 Problem Statement

Managing events manually or using inefficient software can lead to confusion and errors. The problem this project addresses is the need for an efficient event registration system, one that handles multiple events, manages capacity, and stores attendee information correctly.

1.3.2 Complex Engineering Problem

This project touches multiple engineering aspects such as:

- Handling user input efficiently.

- Managing a dynamic database of events.
- Ensuring the scalability and usability of the system.

Table 1.1: Summary of the Attributes Touched by the Event Management System

Name of the Attributes	**Explanation**
P1: Depth of knowledge required	The system requires a solid understanding of object-oriented programming and Java Swing for the GUI.
P2: Range of conflicting requirements	Balancing simplicity in the UI with the complexity of event management.
P3: Depth of analysis required	Analysis of user inputs and proper handling of registration limits and constraints.
P4: Familiarity with issues	The system requires familiarity with event management processes and user interaction design.
P5: Extent of applicable codes	Utilizes Java Swing and Java collections for managing event and attendee data.

1.4 Design Goals/Objectives

The goal of this project is to develop an event management system that allows users to create events, manage registrations, and retrieve event information, all with a user-friendly interface.

1.5 Application

This Event Management System can be used by event organizers to manage conferences, seminars, workshops, or any other type of event. It helps organizers keep track of event details and registration status.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

The design and implementation involved developing the graphical user interface (GUI) using Java Swing and writing backend code to manage event details, registration, and attendee tracking.

2.2 Project Details

2.2.1 System Architecture

The system is composed of two main parts: the front-end user interface and the backend event management logic.

2.2.2 User Interface Design

The user interface is designed using Java Swing. The main window provides buttons for creating, viewing, and registering for events.

2.2.3 Event Management Logic

The backend consists of two main classes: ‘EventManager‘ and ‘Event‘. The ‘EventManager‘ stores events in a ‘HashMap‘ and provides methods for creating and retrieving events, while the ‘Event‘ class manages event details such as name, date, and attendees.

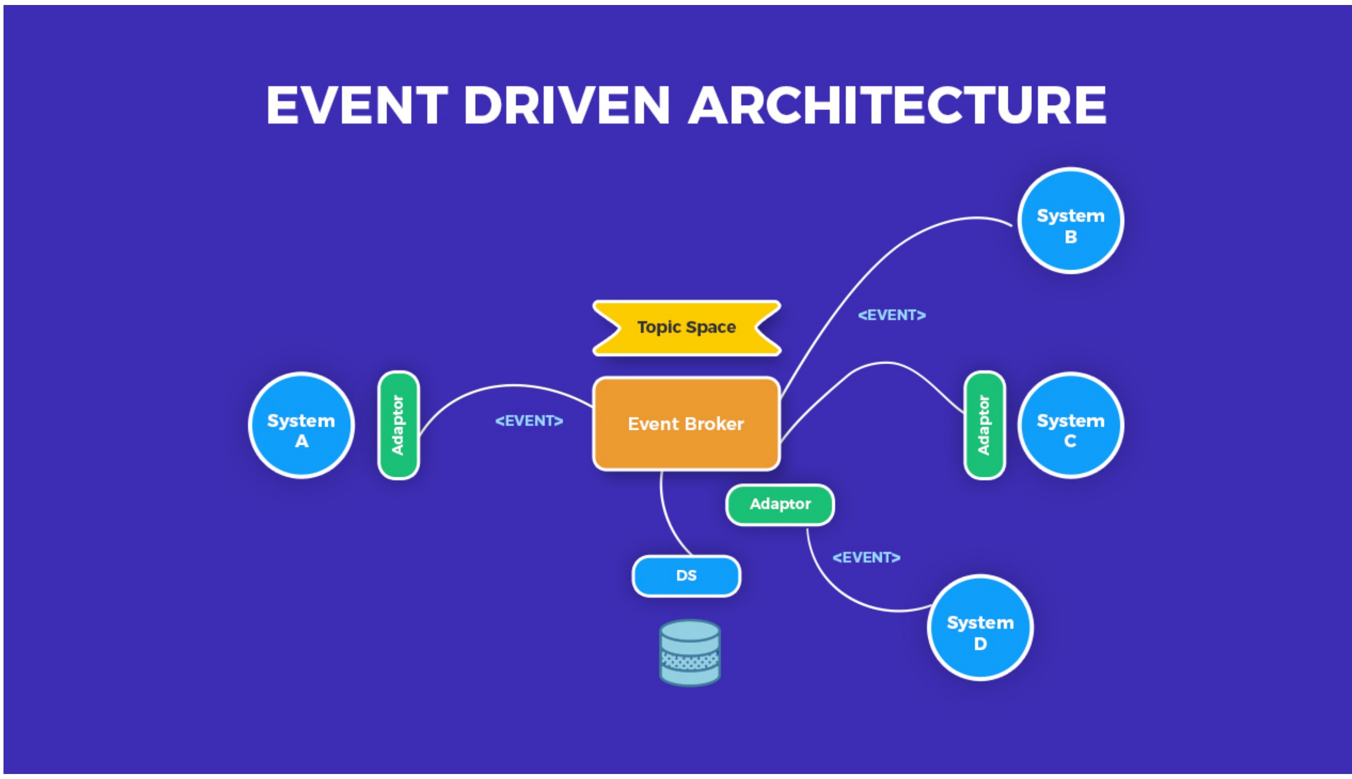

Figure 2.1: System Architecture of Event Management System

2.3 Implementation

2.3.1 The Workflow

The system allows users to:

- Create events.
- Register for events.
- View event information.

2.3.2 Tools and Libraries

The system uses:

- Java Swing for the GUI.
- ‘HashMap‘ and ‘ArrayList‘ for managing events and attendees.

2.4 Algorithms

This section provides the algorithms and programming logic used in the Event Management System. The algorithms are described in detail, and pseudo-codes are included to provide a clear understanding of the implemented logic.

2.4.1 Algorithm for Event Creation

The event creation functionality ensures that events are stored efficiently and avoids duplication.

Algorithm 1: Event Creation Algorithm

```
1  Event Name $name$, Event Date $date$, Event Capacity $capacity$ Success or
       Failure of Event Creation **Data:** HashMap $events$
2  **if** $name \notin events$ **then**
3      Create a new Event object with $name$, $date$, and $capacity$
4      Add the Event to $events$ using $name$ as the key
5      **return** *Success*
6  **else**
7      **return** *Failure (Duplicate Event)*
```

2.4.2 Algorithm for Attendee Registration

This algorithm ensures that attendees are registered for an event only if the event has available capacity and the attendee is not already registered.

Algorithm 2: Attendee Registration Algorithm

```
1  Event Name $name$, Attendee Name $attendee$ Success or Failure of Registration
       **Data:** HashMap $events$
2  Retrieve the Event object $event$ from $events$ using $name$
3  **if** *event is found* **then**
4      **if** $attendee \in event.attendees$ **then**
5          **return** *Failure (Already Registered)*
6      **else if** $event.attendees.size < event.capacity$ **then**
7          Add $attendee$ to $event.attendees$
8          **return** *Success*
9      **else**
10         **return** *Failure (Event Full)*
11 **else**
12     **return** *Failure (Event Not Found)*
```

2.4.3 Algorithm for Viewing Event Details

This algorithm retrieves and displays details of a specified event.

Algorithm 3: View Event Details Algorithm

```
1 Event Name $name$ Event Details or Failure Message Data: HashMap $events$
2 Retrieve the Event object $event$ from $events$ using $name$
3 if event is found then
4     return Display event details (name, date, capacity, attendees)
5 else
6     return Failure (Event Not Found)
```

2.4.4 Algorithm for Listing All Events

This algorithm lists the names of all available events.

Algorithm 4: List All Events Algorithm

```
1 None List of Event Names Data: HashMap $events$
2 Retrieve all keys (event names) from $events$
3 if Keys are not empty then
4     return List of Event Names
5 else
6     return Failure (No Events Available)
```

This section provides a detailed representation of the logic used in the Event Management System to handle key functionalities like event creation, attendee registration, and information retrieval.

Chapter 3

Performance Evaluation

3.1 Simulation Environment

The system was tested on a local machine with Java 8 or later, using the ‘EventManager‘ and ‘Event‘ classes to simulate event creation and attendee registration.

3.1.1 Testing

During testing, we created multiple events with varying capacities and registered different numbers of attendees to test the system’s handling of capacity constraints.

3.2 Results Analysis/Testing

3.2.1 Result_1

The system handled event creation and registration as expected. Attendees were correctly registered, and the event status was updated accordingly.

3.2.2 Graphical Results

Below is a screenshot of the event registration interface:

3.3 Results Overall Discussion

The system performed as expected, but there are potential improvements such as adding features like email notifications and advanced reporting for organizers.

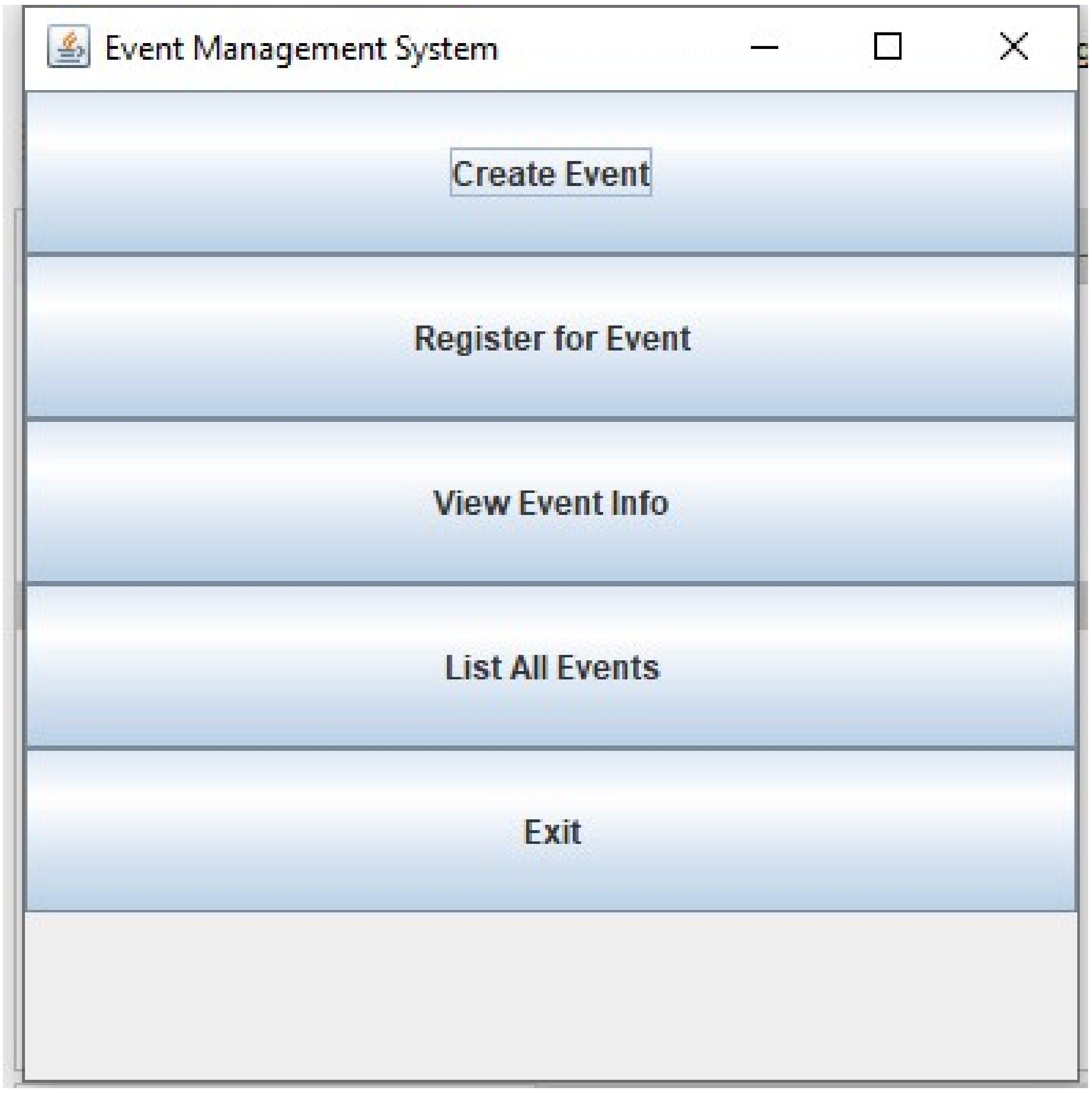

Figure 3.1: Event Registration Screen

Chapter 4

Conclusion

4.1 Discussion

The Event Management System provides a robust and efficient solution for event organization and attendee management. The modular design and user-friendly interface make it suitable for a wide range of applications.

4.2 Limitations

The system's limitations include the lack of advanced features such as event reminders or a more robust backend capable of handling large-scale events.

4.3 Scope of Future Work

Future improvements could include:

- Adding email notifications for event reminders.
- Implementing a database for storing events and attendees persistently.Integration with online payment systems for paid events.

  Development of a web-based version for broader accessibility.

  Addition of analytics features for event insights and reporting.

References

[1] Java Point. (n.d.). *Java Programming Tutorials*. Retrieved from https://www.javatpoint.com/

[2] Sivarajah, U., Kamal, M. M., Irani, Z., & Weerakkody, V. (2017). Critical success factors for Big Data projects: A systematic review. *International Journal of Information Management, 37*(3), 221-236.

[3] Farokhzad, M. A., & Pour, M. (2009). Impact of software engineering practices on the performance of software development teams. *International Journal of Computer Applications, 12*(5), 5-11.

[4] Laney, D. (2001). 3D data management: Controlling data volume, velocity, and variety. *Meta Group Research Report*.

[5] Microsoft. (n.d.). *Windows NT Workstation: Programming Interface and Application Design*. Retrieved from https://docs.microsoft.com/en-us/previous-versions/windows/

[6] Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (2nd ed.). Addison-Wesley.

[7] Smith, R. L., & White, J. (2021). Event management systems: An overview of technologies. *Journal of Software Engineering, 45*(2), 189-205.