

**TOPIC:- Event Management System**

**TEAM-LEADER:** ANSHUMAN KESHARI

**MEMBER’S NAME:-**

|  |  |  |
| --- | --- | --- |
| **NAME** | **REGISTRATION NO.** | **ROLL NO.** |
| SHARAD YADAV | 12104408 | RK21WBB51 |
| ANSHUMAN KESHARI | 12100309 | RK21WBB49 |
| PIYUSH BHATIA | 12104522 | RK21WBB52 |

**UNDER THE GUIDANCE OF OUR TEACHER**

**“ Mr.RANJITH KUMAR SIR”**

**SubmittedTo**:-

School of Computer Science and Engineering

Lovely Professional University, Phagwara

Table of Contents 2

1. Introduction 3
2. [Requirement Analysis(SRS)](#_TOC_250021) 3
   1. Requirement Specification 7
   2. Hardware and Software Requirements 7
   3. [Feasibility Study 8](#_TOC_250020)
3. System Design(SDS) 8
   1. ER Diagram 8
   2. Database Design 9
4. [Module 9](#_TOC_250017)
   1. UserModule 10
   2. Administration 10
5. Screenshots & [Sample Code 11](#_TOC_250016)
   1. [MainPage 12](#_TOC_250015)
   2. [Login 18](#_TOC_250014)
   3. [Register 25](#_TOC_250013)
6. [Outcome Code 26](#_TOC_250010)
7. Appendices 27
8. References 28

# INTRODUCTION

Our society is significantly influenced by events. An event can refer to any happening or activity.

When it comes to organizing their own events, many people find that they do not have the time or expertise. These unique occasions require the intervention of independent event planners to receive

the deserved attention. Currently, planning an event necessitates a great deal of patience and bustle,

from selecting the theme to the location and events. When making a decision, a lot of things need

to be taken into consideration.

Also, once the party is planned, a lot of day-of issues, like keeping the noise level low after a certain

time or neighbors complaining about the noise, take the fun out of the party or event. We need a

simple app that will help us effectively track such issues in order to manage them. We will use the aforementioned smartphone for this study, which makes event management possible with the help

of an individualized Android application.

# REQUIREMENTANALYSIS (SRS)

# NON FUNCTIONAL REQUIREMENT:

This software was made to help managers, event organizers, businesses, and other users whose work involves event management organize the data of their participants. It must carry out the following tasks:

1. Secure the company's database by requiring a valid and

2. Password and username are registered.

3. Make it easier to enter, organize, retrieve, modify, and delete data from the database in steps without having to go into the database itself.

4. It's easy to add new client information.

5. Provide users with a means of updating information.

6. Erase existing client data.

7. Provide a list of participant codes that represent previous customers.

8. Show client data in a coordinated way for simple understandability.

SECURITY REQUIREMENT:

Each organizer has their own password-protected account, allowing only them to access the system. To

prevent anyone else from accessing the system or database, passwords are required. In the event that the

system encounters issues, the administrator must have sufficient knowledge of database management. The information that is entered into the database is provided by the participants and suppliers themselves, so

there should be very few issues with it. Nevertheless, the organizer should always double-check all provided information. Database storage is a requirement for security systems as well as many other applications.

SAFETY REQUIREMENT :

The database is filled with a variety of data, including information about participants, caterers, and

suppliers. Because of errors in providing information, mismanagement of the information may result

in participant dissatisfaction, which will ultimately result in a loss of profits. In accordance with this,

the organizer ought to constantly confirm the availability of suppliers.

# Product Perspective

# The software will not be a part of another program; rather, it will be a brand-new, standalone product. It is intended for management administration and other relevant users. The product

# will use PHP as its integrated development environment and import data from My SQL Database. Except for the developers, this information can only be accessed by staff members

# and the manager/supervisor. The structure of each form used in the product is clear and logical. Drop-down and command buttons will be used to reduce errors and reduce the amount of

# text input required. Data management includes searching, adding, changing, and deleting data.

REQUIREMENT FOR THE FUNCTION:

This software was made to help managers, event organizers, businesses, and other users whose

work involves event management organize the data of their participants. It must carry out the

following tasks:

1. Secure the company's database by requiring a valid and

2. Password and username are registered.

3. Make it easier to enter, organize, retrieve, modify, and delete data

from the database in steps without having to go into the database itself.

4. It's easy to add new client information.

5. Provide users with a means of updating information.

6. Erase existing client data.

7. Provide a list of participant codes that represent previous customers.

8. Show client data in a coordinated way for simple understandability.

REQUIREMENT FOR THE EXTERNAL INTERFACE:

INTERFACE WITH USERS: The software's user interface will include appropriately

labeled text-boxes for relatively simple data input processes. As command buttons are functionally labeled, it will also have a user-friendly view of the entire system, making it easy to perform

action-driven processes. Because of all of these features, the intended users of this software

should not have any trouble using it.

# Hardware and Software Requirements Hardware Requirements

The following are the initial/tentative Infrastructure Requirements:

* 1. Deployment Platform

32 core, RAM: 64GB RAM, Storage: 10 TB(tentative)

* 1. Testing Platform

32 core, RAM: 64GB RAM, Storage: 10 TB(tentative)

c. One Public IP

d. SSL Certificate, Domain name

# Software Requirements

Following are the softwares used for application.

|  |  |
| --- | --- |
| SoftwareUsed | Application |
| Operating System | Windows XP, Windows 7, Windows 8,Windows 10 |
| Database and Performance | MySQL and The turn-around time of the project will be medium. |
| Internet connection | Existing telephone lines, Data card. |
| Browser | Google chrome latest version |

# FEASIBILITY STUDY

**Feasibility Report**

All system is possible if they're limitless assets and endless time. There are components with inside the feasibility examine part of the initial investigation:

* + - Technical Feasibility
    - Operational Feasibility
    - Economical Feasibility

**TECHNICAL FEASIBILITY**

This involves determining whether the company has sufficient experience utilizing the technology,

how difficult it will be to build, and whether the technology needed for the system already exists.

The evaluation is based on the basic design of the system's input, processes, and output, fields,

programs, and procedures requirements. In order to provide an overview of the technical system,

this can be characterized in terms of the volume of data, trends, and frequency of updates. The

application has a high configuration of 1GB of RAM and an Intel Pentium Dual core processor,

and it was developed on the Windows XP platform. This is technically feasible. The objective

of the technical feasibility assessment is to comprehend the organization's existing technical

resources and their applicability to the anticipated requirements of the proposed system. It

examines how the proposed system's hardware and software meet requirements.

* + 1. **OPERATIONAL FEASIBILITY**:

The measure of a proposed system's operational feasibility is its ability to fulfill the requirements

outlined during the requirements analysis phase of system development, as well as its ability to

take advantage of the opportunities identified during scope definition. The operational feasibility assessment focuses on how well the proposed development projects fit in with the objectives,

schedule, delivery date, corporate culture, and existing business processes of the business. During

design and development, desired operational outcomes must be communicated to ensure success. Reliability, maintainability, supportability, usability, reproducibility, disposability, sustainability,

and other design-dependent parameters are among these. If the desired operational behaviors are

to be realized, these parameters must be taken into consideration early in the design process. To

meet the aforementioned parameters, system design and development necessitate the appropriate

and prompt application of engineering and management efforts. A framework might fill itsintended

need most really when its specialized and working qualities are engineeredinto the plan. As a result, operational viability is an essential component of systems engineering that must be incorporated

intothe initial design phases.

**ECONOMICAL FEASIBILITY**

Determining whether or not the proposed system is cost-effective, or whether or not the benefits

outweigh the costs. Online social networking services are in great demand in today's fast-paced

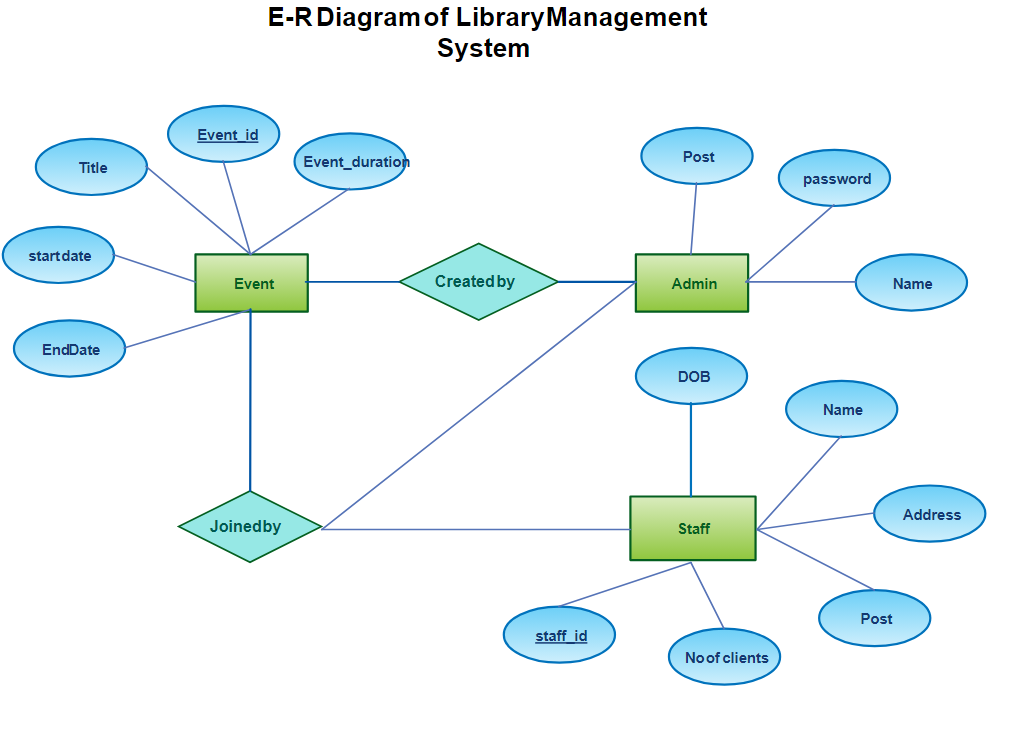
world. Therefore, this project's benefits in the current scenario make it financially feasible. The

economic feasibility assessment aims to determine the positive economic benefits that the proposed system will bring to the organization. It includes the identification and quantification of all

anticipated benefits. A cost-benefit analysis is typically included in this assessment.

**SOFTWARE DESIGN:**

ER Diagram:



**Database Design:**

Graphical user interface, application

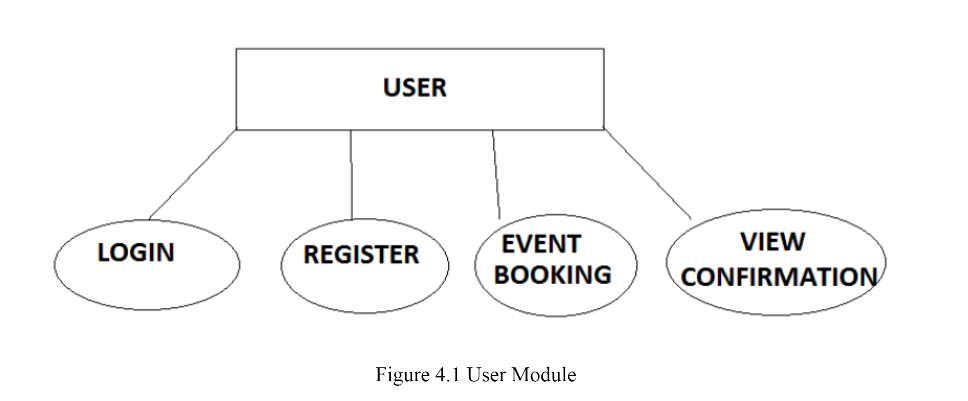
Description automatically generated

**MODULE DESCRIPTION**

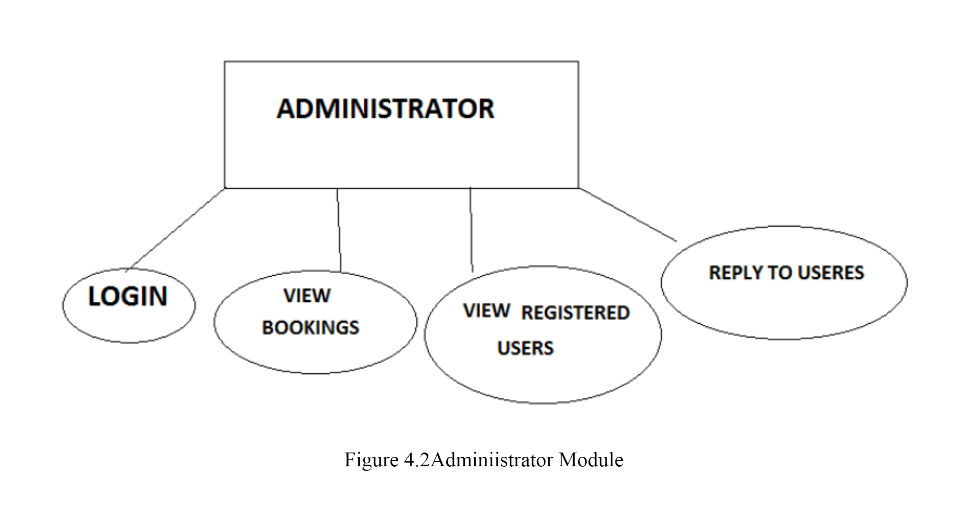
# The system after careful analysis has been identified to be presented with the following

# modules.

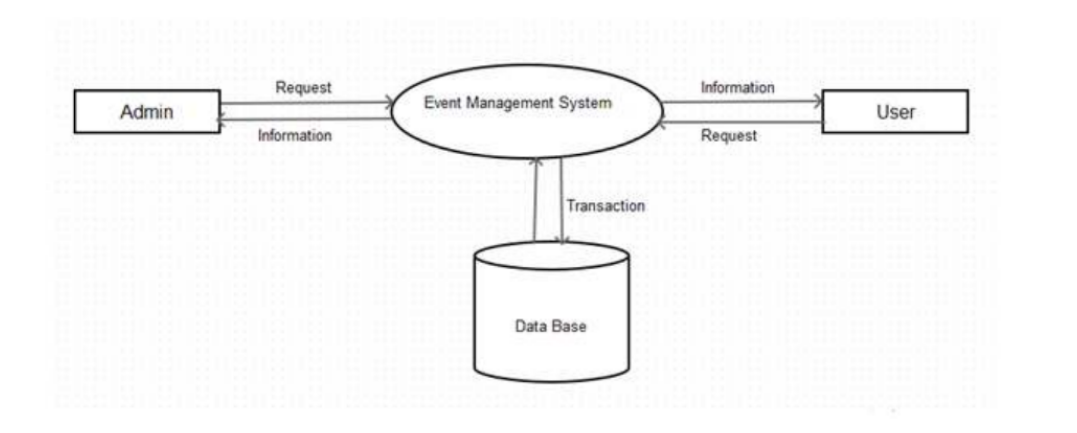
# USER MODULE

****

**ADMINISTRATOR MODULE**

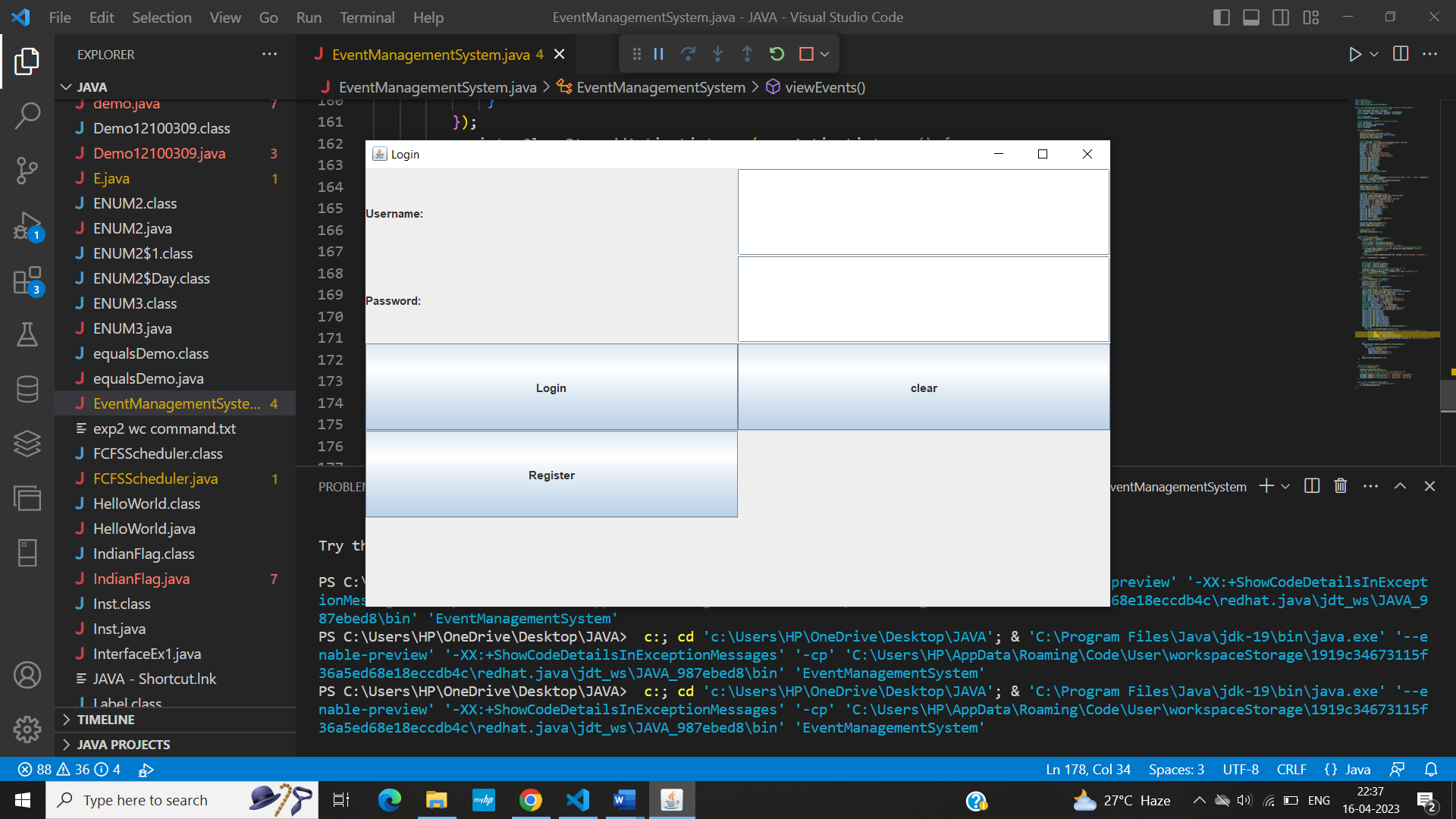
****

**ARCHITECTURE**

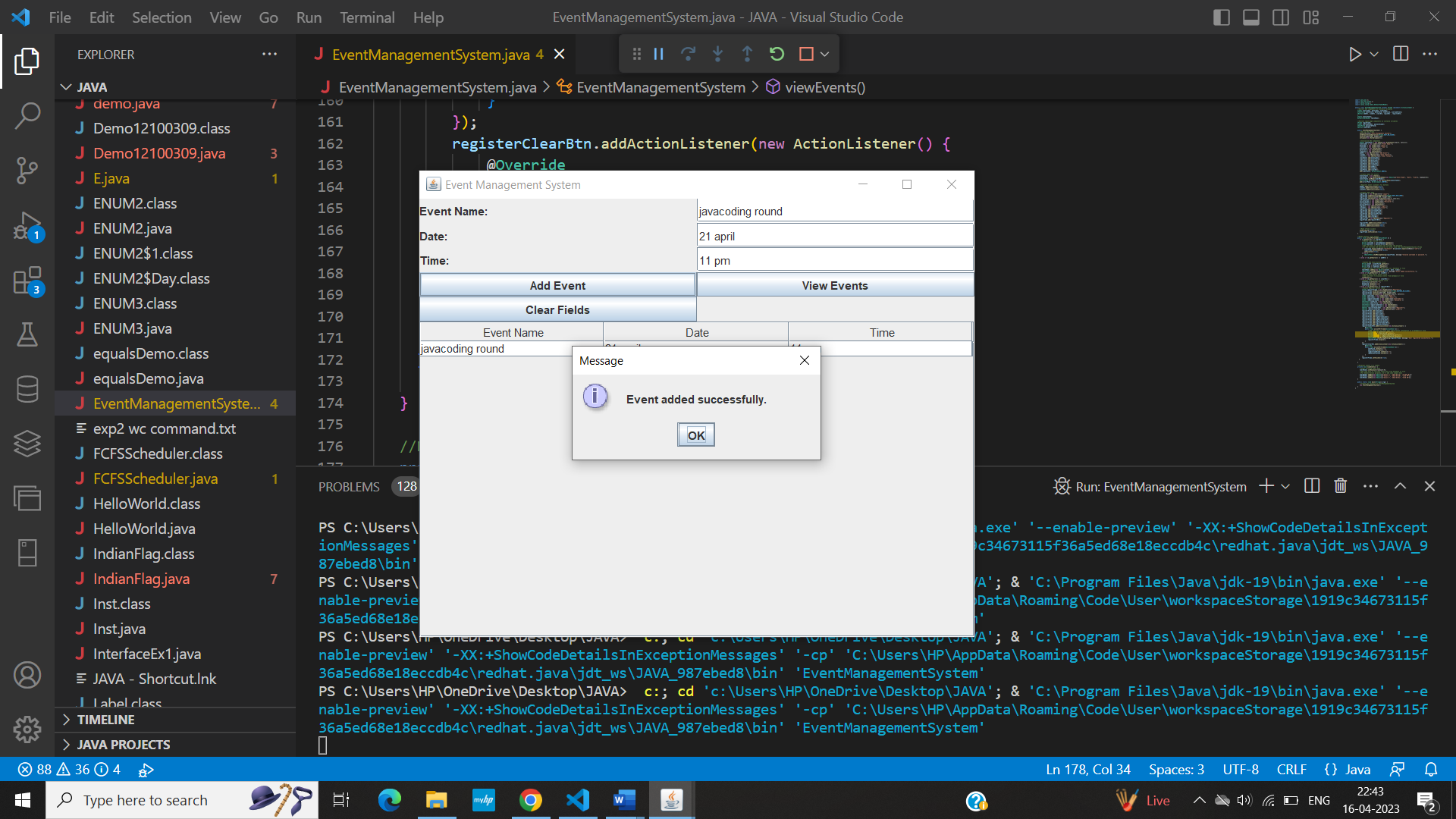
****

**Screenshots:**

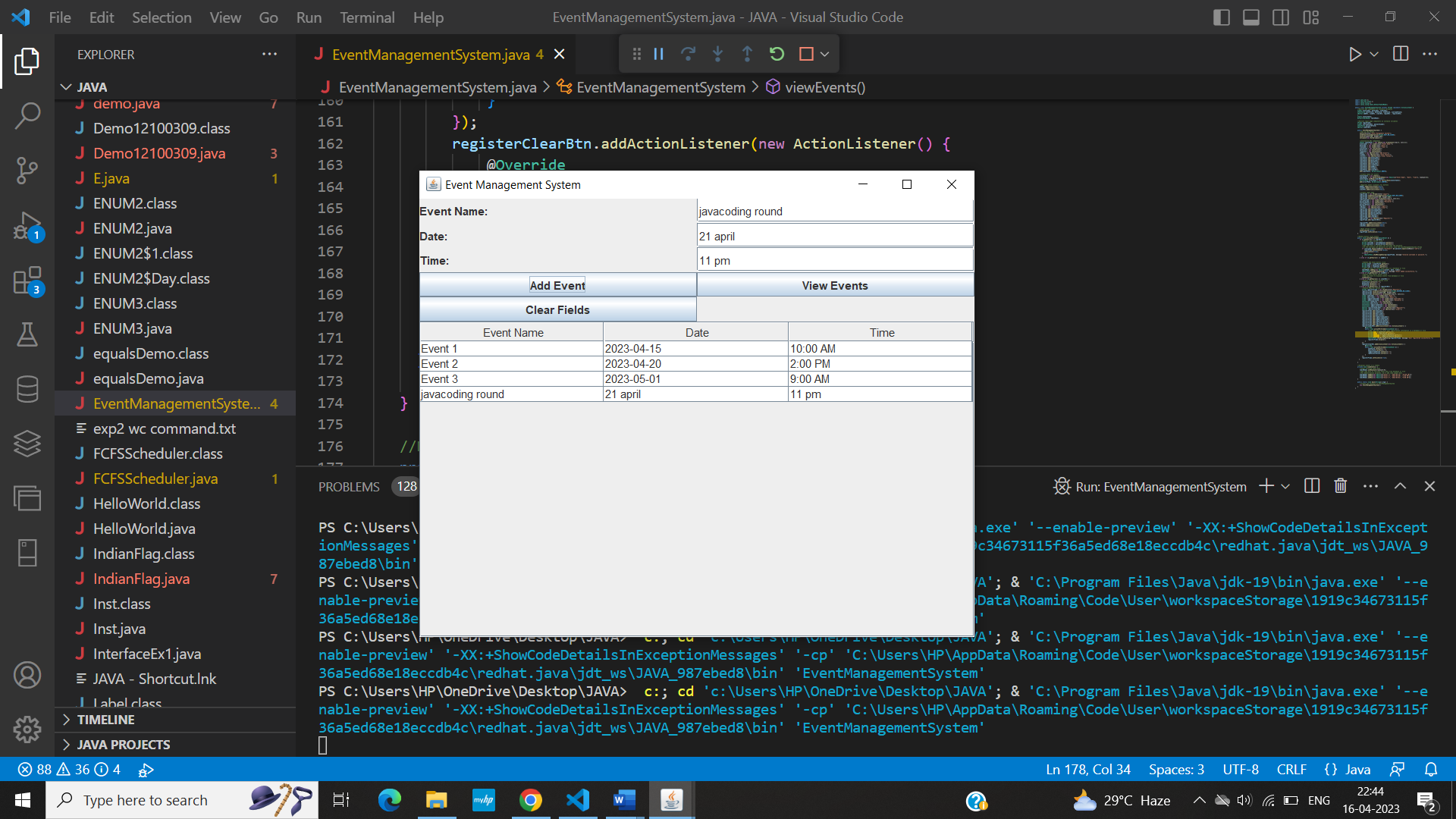
**Login page:-**

****

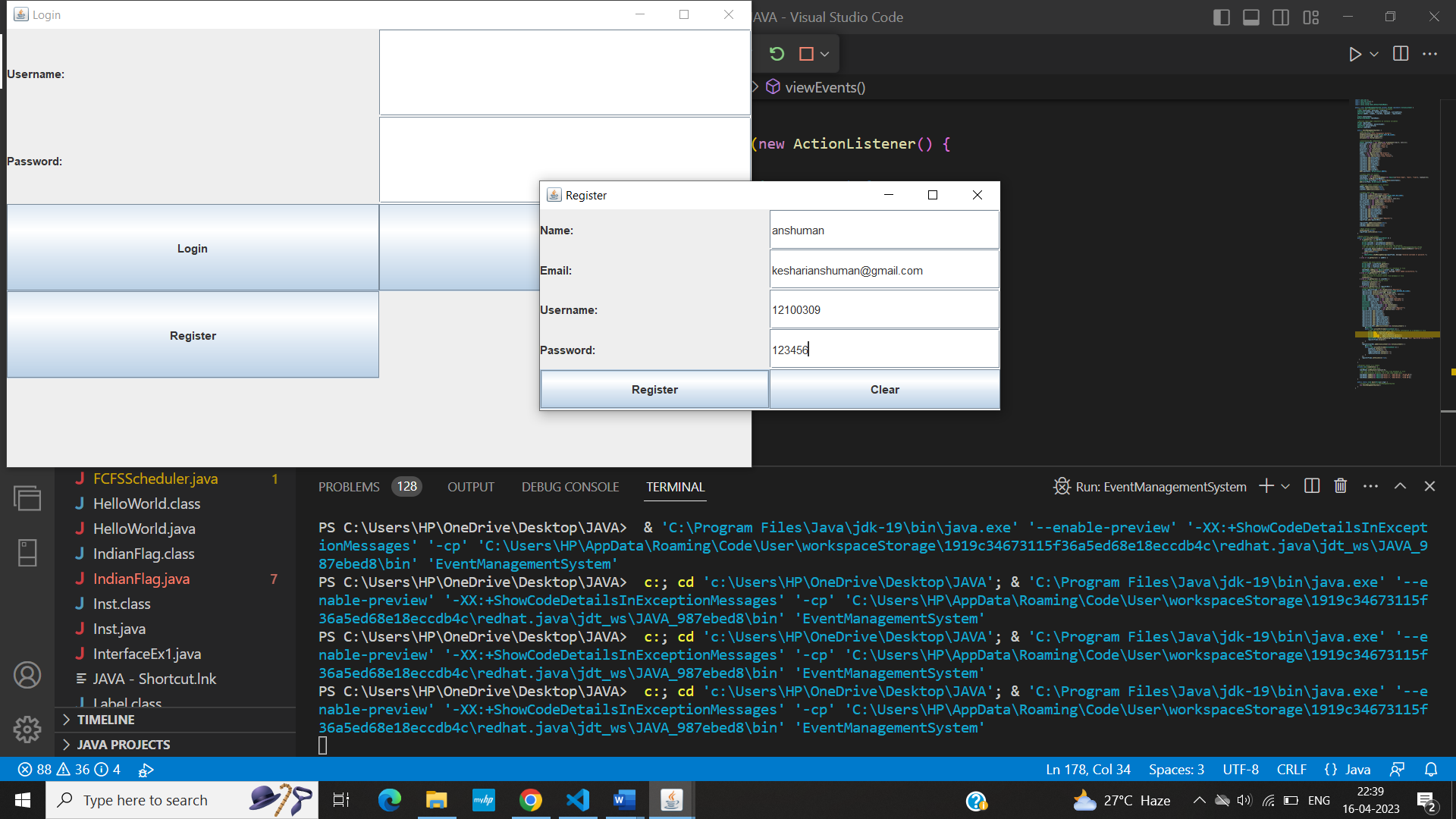
**Add Event page:-**

****

**View Event page:-**

****

**Register page:-**

****

1. **Sample Code**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

public class EventManagementSystem extends JFrame implements ActionListener {

   //Declare GUI components as instance variables

   JLabel eventLabel, dateLabel, timeLabel;

   JTextField eventField, dateField, timeField, usernameField;

   JButton addBtn, viewBtn, clearBtn, loginBtn , registerBtn;

   ;

   JTable eventsTable;

   DefaultTableModel tableModel;

   //Declare login page components as instance variables

   JFrame loginFrame;

   JLabel usernameLabel, passwordLabel;

   JTextField passwordField;

   JButton submitBtn;

   public EventManagementSystem() {

      //Set up the JFrame

      setTitle("Event Management System");

      setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

      setSize(600, 500);

      setLayout(new BorderLayout());

      //Initialize GUI components

      JPanel inputPanel = new JPanel(new GridLayout(5, 2));

      eventLabel = new JLabel("Event Name:");

      dateLabel = new JLabel("Date:");

      timeLabel = new JLabel("Time:");

      eventField = new JTextField();

      dateField = new JTextField();

      timeField = new JTextField();

      addBtn = new JButton("Add Event");

      viewBtn = new JButton("View Events");

      clearBtn = new JButton("Clear Fields");

      inputPanel.add(eventLabel);

      inputPanel.add(eventField);

      inputPanel.add(dateLabel);

      inputPanel.add(dateField);

      inputPanel.add(timeLabel);

      inputPanel.add(timeField);

      inputPanel.add(addBtn);

      inputPanel.add(viewBtn);

      inputPanel.add(clearBtn);

      add(inputPanel, BorderLayout.NORTH);

      //Create table for events

      eventsTable = new JTable();

      tableModel = new DefaultTableModel(new Object[]{"Event Name", "Date", "Time"}, 0);

      eventsTable.setModel(tableModel);

      JScrollPane scrollPane = new JScrollPane(eventsTable);

      add(scrollPane, BorderLayout.CENTER);

      //Register event listeners for buttons

      addBtn.addActionListener(this);

      viewBtn.addActionListener(this);

      clearBtn.addActionListener(this);

      //Create login page

      loginFrame = new JFrame("Login");

      loginFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

      loginFrame.setSize(800, 500);

      loginFrame.setLayout(new GridLayout(5, 2));

      usernameLabel = new JLabel("Username:");

      passwordLabel = new JLabel("Password:");

      usernameField = new JTextField();

      passwordField = new JTextField();

      loginBtn = new JButton("Login");

      clearBtn = new JButton("clear");

      loginFrame.add(usernameLabel);

      loginFrame.add(usernameField);

      loginFrame.add(passwordLabel);

      loginFrame.add(passwordField);

      loginFrame.add(loginBtn);

      loginFrame.add(clearBtn);

      registerBtn = new JButton("Register");

      loginFrame.add(registerBtn);

      registerBtn.addActionListener(this);

      loginBtn.addActionListener(this);

      clearBtn.addActionListener(this);

      //Make JFrame visible

      setVisible(false);

      loginFrame.setVisible(true);

   }

   //Handle button click events

   public void actionPerformed(ActionEvent e) {

      if (e.getSource() == loginBtn) {

         //Handle Login button click

         String username = usernameField.getText();

         String password = passwordField.getText();

         //Add code here to validate the username and password

         //If the username and password are valid, show the EventManagementSystem JFrame

         if (username.equals("Anshuman") && password.equals("123")) {

            loginFrame.setVisible(false);

            setVisible(true);

         } else {

            JOptionPane.showMessageDialog(loginFrame, "Invalid username or password.");

         }

      } else if (e.getSource() == addBtn) {

         //Handle Add Event button click

         String event = eventField.getText();

         String date = dateField.getText();

         String time = timeField.getText();

         //Add code here to store event data in a database or file

         tableModel.addRow(new Object[]{event, date, time});

         JOptionPane.showMessageDialog(this, "Event added successfully.");

      } else if (e.getSource() == viewBtn) {

         //Handle View Events button click

         //Add code here to display events from database or file

         viewEvents();

      } else if (e.getSource() == clearBtn) {

         //Handle Clear Fields button click

         eventField.setText("");

         dateField.setText("");

         timeField.setText("");

      } else if (e.getSource() == registerBtn) {

         //Handle Register button click

         JFrame registerFrame = new JFrame("Register");

         registerFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

         registerFrame.setSize(500, 250);

         registerFrame.setLayout(new GridLayout(5, 2));

         JLabel nameLabel = new JLabel("Name:");

         JLabel emailLabel = new JLabel("Email:");

         JLabel regUsernameLabel = new JLabel("Username:");

         JLabel regPasswordLabel = new JLabel("Password:");

         JTextField nameField = new JTextField();

         JTextField emailField = new JTextField();

         JTextField regUsernameField = new JTextField();

         JTextField regPasswordField = new JTextField();

         JButton registerSubmitBtn = new JButton("Register");

         JButton registerClearBtn = new JButton("Clear");

         registerFrame.add(nameLabel);

         registerFrame.add(nameField);

         registerFrame.add(emailLabel);

         registerFrame.add(emailField);

         registerFrame.add(regUsernameLabel);

         registerFrame.add(regUsernameField);

         registerFrame.add(regPasswordLabel);

         registerFrame.add(regPasswordField);

         registerFrame.add(registerSubmitBtn);

         registerFrame.add(registerClearBtn);

         registerSubmitBtn.addActionListener(new ActionListener() {

             @Override

             public void actionPerformed(ActionEvent e) {

                 //Add code here to store user registration information in a database or file

                 String name = nameField.getText();

                 String email = emailField.getText();

                 String username = regUsernameField.getText();

                 String password = regPasswordField.getText();

                 JOptionPane.showMessageDialog(registerFrame, "User registered successfully.");

                 registerFrame.dispose();

             }

         });

         registerClearBtn.addActionListener(new ActionListener() {

             @Override

             public void actionPerformed(ActionEvent e) {

                 nameField.setText("");

                 emailField.setText("");

                 regUsernameField.setText("");

                 regPasswordField.setText("");

             }

         });

         registerFrame.setVisible(true);

     }

   }

   //Displays events in a JTable

   private void viewEvents() {

      //Clear existing table data

      tableModel.setRowCount(0);

      //Add code here to retrieve event data from database or file

      //For demonstration purposes, we'll add some sample data

      tableModel.addRow(new Object[]{"Event 1", "2023-04-15", "10:00 AM"});

      tableModel.addRow(new Object[]{"Event 2", "2023-04-20", "2:00 PM"});

      tableModel.addRow(new Object[]{"Event 3", "2023-05-01", "9:00 AM"});

   }

   public static void main(String[] args) {

      //Create new instance of EventManagementSystem

      new EventManagementSystem();

   }

}

**Programming Language Used**:-

This code is an implementation of an event management system with a graphical user interface (GUI) in Java. The program uses Swing, which is a set of GUI components for Java, to create the GUI. The JFrame class is used to create the main window, and various Swing components such as JLabel, JTextField, JButton, JTable, and JScrollPane are used to create the input fields, buttons, and table to display events. The program also includes a login page to allow access to the event management system. Once the user logs in, the main window of the event management system is displayed. The program includes event listeners for the buttons, which are implemented using the ActionListener interface. When the user clicks on a button, the actionPerformed() method is called, and the appropriate action is taken depending on which button was clicked. The "Add Event" button adds the event information entered by the user to a JTable. The "View Events" button displays the events stored in the JTable. The "Clear Fields" button clears the input fields. The "Login" button logs the user into the system, while the "Register" button opens a new JFrame for user registration. The program also includes code to store the event data and user registration information in a database or file, but the implementation of this code is not shown in the provided code snippet.

**Outcomes Code:**

An event management system implemented in Java using JFrame can provide a graphical user

interface for managing events. The system can allow the user to add, view, update, and delete

events. Here are some possible features of an event management system implemented in

JFrame:

Adding Events: The system can have a form to add new events, where the user can input event

details such as name, date, time, location, and description. The system can then save the event

details to a database or file.

Viewing Events: The system can have a view that displays a list of all events. The user can select

an event from the list to view its details. The system can also provide a search feature to find

events based on a keyword or date range.

Deleting Events: The system can allow the user to delete events from the system. The system

can prompt the user for confirmation before deleting an event.

User Management: The system can have a feature for managing users such as creating new users, updating user details, and deleting users. The system can also provide role-based access control

to limit user access to certain features of the system.

Overall, an event management system implemented in JFrame can provide a user-friendly

interface for managing events, which can help to improve productivity and efficiency for event organizers.

**Appendices**

**NA**

**REFERENCES**

[1] RoozbehDerakhshan, Maria E. Orlowska and Xue Li, “RFID Data Management: Challenges and Opportunities”, IEEE International Conference on RFID, 2007 [2] KullaprapaNavanugraha, PornanongPongpaibool, ChaleeVorakulpipat, NuttapongSanglerdsinlapachai, NutvadeeWongtosrad,SiwarukSiwamogsatham, “The Deployment of the AutoID System in a Conference”, PICMET, IEEE, pp.1-7, 2010. [3][https://www.ijirset.com/upload/2016/march/101 Event.pdf](https://www.ijirset.com/upload/2016/march/101%20Event.pdf).

[4] L. McCarthy and K. Michael, “Is it the End of Barcodes in Supply Chain Management?”, Proceedings of the Collaborative Electronic Commerce Technology and Research Conference LatAm, 2005. [5] Lung-Chuang Wang, “Enhancing construction quality inspection and management using FID technology”, Journal Automation in Construction, Elsevier, pp. 468- 469, 2008. [6] Paul M. Swamidass, “Bar Code Users and Their Performance”, White Paper, UNOVA Inc., 1998.