# International Credit Hours Programs- Faculty of Engineering- Ain Shams ...

**CSE 241  
Object Oriented Programming**

Project: Event Management System

Team:

Ahmed Tamer Mohamed Bahgat Abdelmoniem Mohamed 24P0179

SAIFELDIN AHMED MOHAMMED MOHAMMED SALEM 24P0153

Eyad Wael Abdel Moneim Ahmed Swelam 24P0137

Amr Elsayed Ahmed Ibrahim Mohamed Semary 24P0066

Abderrhman Khaled Ahmed Sobhy 24P0475

Mohamad wael mohamad abdelkader hanafy 24P0033

# TABLE OF CONTENTS

[1](#_Toc197719361)

[TABLE OF CONTENTS 2](#_Toc197719362)

[1-Program Insight 3](#_Toc197719363)

[2-Program Design 3](#_Toc197719364)

[3-Program Samples 7](#_Toc197719365)

[4-UML Diagram 10](#_Toc197719366)

[5-GitHub Link 11](#_Toc197719367)

[6-References 11](#_Toc197719368)

# 1-Program Insight

This program is developed in Java using Object-Oriented Programming (OOP) principles. It is designed to manage and organize events while providing attendees with information about the designated rooms where **the** event**s** will take place.

# 2-Program Design

In this section, we will talk about the design of this program, the classes implemented, some of the Challenges encountered and how we solved them, and the techniques used to make this program.

2.1-Classes

We started by setting up the classes first. we made:  
  
**-User class:** a public abstract class   
Purpose: it's the Base class for all user types in the system, handling the passwords, usernames, and wallets of all users  
  
**-Wallet class:** a public class   
Purpose: class designed to represent the balance and ownership of the wallet of the attendee  
  
**-Organizer class:** Public class that extends (Inherits) from the Person class  
Purpose: class made to represent the event organizer profile where he can add events, remove them, or show them  
**-Category class:** Public class   
Purpose: Class designed to organize events based on categories

**-Admin class:** Public class that extends (Inherits) from the Person class

Purpose: This is the admin's class that shows information about their roles, working hours, etc  
  
**-Gender Enum:** public Enum that has 2 variables, which are male and female.

**-Attendee class:** public class that extends from person class   
Purpose: Represents the events attendees with interest tracking and event selection **-Database Class:** public class Purpose: Central data storage for the system using static ArrayLists

**-Room Class:** public class  
Purpose: holds all rooms and their events

**-Event Class:** public class

Purpose: holds all events and their times, attendees, organizer, and price

**UserType Enum:** public Enum that has 3 variables, which are Attendee, Organizer, Admin.

**RegisterType Enum:** public Enum that has 3 variables, which are none, attendee, organizer

**Hash Helper Class:** public class  
Purpose: creates the hash of passwords

**ScreenHelper Class:** public class  
Purpose: has the function that centers a node and a function that shows an alert with the given string

**LoginRegister Screen Class:** public class

Purpose: shows the login and register page of the program

**Attendee Screen Class:** public class  
Purpose: Shows the Attendee’s page in the program

**Admin Screen Class:** public class  
Purpose: shows the admin’s page in the program

**Organizer Screen Class:** public class

Purpose: shows the Organizer’s page in the program

2.2-Challenges encountered

The most challenging issue we encountered was getting JavaFX to build successfully, as it took us the longest time on this project to solve it

2.3-Detailed Solution

After a deep search, we determined that the issue was a misconfiguration within the IDE. Therefore, we went through the configuration and ensured that the JavaFX SDK was correctly added, then we restructured our build configuration, rebuilt the project, and successfully implemented JavaFX into the Program.

2.4-Techniques used:

* Inheritance, Encapsulation, Polymorphism, Composition
* Collections Framework
* ArrayList for data storage
* Enum for gender representation
* Date/Time Handling
* LocalDate and LocalTime for scheduling
* SimpleDateFormat for display
* Exception Handling
* Static members
* Static database class for shared data

# 3-Program Samples

3.1-Register page

This is the register page where you register as an Attendee or an Organizer, You have to input your Username, Password, Birthday, Address, Gender, and interests.

A screenshot of a computer

AI-generated content may be incorrect.

3.2-Admin interface

A screenshot of a computer

AI-generated content may be incorrect.Admins of the program can manage the Attendees, Rooms, Categories, and Events

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

3.3-Organizer Interface

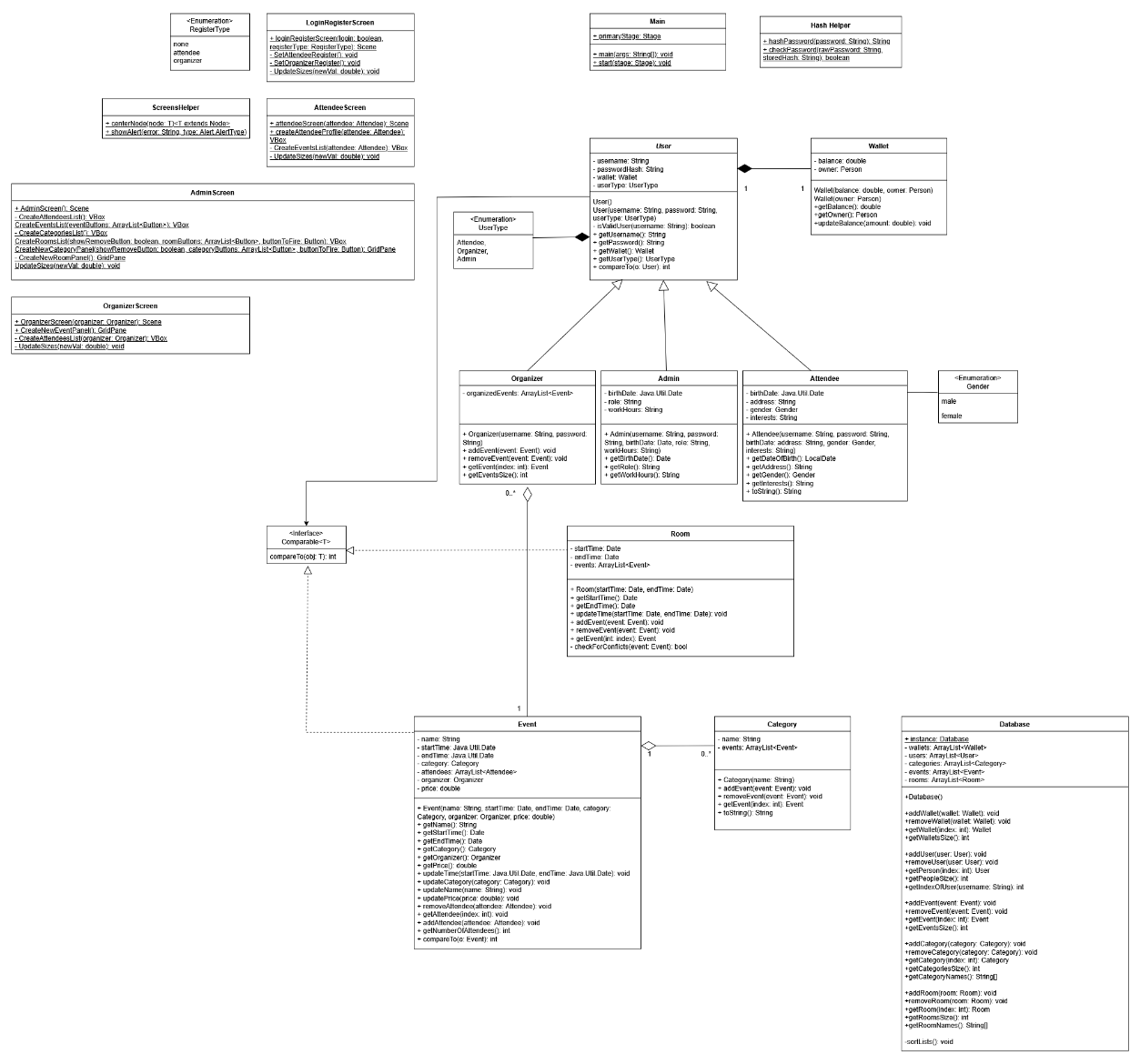
Event Organizers can manage events through this page

A screenshot of a computer

AI-generated content may be incorrect.

# 4-UML Diagram

This is the whole UML diagram for the program.



# 5-GitHub Link

<https://github.com/EyadDev/EventManagementSystem>

# 6-References

-OOP textbook <https://lms.eng.asu.edu.eg/mod/resource/view.php?id=169327>  
  
-OOP project requirements PDF

<https://lms.eng.asu.edu.eg/mod/resource/view.php?id=171453>

-Oracle on JavaFX

<https://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm>