**Group Members:**

Abdulmanan (22i-2391)

Maryam Sajid (22i-2383)

Ahmad Talal (21i-0734)

**Project Title:** SwiftEvent: Event Scheduling and Management System

**Project Overview**

This project is a Java-based application designed to manage and schedule events. The application uses JavaFX for the graphical user interface and works with an MYSQL database to manage data persistence. The features of the application include creating and canceling events, registering for events, submitting feedback, and generating reports etc.

**File Structure Overview**

The project consists of three main packages:

1. **application**: Contains the main entry point and UI files. (fxml files)
2. **application.controller**: Manages the logic for different screens and user interactions.
3. **application.model**: Contains the classes representing the data models.

**Detailed Description of Files**

**1. application Package**

This package contains the main class and the user interface (UI) files for different screens.

* **Main.java**: This is the entry point of the application. It initializes the JavaFX application and loads the main UI screen.
* **UI Files**:
  + **AttendeeScreen.fxml**: UI layout for the attendee's screen.
  + **CancelEventScreen.fxml**: UI layout to cancel events.
  + **CreateEventScreen.fxml**: UI for creating new events.
  + **FeedbackScreen.fxml**: UI for submitting event feedback.
  + **MainScreen.fxml**: The main screen of the application.
  + **OrganizerScreen.fxml**: UI for organizing events.
  + **RegisteredEventsScreen.fxml**: Displays registered events for a user.
  + **RegisterScreen.fxml**: UI for registering for events.
  + **ReportScreen.fxml**: UI to view and generate event reports.
  + **ScheduledEventScreen.fxml**: UI to display scheduled events.
  + **VenueBookScreen.fxml**: UI for booking event venues.

To create the documentation based on the structure shown in the provided screenshot, here's a sample outline and brief documentation that can be included in an MS Word file. I'll break it down by each part of the project and the associated files.

**Project Documentation**

**Project Overview**

This project is a Java-based application designed to handle events, feedback, and registrations. The application uses JavaFX for the graphical user interface and works with an SQL database to manage data persistence. The features of the application include creating and canceling events, registering for events, submitting feedback, and generating reports.

**File Structure Overview**

The project consists of three main packages:

1. **application**: Contains the main entry point and UI files.
2. **application.controller**: Manages the logic for different screens and user interactions.
3. **application.model**: Contains the classes representing the data models (e.g., Event, EventDetails).

**Detailed Description of Files**

**1. application Package**

This package contains the main class and the user interface (UI) files for different screens.

* **Main.java**: This is the entry point of the application. It initializes the JavaFX application and loads the main UI screen.
* **UI Files**:
  + **AttendeeScreen.fxml**: UI layout for the attendee's screen.
  + **CancelEventScreen.fxml**: UI layout to cancel events.
  + **CreateEventScreen.fxml**: UI for creating new events.
  + **FeedbackScreen.fxml**: UI for submitting event feedback.
  + **MainScreen.fxml**: The main screen of the application.
  + **OrganizerScreen.fxml**: UI for organizing events.
  + **RegisteredEventsScreen.fxml**: Displays registered events for a user.
  + **RegisterScreen.fxml**: UI for registering for events.
  + **ReportScreen.fxml**: UI to view and generate event reports.
  + **ScheduledEventScreen.fxml**: UI to display scheduled events.
  + **VenueBookScreen.fxml**: UI for booking event venues.

**2. application.controller Package**

This package contains the controller classes that handle user interactions and business logic for each of the UI screens.

* **AttendeeController.java**: Manages the logic for the attendee's screen, such as viewing and registering for events.
* **CancelEventController.java**: Handles the cancellation of events.
* **CreateEventController.java**: Manages the creation of events, including event details and feedback handling.
* **FeedbackController.java**: Manages user-submitted feedback, including saving feedback to the database.
* **MainController.java**: The controller for the main screen that orchestrates the navigation between other screens.
* **OrganizerController.java**: Manages the organizer's dashboard for creating, managing, and canceling events.
* **RegisterController.java**: Handles the registration process for attendees to events.
* **RegisteredEventController.java**: Manages the list of registered events and their details.
* **RegisterEventController.java**: Handles the logic for registering for a specific event.
* **ReportController.java**: Handles the logic to display reports for events, including their average ratings and attendee lists.
* **VenueBookController.java**: Manages venue booking logic for events.
* **ViewEventController.java**: Handles viewing details of a specific event.

**Key Features:**

1. **Event Management**: Users can create and cancel events via the organizer's interface.
2. **Registration**: Attendees can register for events and view a list of registered events.
3. **Feedback**: Attendees can submit feedback, which includes a rating and comments on events.
4. **Reports**: The report screen generates various reports about events, such as average ratings.
5. **Venue Booking**: Organizers can book venues for the events.

**Design Considerations**

1. **Modular Design**: The application is divided into modular components such as controllers, models, and views to ensure scalability and ease of maintenance.
2. **Separation of Concerns**: The model, view, and controller are clearly separated, adhering to the MVC (Model-View-Controller) design pattern.
3. **Error Handling**: Appropriate error handling is implemented to ensure smooth user interactions and avoid application crashes.

**Conclusion**

This application serves as a platform for managing events, feedback, and registrations, with separate modules for different functionalities, including event creation, attendee registration, and feedback submission. The database integration ensures data persistence, while the JavaFX interface provides a user-friendly experience.