AT2-Milestone 1

Further Programming (2407)  
**COSC2288 / COSC2648**

RMIT Online

Due Date: Friday, 19th April 2024

Student : Amit Munjal

Id# s3925455

Email:s3925455@student.rmit.edu.au

Table of Contents

[Classes 3](#_Toc164191213)

[Class Description: 3](#_Toc164191214)

[Relationship between classes 5](#_Toc164191215)

[Class Diagram 7](#_Toc164191216)

[Entity Relationship Diagram (ERD) 8](#_Toc164191217)

[ER Diagram 9](#_Toc164191218)

[User Interface 10](#_Toc164191219)

[Login screen design 10](#_Toc164191220)

[Login success/ unsuccess screen design 10](#_Toc164191221)

[Main Menu design 11](#_Toc164191222)

[Display existing jobs screen design 12](#_Toc164191223)

[Select by Category screen design 12](#_Toc164191224)

[References 14](#_Toc164191225)

## Classes

### Class Description:

**1. FileHandler**

• **Attributes**:

• fileToRead: File to be read.

• fileToWrite: File to be written.

**• Methods:**

• readJobRequestsFromFile(): Reads job requests from a file.

• writeJobRequestsToFile(): Writes job requests to a file.

**2. LoginController**

**• Attributes:**

• userName: Username for login.

• password: Password for login.

**3. ShowHiringMenu**

**• Attributes:**

• client: Client's name.

• title: Title of the hiring.

• artist: Artist for the hiring.

• date: Date of the hiring.

• time: Time of the hiring.

• target: Target audience.

• type: Type of the hiring.

• category: Category of the hiring.

• **Methods:**

• showHiringMenu(): Displays the hiring menu.

**4. UnitTest**

**• Methods:**

• testGetName(): Unit test for getName() method.

• testGetVenues(): Unit test for getVenues() method.

• testGetOldJobRequests(): Unit test for getOldJobRequests() method.

• testDisplayOldJobRequests(): Unit test for displayOldJobRequests() method.

• testSelectByCategory(): Unit test for selectByCategory() method.

• testSelectFromVenueList(): Unit test for selectFromVenueList() method.

• testSelectFromMatchingList(): Unit test for selectFromMatchingList() method.

5. **CustomException**

**• Attributes:**

• message: Message describing the exception.

• **Methods:**

• customException(): Constructs a custom exception with a message.

**6. VenueMatcher**

**• Attributes:**

• listJobRequests: List of old job requests.

• browseByCategory: Browse venue by category.

• searchByName: Search venue by name.

• autoMatchEvents: Auto-match events with suitable venues.

• showOrderSummary: Show order summary.

**• Methods:**

• main(): Main method of the application.

• start(): Initializes the JavaFX application.

• handleUserInput(): Handles user input.

• showAlert(): Displays an alert message.

• launchDisplayOldJobRequest(): Launches the display of old job requests.

• launchSelectByCategory(): Launches the selection by category.

• launchSelectFromVenueList(): Launches the selection from venue list.

• launchAutoMatch(): Launches the auto-matching process.

• launchShowOrderSummary(): Launches the display of order summary.

**7. SaveHiringDetails**

**• Attributes:**

• client: Client's name.

• title: Title of the hiring.

• artist: Artist for the hiring.

• date: Date of the hiring.

• time: Time of the hiring.

• target: Target audience.

• type: Type of the hiring.

• category: Category of the hiring.

**• Methods:**

• saveHiringDetails(): Saves hiring details.

**8. DateValidator**

**• Attributes**:

• date: Date to be validated.

• month: Month to be validated.

• year: Year to be validated.

**• Methods:**

• DateValidator(): Constructor for date validation.

**9. TimeValidator**

**• Attributes:**

• time: Time to be validated.

• numberOfHours: Number of hours to be validated.

**• Methods:**

• TimeValidator(): Constructor for time validation.

**10. Venue**

**• Attributes:**

• name: Name of the venue.

• capacity: Capacity of the venue.

• suitableFor: Type of events suitable for the venue.

• category: Category of the venue.

**• Methods:**

• toString(): Returns string representation of the venue.

• getName(): Gets the name of the venue.

• getCapacity(): Gets the capacity of the venue.

• getSuitableFor(): Gets the type of events suitable for the venue.

• getCategory(): Gets the category of the venue.

**11. Order**

**• Attributes:**

• quantity: Quantity of the order.

• subTotal: Subtotal of the order.

**• Methods:**

• calculateDiscount(): Calculates discount for the order.

• display(): Displays the order details.

**12. CheckBookingClash**

**• Attributes:**

• date: Date to check for booking clash.

• time: Time to check for booking clash.

• numberOfHours: Number of hours for booking.

• **Methods:**

• checkBookingClash(): Checks for booking clash.

### Relationship between classes

Here are the relationships between the provided classes based on their attributes and methods:

**FileHandler and LoginController:**

The LoginController class use the FileHandler class to read or write login-related information, such as usernames and passwords, from or to a file. For example, the LoginController may call the readJobRequestsFromFile() method of FileHandler to read login credentials from a file.

**ShowHiringMenu and Venue:**

The ShowHiringMenu class includes attributes like client, title, artist, date, time, target, type, and category, which relate to information about a hiring event. The Venue class also includes attributes like name, capacity, suitableFor, and category, which are relevant to a venue. Therefore, there is a relationship where a ShowHiringMenu object references a Venue object to associate a hiring event with a specific venue.

**VenueMatcher and ShowHiringMenu:**

The VenueMatcher class is to control menu options related to venues, such as listing old job requests, browsing by category, searching by name, auto-matching events, and showing order summaries. These functionalities are closely related to the attributes and methods of the ShowHiringMenu class, indicating that VenueMatcher may interact with ShowHiringMenu to manage venue-related tasks.

**UnitTest:**

The UnitTest class includes methods for testing various functionalities of other classes. It doesn't seem to have direct relationships with other classes for data management but rather serves as a testing utility.

**CustomException:**

The CustomException class is used to create custom exceptions with a specific message. It may be thrown or handled by other classes, especially those dealing with file handling or data validation.

**SaveHiringDetails and ShowHiringMenu:**

Both classes handle hiring-related information, with SaveHiringDetails specifically focused on saving details. There might be a relationship where SaveHiringDetails interacts with ShowHiringMenu to save hiring details entered through the menu.

**DateValidator and TimeValidator:**

Both classes are responsible for validating dates and times, respectively. They may be used by other classes, such as ShowHiringMenu or SaveHiringDetails, to ensure that the entered date and time values are valid.

**CheckBookingClash:**

The CheckBookingClash class checks for booking clashes based on date, time, and number of hours. It may be used by classes involved in booking management, such as ShowHiringMenu or SaveHiringDetails, to prevent conflicts in bookings.

These relationships outline how the classes may interact or depend on each other to perform various functionalities within the system.

### Class Diagram

## Entity Relationship Diagram (ERD)

An Entity Relationship Diagram (ERD) visually represents the relationships between entities in a database. Below is the plan for the database tables based on the provided entities:

**1. User Table:**

**• Attributes:**

• UserID (Primary Key)

• Login

• Password

• Role

**2. Venue Table:**

**• Attributes:**

• VenueID (Primary Key)

• Name

• Capacity

• SuitableFor

• Category

• PricePerHour

**3. Job\_Request Table:**

**• Attributes:**

• RequestNumber (Primary Key)

• Venue (Foreign Key referencing VenueID from Venue Table)

• NumberOfHours

• Date

• Time

• EventName

• ArtistName

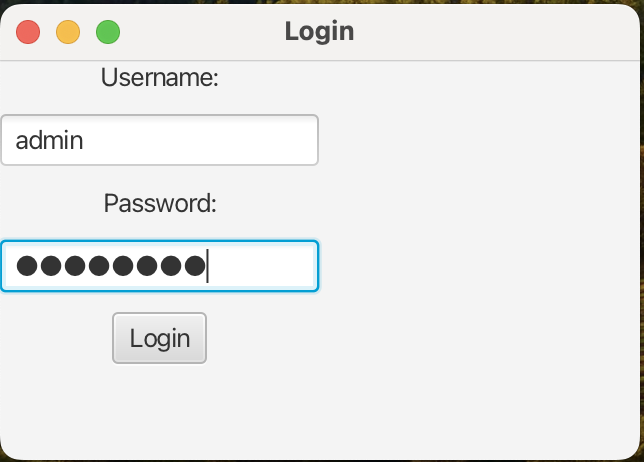
• RequesterName

• EventType

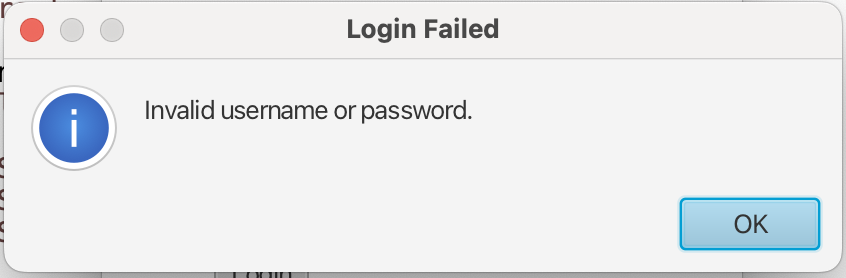
### ER Diagram

## User Interface

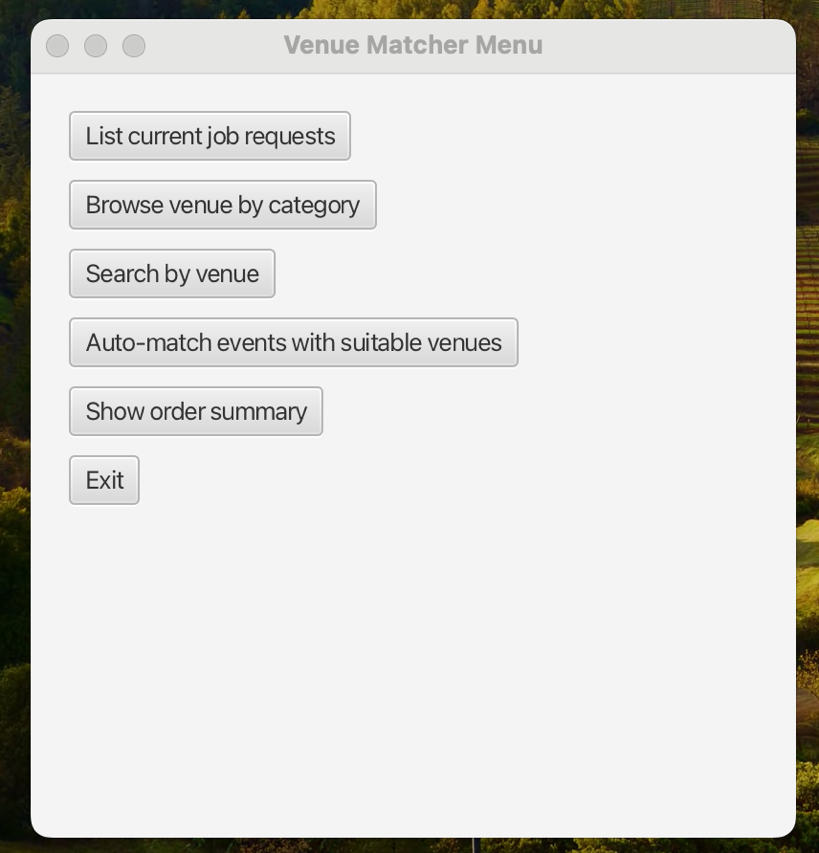
### Login screen design



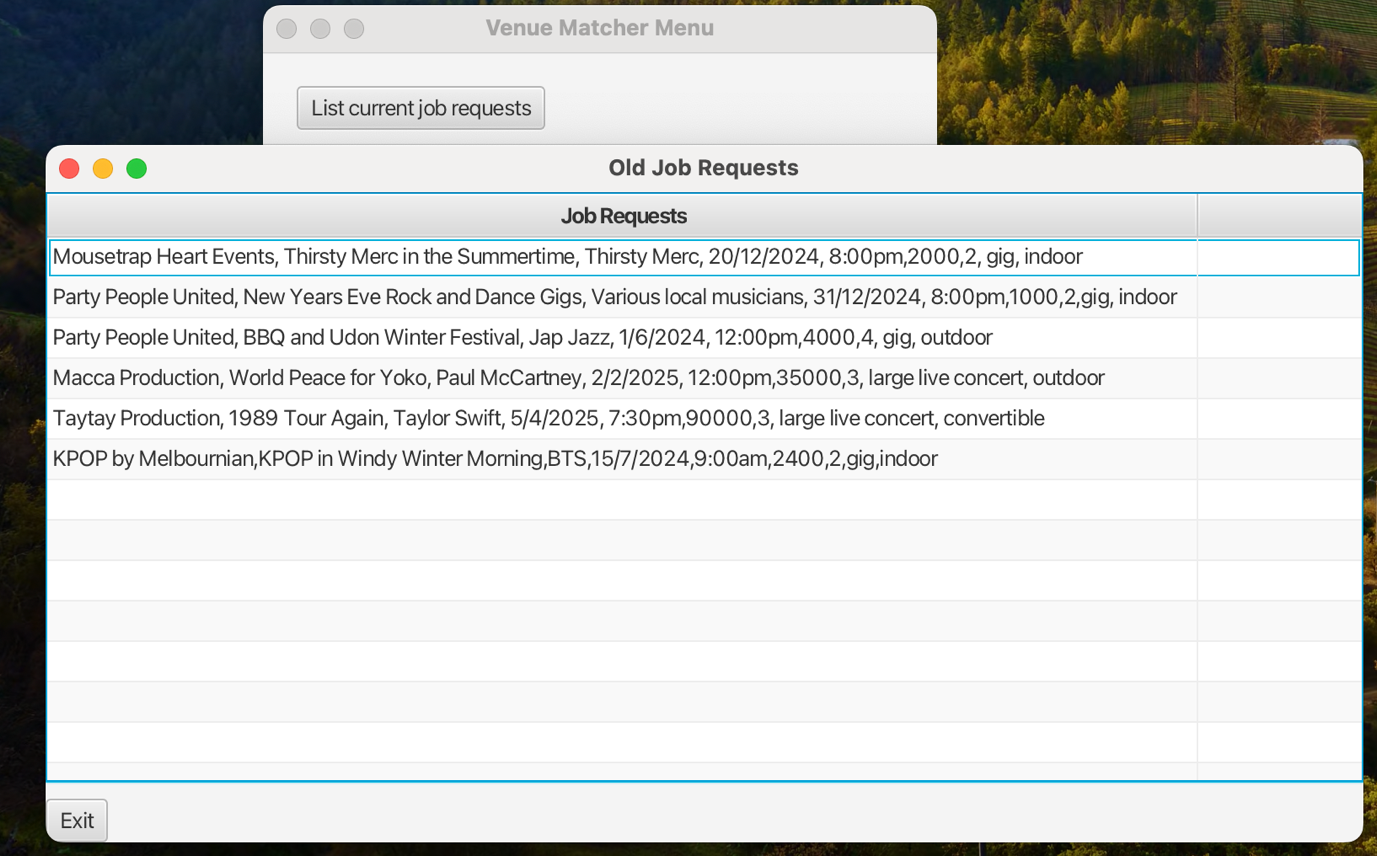
### Login success/ unsuccess screen design



### Main Menu design



### Display existing jobs screen design



### Select by Category screen design

## References