Object-Oriented Programming Project Reservation System

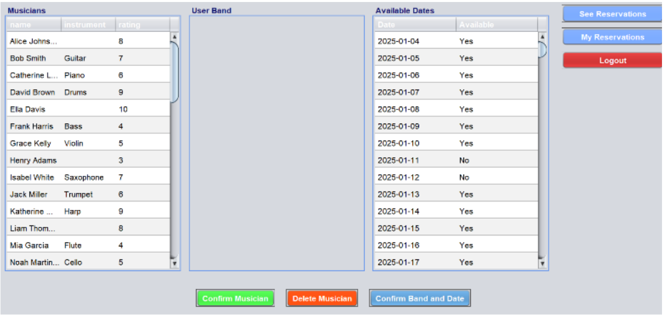
Lab Assistant: Andrei Pelle
Student: Joshua Pop
GitHub Repository: Reservation System Final

Proposed Features

Core System Features

- **Login System:** Automatically distinguishes between Member and Admin user roles upon login.
- **Registration Page:** Allows new users to register, storing their details securely in a database.
- **User Authentication:** Passwords are hashed using a one-way encryption mechanism during registration for secure storage.

Member Features



name	instrument	rating
Alice Johns...		8
Bob Smith	Guitar	7
Catherine L...	Piano	6
David Brown	Drums	9
Ella Davis		10
Frank Harris	Bass	4
Grace Kelly	Violin	5
Henry Adams		3
Isabel White	Saxophone	7
Jack Miller	Trumpet	6
Katherine ...	Harp	9
Liam Thom...		8
Mia Garcia	Flute	4
Noah Martin...	Cello	5

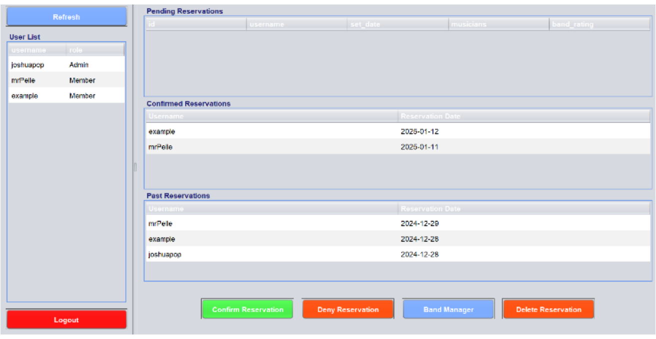
Date	Available
2025-01-04	Yes
2025-01-05	Yes
2025-01-06	Yes
2025-01-07	Yes
2025-01-08	Yes
2025-01-09	Yes
2025-01-10	Yes
2025-01-11	No
2025-01-12	No
2025-01-13	Yes
2025-01-14	Yes
2025-01-15	Yes
2025-01-16	Yes
2025-01-17	Yes

- **Band Manager Tab:**
    - Displays three sections:
        - **Musicians:** A list of pre-approved performers, added manually by Admin. Each musician includes a rating.
        - **User Band:** A custom band created by the user, with size and score restrictions (band size ≤ 5, cumulative score > 20).

        - **Available Dates:** Tracks unavailable dates based on admin-approved reservations, showing free dates for selection.

- **Musician Management:**
    - **Confirm Musician:** Adds a selected musician from the Musicians table to the User Band.
    - **Delete Musician:** Removes a musician from the User Band.
- **Band and Date Management:**
    - **Confirm Band and Date:**
        - Validates the User Band for score and size requirements.
        - Checks date availability.
        - Submits a reservation request to the Admin for approval.
- **Reservation Views:**
    - **See Reservations:** Opens a pop-up displaying all future concert dates.
    - **My Reservations:** Displays all reservation requests made by the user, including their status and details.
- **Logout:** Logs the user out of the system.

Admin Features



User List

username	role
joshuapop	Admin
mrPelle	Member
example	Member

Pending Reservations

id	username	set_date	musicians	band_rating

Confirmed Reservations

Username	Reservation Date
example	2025-01-12
mrPelle	2025-01-11

Past Reservations

Username	Reservation Date
mrPelle	2024-12-29
example	2024-12-28
joshuapop	2024-12-28

- **Dashboard Overview:**
    - Displays three tables:
        - **Pending Requests:** Shows all reservation requests submitted by Members awaiting approval.
        - **Confirmed Reservations:** Displays all approved reservations.
        - **Past Reservations:** Shows a history of completed reservations.

- **Reservation Management:**
    - **Confirm Reservation:** Approves a selected pending reservation and moves it to Confirmed Reservations.
    - **Deny Reservation:** Rejects a selected pending reservation.
    - **Delete Reservation:** Deletes a selected confirmed reservation.
- **Band Management:**
    - Admin has access to the Member Band Manager features for review or testing purposes.
- **User Management:**
    - Displays a list of all registered users and their roles (Member or Admin).

Instructions and Requirements

Environment Setup

1. Install **IntelliJ IDEA** (version supporting Java 21 or higher).
2. Install **PostgreSQL** and set up a database for the project.
3. Configure the **JDBC Driver** in IntelliJ IDEA to connect the application to the database.

Running the Application

- Clone the GitHub repository and import it into IntelliJ IDEA.
- Configure the database connection string and credentials in the application’s settings.
- Run the project to access the login and registration features.

Additional Notes

- **Security:**
    - Passwords are securely hashed and stored.
    - Ensure input validation to prevent SQL injection.
- **Concurrency:**
    - Implement mechanisms to handle simultaneous reservation requests to avoid conflicts.
- **Scalability:**

  - Use database indexing and optimized queries to support a growing number of users and reservations.