

VISUAL PROGRAMMING

CS342

CAR RENTAL SOFTWARE

GROUP:611

PREPARED BY:

Hanan Alrashidi
Lamis Alharbi
Layan Alswilem
Shahad Alharbi

SUPERVISOR:

Dr.fatimah Alkhudayr

TABLE OF CONTENTS

1.Application Overview.....	3
1.1 Main Screen Page	3
1.2 Log in Page.....	3
1.3 Sign up Page.....	3
1.4 Users Pages.....	4
1.5 Admin Pages.....	5
2.Database Connection.....	6
3.GUI Elements.....	7
3.1 Main Screen GUI.....	7
3.2 Login GUI.....	8
3.3 SignUp GUI.....	9
3.4 Get to Know Us GUI.....	10
3.5 Users GUI.....	11
3.6 AdminWindow GUI.....	13
3.6.1 CarRentalHistory GUI.....	15
3.6.2 ModificationOnCars GUI.....	16
4.Exception Handling.....	17
5.Testing and Validation.....	21
5.1 Testing Main Screen Buttons.....	21
5.2 Testing Interactions on the Log In Page.....	21
5.3 Testing Interactions on the Sing Up Page.....	22
5.4 Testing the Users Page.....	25
5.5 Testing the Admin Dashboard.....	27
6.Work Section.....	30

1. APPLICATION OVERVIEW :

1.1 Main Screen Page:

featuring three primary buttons: **Login** and **SignUp** and **Get to Know Us** .

The Get to Know Us button provides simplified information about the app in a new page.

1.2 Log in Page:

This page allows users to enter their username and password, with an option to select their user type (Customer or Admin). If the entered information is correct, you are redirected to the page corresponding to your role. If there are any errors, you will be notified

1.3 Sign up Page:

The Sign Up page allows new users to create an account on the application by providing their personal information, including full name, username, password, email, and phone number. This page ensures proper validation of the entered data and guides users through a secure registration process.

1.4 Users Page

The User Page is designed to provide users with a comprehensive interface for managing car rentals and viewing their booking history. The upper section allows users to load and view available cars by clicking the Load Available Cars button, which displays the cars in a table. A Search Car feature enables users to locate specific cars easily. Once a car is selected, users can book it using the Book Car button. The system then prompts the user to input a start date, ensuring it is valid and not in the past. After confirming a correct start date, the system requests an end date, which must be later than the start date. Once both dates are validated, the system presents a summary of the booking details and asks for confirmation. If the user confirms by selecting Yes, the booking is saved to the database, and the car becomes rented. If the user selects No or closes the confirmation dialog, the booking is not processed.

The lower section of the page focuses on rental history and active booking management. By clicking the Load My Rentals button, users can view their booking history, including active, completed and canceled rentals. For active rentals, users have the option to cancel a booking by selecting it and clicking the Cancel Rental button. The system then displays the car name, the remaining days of the booking, and a confirmation prompt. If the user confirms the cancellation by selecting Yes, the booking status is updated to "Canceled," and the car is made available again. Selecting No leaves the booking unchanged. Finally, the Logout button allows users to securely exit the page, ensuring a smooth and user-friendly experience for managing rentals.

1.5 Admin Pages:

Admin window provides the administrators an interface for car rentals management. At the top, this "All Cars" section gives him the table of all available vehicles showing, for each one, the make, model, year, type, color, and day rent price. The section below it, "Returned Cars," includes a list of returned cars with details such as booking ID, user ID, vehicle ID, start date, end date, status, and total price. The interface also provides three buttons at the bottom: "Modification On Cars" to add or remove cars, "View Rental History" to view the full history of rentals, and "Log Out" to log out securely.

In Modify page, the admin modifies the inventory of cars. Inside the "Add Car" section, it provides fields for typing the details of a new car. The field will require a car type, company, model, color, year, and daily rental price. This "Add Car" button saves the new car to the system. The "Delete Car" section shows a table of all the cars so that the admin selects and deletes a car when the "Delete Car" button is clicked. Also, the page has options for going back to the Admin Window by using "Back" button.

In Rental History page gives the administrator a detailed view of the rental history in the system. The interface includes a table that displays information on each rental transaction, including the unique ID of the rental, customer name, car model, rental and return dates, daily rental price, total price, and the booking status (e.g., active or canceled). Administrators can use this page to monitor all past and ongoing rentals, ensuring efficient management of the system. At the bottom of the page, a "Back" button allows administrators to return to the previous interface for further actions.

2. DATABASE

2.1 Database Connection

The Car Rental Software project uses MySQL as the database solution. The database is hosted on Amazon RDS (Relational Database Service), enabling consistent accessibility and eliminating the need for extensive local configurations. The connection between the application and the database was established using the MySQL Connector/J driver, integrated into the NetBeans IDE. This setup allows seamless communication between the Java-based application and the database.

A dedicated code module was developed to manage the database connection efficiently. The module includes a `connect()` method that initializes a secure connection using predefined credentials and a database URL. This reusable method ensures reliable access to the database and minimizes connection errors during runtime.

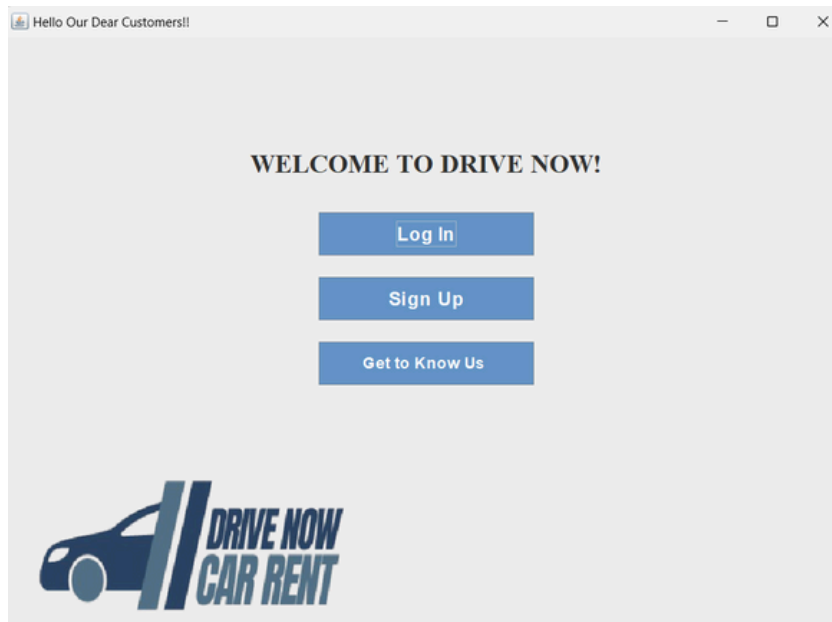
2.2 Database Description

The database for the Car Rental Software consists of four primary tables designed to manage users, vehicles, bookings, and reports efficiently:

1. **Users Table:** Stores user information, including login credentials, roles (customer/admin), and contact details.
2. **Vehicles Table:** Maintains details about the available vehicles, such as make, model, year, status (available/rented), and rental price per day.
3. **Bookings Table:** Tracks rental transactions, including the booking duration, associated user and vehicle, total cost, and booking status.

3. GUI ELEMENTS

3.1 Main Screen GUI:



1. Main Title Label:

- A central JLabel displaying the welcoming message:
- "WELCOME TO DRIVE NOW!" in a large, bold font.

2. Buttons:

- **Log In:** Redirects users to the login page.
 - **Sign Up:** Redirects users to the registration page.
 - **Get to Know Us:** Opens a new page with information about the app.
- These buttons are arranged vertically using BorderLayout for a neat, stacked appearance with proper spacing.

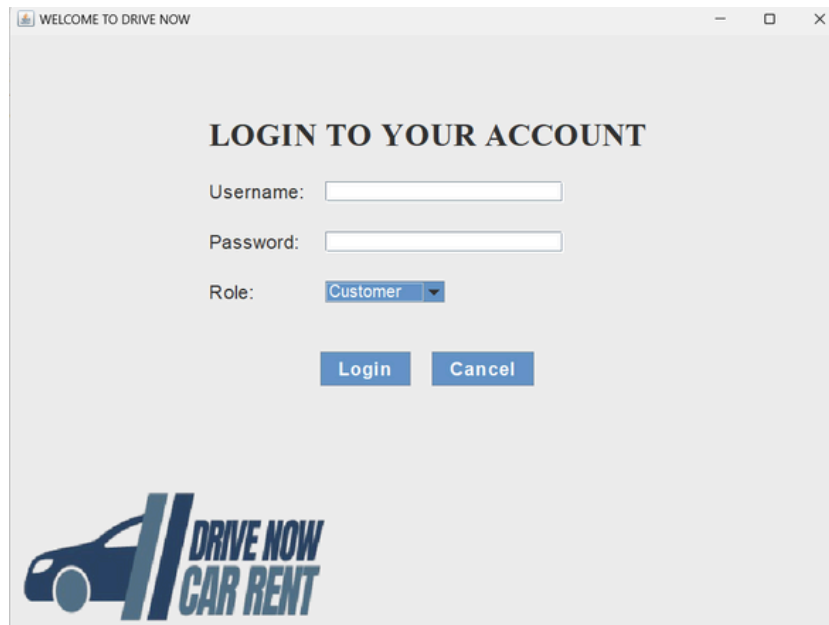
3. Image:

- An image is displayed at the bottom of the screen, aligned to the left, enhancing the visual appeal of the page.

4. Layout:

- **mainPanel:** The primary container using BorderLayout, holding the title, buttons, and image sections.
- **labelPanel:** The title label l1 is placed in the NORTH section with extra top padding.
- **buttonPanel:** Contains the buttons stacked vertically using BorderLayout, with spacing between them.
- **imagePanel:** Displays the welcome image at the bottom in the SOUTH section, with padding around it.

3.2 Login GUI:



1.Labels:

- **titleLabel:** Displays "LOGIN TO YOUR ACCOUNT" as the heading at the top of the login form.
- **usernameLabel:** Label for the username input field.
- **passwordLabel:** Label for the password input field.
- **roleLabel:** Label for the role selection dropdown (Customer/Admin).

2.Text Fields:

- **usernameText:** A text field where users can enter their username.
- **passwordField:** A password field where users can securely enter their password.

3.ComboBox:

- **cmbRole:** A dropdown menu where users can select their role (Customer or Admin).

4.Buttons:

- **login:** Validates the login credentials and redirects users based on their role:
Admin: Navigates to the Admin Window.
Customer: Navigates to the User Window.
- **cancel:** Redirects users back to the Main Menu page (FirstPage).

3.Image:

- An image is displayed at the bottom of the screen, aligned to the left, enhancing the visual appeal of the page.

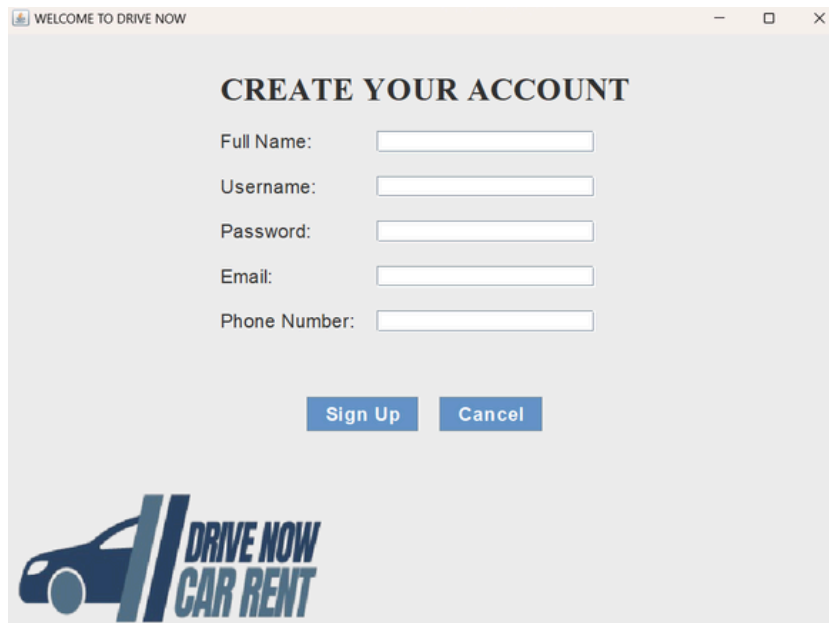
3.Panel:

- **formPanel** is the main container for the login form, using **GridBagLayout** to arrange the components. It includes:
 - **titleLabel** (displays "LOGIN TO YOUR ACCOUNT").
 - **Input fields:** username, password, and role.
 - **Buttons:** **loginButton** and **cancelButton**, arranged vertically and centered.

1.imagePanel:

- This panel is positioned in the **SOUTH** section of the frame and displays the welcome image aligned to the left, using **FlowLayout**.

3.3 SignUp GUI:



WELCOME TO DRIVE NOW

CREATE YOUR ACCOUNT


Full Name:

Username:

Password:

Email:

Phone Number:



1.Labels:

- titleLabel: Displays "CREATE YOUR ACCOUNT" as the title at the top of the form.
- fullNameLabel: Label for the full name input field.
- usernameLabel: Label for the username input field.
- passwordLabel: Label for the password input field.
- emailLabel: Label for the email input field.
- phoneLabel: Label for the phone number input field.

2.Text Fields:

- fullNameText: Input field for entering the full name.
- usernameText: Input field for entering the username.
- emailText: Input field for entering the email.
- phoneText: Input field for entering the phone number.

3.Password Field:

- passwordField: A field for entering the password.

4.Buttons:

- signUpButton: Submits the form, validates the data, and saves it to the database. If successful, redirects to the Users page.
- cancelButton: Redirects the user back to the Main Menu (FirstPage).

5.Panels:

• formPanel:

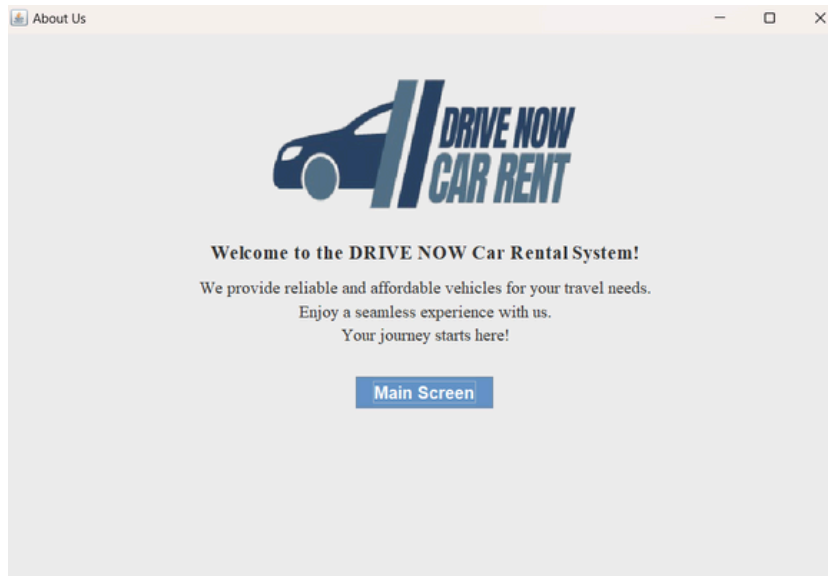
The main container for the sign-up form using GridBagLayout to arrange the labels, text fields, and buttons.

Contains titleLabel, input fields for full name, username, password, email, and phone, and the signUpButton and cancelButton.

• imagePanel:

- Positioned in the SOUTH section, this panel displays the welcome image. The image is aligned to the left using FlowLayout and includes padding

3.4 Get to Know Us GUI:



1.Labels:

- **aboutLabel:** Displays an introductory message about the Car Rental System. The text is centered and includes details about the service.
- **imageLabel:** Displays an image for branding or decoration. The image is centered at the top of the screen.

2.Buttons:

- **mainMenuButton:** Redirects users back to the Main Menu (FirstPage)

3.Panels:

- **mainPanel:**
 - The primary panel containing all components arranged vertically using BoxLayout.
 - Includes imageLabel (the image), aboutLabel (the text), and mainMenuButton (the button to navigate to the main menu).
- **imagePanel:**
 - Positioned in the SOUTH section to display the image at the bottom of the window.

3.5 Users GUI:

WELCOME TO DRIVE NOW

Search Car

Available Cars

ID	Make	Model	Type	Color	Price Per Day
----	------	-------	------	-------	---------------

Book Car Load Available Cars

My Rentals

Booking ID	Car	Start Date	End Date	Total Price	Status
------------	-----	------------	----------	-------------	--------

Load My Rentals Cancel Rental Logout

1.Labels:

- **AvailableCarsLabel:** Displays the heading "Available Cars" in bold.
- **MyRentalsLabel:** Displays the heading "My Rentals" in bold.

2.Buttons:

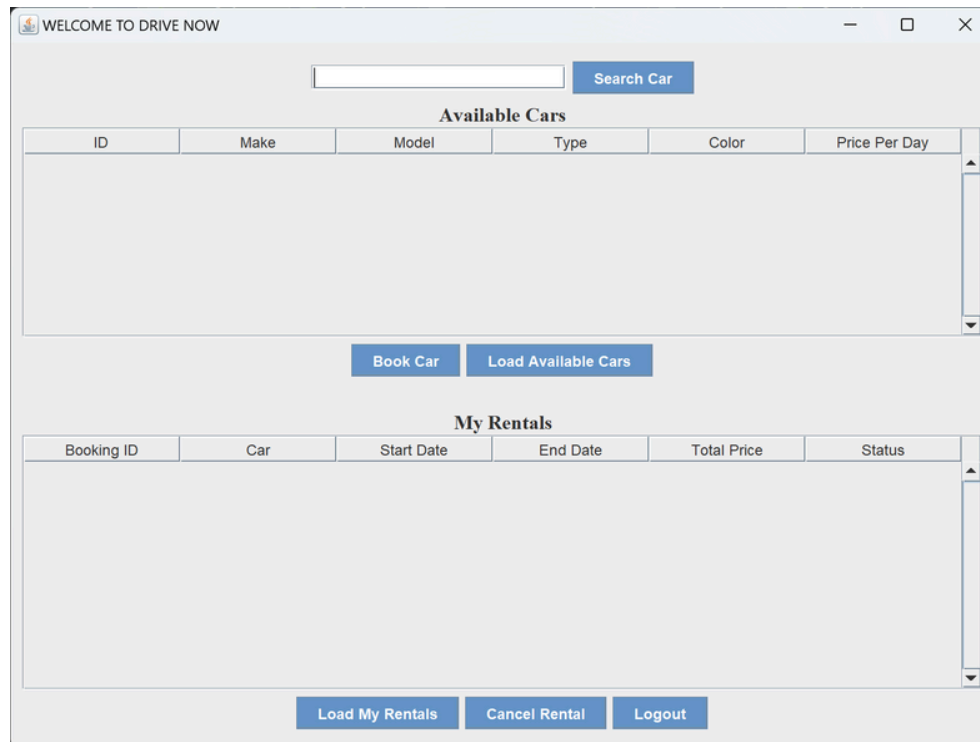
- **b1 ("Search Car"):** To search for a specific car.
- **b2 ("Book Car"):** To book a car.
- **b3 ("Load Available Cars"):** To load the list of all available cars.
- **b4 ("Load My Rentals"):** To load the list of rentals made by the user.
- **b5 ("Cancel Rental"):** To cancel an existing rental.
- **b8 ("Logout"):** To log out of the system.

3.Tables:

- **availableCarsTable:** Displays a list of available cars with columns:
 - ID, Make, Model, Type, Color, Price Per Day.
- **myRentalsTable:** Displays the user's rental details with columns:
 - Booking ID, Car, Start Date, End Date, Total Price, Status.

o

3.5 Users GUI:



4. Panels:

- **Available Cars Panel**
 - Contains:
 - A search panel (searchPanel) with the text field and "Search Car" button.
 - The AvailableCarsLabel.
 - A table (availableCarsTable) to display available cars.
 - Buttons: "Book Car" and "Load Available Cars".
- **My Rentals Panel**
 - Contains:
 - The MyRentalsLabel.
 - A table (myRentalsTable) to display user rentals.
 - Buttons: "Load My Rentals", "Cancel Rental", and "Logout".
- **Main Panel**
 - Combines:
 - AvailableCarsPanel on the top half.
 - MyRentalsPanel on the bottom half.

5. Text Field :

- t1 : A text field for searching cars.

3.6 AdminWindow GUI:

The screenshot shows a window titled "Admin Window" with a standard Windows-style title bar. Inside the window, there are two tables and three buttons at the bottom.

All Cars Table:

Make	Model	Year	Type	Color	Status	Price/Day
Mercedes-Benz	S-Class	2023	Sedan	Black	rented	500.0
BMW	7 Series	2023	Sedan	White	rented	450.0
Porsche	Cayenne	2023	SUV	Blue	available	550.0
Tesla	Model S	2023	Sedan	Red	available	600.0
Rolls-Royce	Phantom	2023	Sedan	Black	rented	1000.0
Audi	A8	2022	Sedan	Silver	available	380.0
Ferrari	Roma	2022	Coupe	Red	rented	1150.0
Maserati	Quattroporte	2022	Sedan	White	rented	780.0
Range Rover	Velar	2022	SUV	Silver	rented	630.0
Jaguar	F-Type	2022	Coupe	Green	rented	720.0

Returned Cars Table:

Booking ID	User Id	Vehicle Id	Start Date	End Date	Status	Total Price
------------	---------	------------	------------	----------	--------	-------------

Buttons:

- Modification On Cars
- View Rental History
- Log Out

1.Labels:

- **AdminWindow:** Displays the heading "Admin Window" as the title at the top of the form.

2.Buttons:

- **Modification Button :** Opens the "Modification On Cars" window to allow modifications to the car database.
- **History Button:** Opens the "Car Rental History" window to view rental history and Reporting module for tracking revenue,.
- **LogOut Button:** Logs out the admin and redirects to the login page.

3.Tables:

- **carTable:** Displays all cars in the database, showing columns for:
 - Make, Model, Year, Type, Color, Price/Day
- **returnedCarsTable:** Displays details of returned cars, showing columns for: Booking ID, User ID, Vehicle ID, Start Date, End Date, Status, Total Price

3.6 AdminWindow GUI:

The screenshot shows a window titled "Admin Window" with a standard Windows-style title bar. The window contains two tables and a button panel at the bottom.

All Cars Table:

Make	Model	Year	Type	Color	Status	Price/Day
Mercedes-Benz	S-Class	2023	Sedan	Black	rented	500.0
BMW	7 Series	2023	Sedan	White	rented	450.0
Porsche	Cayenne	2023	SUV	Blue	available	550.0
Tesla	Model S	2023	Sedan	Red	available	600.0
Rolls-Royce	Phantom	2023	Sedan	Black	rented	1000.0
Audi	A8	2022	Sedan	Silver	available	380.0
Ferrari	Roma	2022	Coupe	Red	rented	1150.0
Maserati	Quattroporte	2022	Sedan	White	available	780.0
Range Rover	Velar	2022	SUV	Silver	rented	630.0
Jaguar	F-Type	2022	Coupe	Green	rented	720.0

Returned Cars Table:

Booking ID	User Id	Vehicle Id	Start Date	End Date	Status	Total Price
------------	---------	------------	------------	----------	--------	-------------

Button Panel:

- Modification On Cars
- View Rental History
- Log Out

4.Panels:

- **mainPanel:**
 - Contains the carTable at the top for "All Cars."
 - Contains the returnedCarsTable at the bottom for "Returned Cars."
- **ButtonPanel:**
 - Positioned below the tables and contains the buttons: "Modification On Cars," "View Rental History," and "Log Out."

3.6.1 CarRentalHistory GUI:

Car Rental History

Rental History

Rental ID	Customer Name	Car Model	Rental Date	Return Date	Daily Price	Total Price	Booking Status
41	abdulrahman	BMW 7 Series	2024-12-11	2024-12-11	450.0	450.0	active
42	abdulrahman	Audi A8	2024-12-20	2024-12-31	380.0	4180.0	canceled
43	tala mohameed	Mercedes-Ben...	2024-12-11	2024-12-31	500.0	10000.0	active
44	layan	Rolls-Royce P...	2024-12-12	2024-12-12	1000.0	1000.0	active
45	layan	Ferrari Roma	2024-12-13	2024-12-20	1150.0	8050.0	active
46	layan	Jaguar F-Type	2024-12-25	2025-01-01	720.0	5040.0	active
47	hanan abdulra...	Audi A8	2024-12-11	2024-12-31	380.0	7600.0	canceled
48	abdulrahman	Range Rover V...	2024-12-12	2024-12-31	630.0	11970.0	active

Car ID	Car Model	Times Rented	Monthly Revenue
28	BMW 7 Series	1	450.0
32	Audi A8	2	11780.0
27	Mercedes-Benz S-Class	1	10000.0
31	Rolls-Royce Phantom	1	1000.0
33	Ferrari Roma	1	8050.0
36	Jaguar F-Type	1	5040.0
35	Range Rover Velar	1	11970.0
34	Maserati Quattroporte	1	7800.0

Month	Year	Total Revenue
DECEMBER	2024	56090.0

Back

1.Label:

- **titleLabel:** Displays the heading "Rental History" in bold.

2.Buttons:

- **back("Back"):** To back to Admin window page.

3.Tables:

- **historyTable:** Displays reservation details for cars and their status in a scrolling.
- **statsTable:** Shows the number of times the car has been booked and the total value of the bookings.
- **totalRevenueTable:** Monthly income calculation.

3.6.2 ModificationOnCars GUI

Add Car

Type: Model: Year:

Company: Color: Daily Rent Price:

Delete Car

Make	Model	Year	Type	Color	Price/Day
Mercedes-Benz	S-Class	2023	Sedan	Black	500.0
BMW	7 Series	2023	Sedan	White	450.0
Porsche	Cayenne	2023	SUV	Blue	550.0
Tesla	Model S	2023	Sedan	Red	600.0
Rolls-Royce	Phantom	2023	Sedan	Black	1000.0
Audi	A8	2022	Sedan	Silver	380.0
Ferrari	Roma	2022	Coupe	Red	1150.0
Maserati	Quattroporte	2022	Sedan	White	780.0
Range Rover	Velar	2022	SUV	Silver	630.0
Jaguar	F-Type	2022	Coupe	Green	720.0

1.Label:

- **AddCarLabel:** Displays the heading "Add Car" in bold.
- **DeleteCarLabel:** Displays the heading "Delete Car" in bold.

2.Buttons:

- **DeletB("Delete Car"):** To delete a car from data base.
- **BackB ("Back"):** To back to Admin window page.
- **AddCarB("Add Car"):** To add a new car to data base.

3.Tables:

- **carTable:** Displays all cars in data base.

4.Text Field:

- **TypeT:** To add a type of a new car.
- **ModelT:** To add a model of a new car.
- **CompanyT:** To add a company of a new car.
- **ColorT:** To add a color of a new car.
- **PriceT:** To add a of a new car.

5.Combo Box:

- **year:** Select the year of manufacture of the car.

4.EXCEPTION HANDLING:

4.1 Login Package:

EmptyException:

Thrown when the username or password fields are empty.

IncorrectLoginException:

Thrown when the login credentials (username, password, role) are incorrect.

SQLException:

Catches database-related errors, such as issues with connection or query execution.

4.2 SignUp Package:

EmptyFieldException:

Thrown when any of the required fields (Full Name, Username, Password, Email, or Phone Number) is left empty during the sign-up process.

InvalidFieldFormatException:

Thrown when a field does not match the expected format. For example, the Full Name field should only contain letters and spaces, and cannot contain numbers or special characters.

InvalidPhoneNumberException:

Thrown when the entered phone number does not meet the required format (e.g., it must start with "5" and be 8 digits long).

InvalidEmailException:

Thrown when the entered email address does not follow the correct email format (e.g., user@example.com).

InvalidPasswordException:

Thrown when the password does not meet the security requirements, such as containing at least one uppercase letter, one number, one special character, and being of a minimum length.

SQLException:

Catches database-related errors, such as issues with connection or query execution.

4.EXCEPTION HANDLING:

4.3 Users Package:

4.3.1 Class SearchCar:

- **CarNotFoundException:**
 - Thrown when no car matches the search query.
- **EmptySearchQueryException:**
 - Thrown when the search field is empty.
- **SQLException:**
 - This exception is thrown when there is an issue with database connection or query execution related to searching for cars. An error message with the exception details is displayed to the user.

4.3.2 Class LoadAvailableCars:

- **SQLException:**
 - Similar to SearchCar, this exception is thrown if there is an issue with database connection or query execution related to loading available cars. An error message with the exception details is shown to the user.

4.3.3 Class BookCar:

- **DateTimeParseException:**
 - This exception is thrown when the date format entered by the user is incorrect (e.g., entering a date in a format other than YYYY-MM-DD). A message is displayed prompting the user to enter the date in the correct format.
- **IllegalArgumentException:**
 - This exception is thrown when the user inputs invalid or unacceptable data (e.g., entering an invalid value). A message is shown to the user indicating that the input is incorrect.
- **SQLException:**
 - This exception is thrown if there is an issue with database connection or query execution related to booking a car. The user is shown a message with the exception details.

4.EXCEPTION HANDLING:

4.4 Admin Package:

4.4.1 loadCarsFromDatabase:

- **SQLException:**

- This exception is thrown when there is an issue with establishing a database connection or executing the SQL query to retrieve car data

4.4.2 Class Add:

- **ValidationException:**

- This custom exception is thrown if any of the input fields are empty or invalid. An error message is displayed to the user providing feedback on the validation error

- **SQLException**

- This exception is thrown if there is a database error while attempting to add a new car. The exception is caught, and a ValidationException is thrown with a specific message indicating the database error.

- **NumberFormatException:**

- This exception is thrown when the price input cannot be parsed into a valid number. The exception is caught, and a ValidationException is thrown with the message
"Invalid price. Please enter a valid number."

4.4.3 Class Delete:

- **ValidationException:**

- This custom exception is thrown if no car is selected for deletion or the deletion process fails. An error message is displayed to the user providing feedback on the validation error.

- **SQLException:**

- This exception is thrown if there is a database error while attempting to delete a car. The exception is caught, and a ValidationException is thrown with a specific message indicating the database error

4.EXCEPTION HANDLING:

4.4.4 CarRentalHistory Class:

- **SQLException:**
 - This exception is thrown if there is an issue with the database connection, the SQL query execution, or retrieving data from the database tables. An error message is displayed to the user the message "Error loading rental history" followed by the specific exception details

4.4.5 AdminWindow1 Class:

- **SQLException:**
 - This exception is thrown when input fields are empty or invalid, or when issues arise with establishing a database connection or executing SQL queries for fetching car data or completed bookings. In all cases, an error message is displayed to the user, providing feedback on the validation error or showing specific messages such as "Error loading cars" or "Error loading returned cars," along with detailed exception information.

5. TESTING AND VALIDATION:

5.1 Testing Main Screen Buttons:

All buttons in the **main Screen** were tested, and they functioned correctly as intended.

Button: Get to Know Us:

Action: Clicking this button navigated to the "GetToKnowUs" page seamlessly.

Additional Test: The button on the "GetToKnowUs" page successfully returned me to the previous page without issues. **Button: Log In:**

Action: When clicked, this button navigated directly to the "Log In" page. The transition was quick and smooth, with no issues or delays encountered.

Additional Test: On the "Log In" page, the Cancel button worked as expected, returning the user to the main Screen page without any issues.

Button: Sign Up:

Action: This button also functioned correctly, redirecting the user to the "Sign Up" page without any errors. The page transition was seamless, providing a positive user experience.

Additional Test: The Cancel button on the "Sign Up" page worked flawlessly, returning the user to the main menu without errors.

5.2 Testing Interactions on the LogIn Page:

Scenario 1: Entering Incorrect Information:

Test Steps:

- Entered an incorrect username, password, or role (e.g., a non-existent username or incorrect password).

Outcome:

- An error message was displayed, indicating that one or more of the provided details were incorrect. Specifically, the system threw an `IncorrectLoginException`, and the message "Invalid login credentials" was shown to the user.

5. TESTING AND VALIDATION:

Scenario 2: Leaving Fields Empty:

Test Steps:

- Left the username or password field empty, or left both fields blank.

Outcome:

- A warning message appeared, notifying the user that one or more required fields were not filled. The system threw an **EmptyException** with the message "Username and password cannot be empty", and the user was prompted to fill in the missing fields.

Scenario 3: Entering Correct Information for a User:

Test Steps:

- Entered valid credentials (username, password, and role as "Customer").

Outcome:

- A welcome message was displayed, greeting the user by name, such as "Welcome, [username]!", and indicating their role as Customer. Upon clicking the OK button, the system seamlessly redirected the user to the Users page, where they could interact as a customer.

Scenario 4: Entering Correct Information for an Admin:

Test Steps:

- Entered valid credentials (username, password, and role as "Admin").

Outcome:

- A welcome message was displayed, greeting the admin by name, such as "Welcome, [username]!", and indicating their role as Admin. Upon clicking the OK button, the system smoothly redirected the admin to the AdminWindow1, allowing them to manage administrative tasks.

5.3 Testing Interactions on the SignUp Page:

Scenario1: Leaving Fields Empty:

Test Steps:

- Left one or more fields (Full Name, Username, Password, Email, Phone Number) empty.

Outcome:

- The system displayed a warning message: "All fields are required."
- The user was prompted to fill in all required fields before proceeding.

5. TESTING AND VALIDATION:

Scenario2: Invalid Full Name Format:

Test Steps:

- Entered a full name containing non-alphabetical characters, numbers.

Outcome:

- An error message appeared: "Full name can only contain letters and spaces."
- The user was required to input a valid name format.

Scenario3: Invalid Password Format

Test Steps:

- Entered a password that does not meet the security requirements (e.g., missing uppercase letter, number, or special character).

Outcome:

- The system displayed an error message: "Password must include at least one uppercase letter, one number, and one special character."
- The user was prompted to update their password to meet the required security standards.

Scenario4: Invalid Phone Number Format:

Test Steps:

- Entered a phone number that does not start with the number "5" or had an incorrect length.

Outcome:

- The system displayed an error message: "Invalid phone number. It must be 9 digits and start with 5."
- The user was prompted to correct the phone number format.

Scenario5: Invalid Email Format:

Test Steps:

- Entered an invalid email format (e.g., missing "@" or domain).

Outcome:

- An error message was displayed: "Invalid email format."
- The user was prompted to enter a valid email address.

Scenario6: Password Similar to Username or Email:

Test Steps:

- Entered a password that was too similar to the username or email (e.g., containing the username or email in lowercase).

5. TESTING AND VALIDATION:

Outcome:

- The system displayed an error message: "Password should not be similar to username or email."
- The user was prompted to choose a different password.

Scenario7: Duplicate Email, Phone, or Password:

Test Steps:

- Attempted to register using an email, phone number, or password already registered in the system.

Outcome:

- The system checked the database and displayed an error message indicating the duplicate field (e.g., "Email is already registered. Please use a different one.").
- The user was required to enter a unique email, phone, or password

Scenario8: Entering Correct Information

Test Steps:

- Entered valid details (full name, username, password, email, phone number).

Outcome:

- The system validated all fields and successfully inserted the user information into the database.
- A success message was displayed with the user's full name and a confirmation of the successful sign-up.
- After clicking the "OK" button, the user was redirected to the Users page.

Scenario9: Cancel Button Functionality:

Test Steps:

- Clicked the "Cancel" button.
- Outcome:
- The system successfully returned the user to the previous page (the FirstPage).
- No data was saved, and the sign-up process was aborted.

5. TESTING AND VALIDATION:

5.4 Testing the Users Page:

Testing the Search Car Button:

Action:

- Clicking the "Search Car" button after entering a car name.

Outcome:

- If the search query matches an available car, the table displayed the car details (Make, Model, Type, Color, Price Per Day).
- If no matching car is found, a message "Car not available." was displayed.
- If the search field was left empty, an error message "Please enter a car name to search." was shown.

Testing the Load Available Cars Button:

Action:

- Clicking the "Load Available Cars" button.

Outcome:

- The table was updated with all available cars in the database.
- If the database connection failed, an error message was displayed.

Testing the Book Car Button:

Action:

- Clicking the "Book Car" button after selecting a car from the table.

Outcome:

- If no car was selected, a warning message "No car selected. Please select a car first." was displayed.
- Upon selecting a car, the user was prompted to input the rental dates.
- Valid Dates: Booking was successfully saved to the database, and the car's status was updated to "rented."
- Invalid Dates:
 - Entering a date in an incorrect format triggered an error message "Invalid date format. Please enter a valid date in the format YYYY-MM-DD."
 - Entering a start date in the past or an end date before the start date prompted specific error messages.

5. TESTING AND VALIDATION:

- After confirmation, the system displayed the total booking cost and successfully navigated to the Bill page.

Testing the Load My Rentals Button:

Action:

Clicking the "Load My Rentals" button.

Outcome:

- The table displayed all bookings made by the user, including car details, rental dates, total price, and booking status.
- If no rentals were found, the table remained empty.
- Database connection errors displayed appropriate error messages.

Testing the Cancel Rental Button:

Action:

- Clicking the "Cancel Rental" button after selecting a booking.

Outcome:

- If no booking was selected, a warning message "No car selected. Please select a car first." was displayed.
- If the selected booking was not active, a warning message "Only active bookings can be cancelled." was shown.
- Upon confirming cancellation, the booking status was updated to "cancelled," and the car's status was reverted to "available."
- If the user aborted the cancellation, no changes were made, and a message "The booking was not canceled." was displayed.

Testing the Logout Button:

Action:

- Clicking the "Logout" button.

Outcome:

- The user was successfully redirected to the FirstPage.

5. TESTING AND VALIDATION:

5.5 Testing the Admin Dashboard:

The Admin Window:

Testing the Modification On Cars Button:

Action:

- Clicking the "Modification On Cars" button.

Outcome:

- The system navigates to the "Modification On Cars" window successfully.
- The "Modification On Cars" window displays all required components, including text fields, labels, buttons, and a table for car details.

Testing the View Rental History Button:

Action:

- Clicking the " View Rental History " button.

Outcome:

- The system navigates to the "Car Rental History" window successfully.
- The "Car Rental History" window displays all user rental records correctly.
- Database connection errors are displayed as appropriate error messages.

Testing the "Log Out" Button:

Action:

- Clicking the "Log Out" button.

Outcome:

- The system logs out the admin user and redirects to the login page.
- No data is retained from the admin session after logout.

5. TESTING AND VALIDATION:

The CarRentalHistory Window:

Testing the Back Button:

Action:

- Clicking the "Back" button.

Outcome:

- After pressing this button, you will be returned to the administrator's home page.

The ModificationOnCarsWindow:

Adding a Car:

Scenario 1: Adding a Car:

Test Steps:

- Filled in all fields in the "Add Car" section (Type, Company, Model, Color, Year, and Daily Rent Price).
- Clicked the "Add Car" button.

Outcome:

- The system validated the input. If all fields were correctly filled, the new car details were added to the inventory table. A confirmation message, "Car added successfully!", was displayed. If any field was left blank or invalid, an error message appeared, "All fields are required. Please fill in all details."

Scenario 2: Leaving Fields Empty in Add Car

Test Steps:

- Left one or more fields in the "Add Car" section blank.
- Clicked the "Add Car" button.

Outcome:

- The system displayed a warning message, "All fields must be filled to add a car," and prevented the car from being added.

5. TESTING AND VALIDATION:

Scenario 3: Deleting a Car

Test Steps:

- Selected a car from the "Delete Car" table by clicking on its row.
- Clicked the "Delete Car" button.

Outcome:

- The selected car was removed from the inventory. A confirmation message, such as "Car deleted successfully!", was displayed. If no car was selected, the system prompted the user with a warning message like "Please select a car to delete."

Scenario 5: Clicking "Back"

Test Steps:

- Clicked the "Back" button at the bottom of the page.

Outcome:

- The system returned the user to the Admin Window interface.

6.WORK SECTION:

Name	Section work
Lamis Alharbi	<ul style="list-style-type: none">• Booking system allowing users to reserve and cancel rentals.• Pricing calculation based on rental duration and vehicle type.• Create the DataBase.
Layan Alswilem	<ul style="list-style-type: none">• Return processing and overdue management for rented vehicles.• Generate rental agreements and invoices for each transaction.• Notification system for booking confirmations and reminders.• GUI Design (Admin Window ,ModificationOnCars , Bill)
Hanan Alrashidi	<ul style="list-style-type: none">• User authentication system (log in,sgin up and registration for customers and admins).• Vehicle browsing and search functionality with real-time availability.• GUI Design (Main Screen, Log in , Sign up, Get to Know Us, Users)
Shahad Alharbi	<ul style="list-style-type: none">• Administrative dashboard to manage vehicle inventory and view rental history.• Reporting module for tracking revenue, bookings, and customer data.• GUI Design (CarRentalHistory)