

3ala Feen?

Name	Email	Id
AlaaElddin Ibrahim said	aladdin.salameh.2023@aiu.edu.eg	22101463
Rana HossamEldin Mohamed	rana.moussa.2023@aiu.edu.eg	22101478
Muhammed Mustafa Muhammed Farrag	mohamed.mustafa.2023@aiu.edu.eg	22101336
Ahmed Fathy Ahmed	ahmed.sweed.2023@Aiu.edu.eg	22101981
Noureen Yasser Hassan Muhammed	noureen.nuhammed.2023@aiu.edu.eg	22101109

Project description

- Problem Statement:

The transportation industry faces challenges such as inefficiencies in route planning, the unavailability of minibuses or buses, lack of real-time visibility, high operational costs related to comfort, and poor customer satisfaction.

This project aims to address these issues by developing a comprehensive software transportation system that improves the efficiency and management of transportation operations.

- **Project Aims:**

The main objective of this project is to design a Transportation Management System (TMS) that optimizes fleet operations, reduces transportation costs, and enhances customer satisfaction. The system will integrate features such as real-time tracking, route optimization, scheduling, security, and customer retention.

System Description

The proposed system will include features such as:

- **Real-time tracking:** Provides real-time visibility of vehicles and goods.
- **Route optimization:** Automatically calculates the most efficient routes.
- **Fleet management:** Manages vehicle scheduling, maintenance, and availability.
- **Dispatching and scheduling:** Automates task assignment and delivery schedules.
- **Performance analytics:** Generates reports on vehicle performance, route efficiency, and fuel usage
- **Retention:** Rent public wedding car or private car for your own ride

Silent Features of the System

- **intelligent Routing:** Uses AI-based algorithms to find the best routes, considering traffic and road conditions.
- **Automated Alerts:** Sends notifications to drivers and operators for route changes, maintenance needs, and delivery updates.
- **Cost Optimization:** Tracks and reduces fuel consumption, maintenance costs, and idle time.
- **Integration Capabilities:** Can integrate with GPS, ERP, and fuel management systems for smooth operations.
- **Payment Management:** Facilitates payment processing, invoicing, and integration with financial systems for customers and transport operators. It can support multiple payment methods (e.g., online payments, direct transfers) and manage billing for services.

Boundaries and Constraints

- **Geographical Limitations:** Initially, the system will be deployed within a limited region, expanding as the system matures.
- **Hardware Dependency:** The effectiveness of real-time tracking relies on the availability and accuracy of GPS and mobile network services.
- **Budget and Resource Constraints:** Development must adhere to predefined budget and time constraints.

- **Legal and Compliance:** The system will comply with local transportation laws and regulations, with adjustments required for deployment in new regions.

Business Requirements

1. Provide real-time tracking of vehicles to keep customers and managers informed.
2. Use smart scheduling and planning to make the best use of all vehicles.
3. Make the application easy to use, with multiple language options and flexible payments.
4. Grow revenue by expanding to new areas and handling more customers smoothly.
5. Keep customer information safe and follow the law to maintain trust.
6. Use reports and data to help managers make better decisions.
7. Stand out from competitors with unique features and great service quality.

Customer Requirements:

1. User Interface and Accessibility

1.1.2 The system makes it easy to use the payment methods.

1.1.3 The system makes it easy access to customer service feature.

1.1.4 The system makes it easy to detect the location on the map

1.1.5 The system supports multiple languages so customers can select their preferred language.

1.1.6 The system gives service accessible on smartphones and tablets.

2. Security and Compliance

2.1 System always keeps your data secure with encryption for sensitive information.

2.2 System is divergent in rules and regulations depending on your location city.

2.3 System should send authentication message multi-factor authentication (MFA) for all users to prevent unauthorized access.

3. Tracking and route optimization:

3.1 Tracking:

3.1.1 System should provide a live location of the vehicle on the map.

3.1.2 System should send notifications to the user that the vehicle has arrived or about to arrive at the user's destination.

3.2 Route Optimization:

3.2.1 System should display the estimated time for the vehicle to reach its destination and update it if there is traffic or delay.

3.2.2 System should calculate the fastest and shortest route to the user's destination to minimize the trip time.

3.2.3 System should allow the user to view alternate routes and choose the route he desires.

4. Payment and Invoicing:

4.1 Payment:

4.1.1 System should allow users to pay using multiple methods (credit card, digital wallets, cash).

4.1.2 System should send confirmation message through the app confirming the transaction.

4.1.3 System should allow the user to get a refund for cancelled trips.

4.1.4 System should support discounts or promo codes that user can use before booking trip.

4.2 Invoice:

4.2.1 System should send users invoices with the detailed fees and total amount charged.

4.2.2 System should allow users to access their invoice through the app and show the history of user's invoices with the dates for each invoice and its trip.

5.1 Fleet Maintenance and Scheduling:

5.1.1 System should automatically send a Notification for the car want maintenance based on the Kilometers traveled .

5.1.2 The system prioritizes vehicles with high complaints for repair.

5.1.3 Optimize fleet scheduling to ensure efficient vehicle utilization

5.2 Admin Management:

5.2.1 The system must show the details of the car and the number of trips when the admin searches for it.

5.2.2 The system should Displays you when the admin searches for a busy car .

5.2.3 The system should Displays you when the admin searches for a Available Car.

5.2.4 The admin can add a car to the system's maintenance list.

5.2.5 The admin can add a car to the system's fleet list

5.2.6 The admin can remove a car from the system's fleet list

5.2.7 The admin can add a driver to the system's fleet list

5.2.8 The admin can add a trip to the system's trip list

5.3 Driver Performance Evaluation:

5.3.1 The system should allow the user to rate each trip he used to rate the driver.

5.3.2 The user and driver can report lost items.

6.1 Customer Engagement and Support Requirements:

6.1.1 The system shall provide a live chat feature where customers can interact with support agents in real lifetime.

6.1.2 The system shall include an AI-driven help bot to assist users with frequently asked questions and general support inquiries.

6.1.3 The system shall offer feedback form for users to submit comments, suggestions, or report issues.

6.1.4 The system shall provide a searchable knowledge base of help articles, tutorials, and FAQs.

6.1.5 The system shall support multiple communications channels, including emails, chat, and phone support.

7. Integration and Scalability

7.1 Integration with GPS and Payment Gateways

7.1.1 System should support Intergration with GPS services for real-time tracking.

7.1.2 System should allow users to track the location of their booked vehicles.

7.1.3 The system should provide drivers with GPS navigation to optimize routes and improve service efficiency.

7.1.4 System should integrate with secure payment getaways for transactions.

7.1.5 System should support multiple payment options like credit card, digital wallets, and online banking.

7.1.6 Payment information should be processed securely.

7.1.7 Customers should be able to save their payment details for faster checkout.

7.2 Scalability to Support Different Types of Vehicles

7.2.1 System should handle various kinds of vehicles such as cars, buses, and motorcycles.

7.2.2 Admins should be able to add new types of vehicles and update vehicle details easily like changing prices or adding new features.

7.2.3 Each vehicle type should have its own details like seating capacity and license needs that the system can manage.

7.2.4 System should be flexible to change as needed

7.2.5 The system should allow admins to remove vehicles that are no longer in use.

7.2.6 The system should work well whether there are a few vehicles or many vehicles.

7.2.7 System should keep running smoothly during busy times.

7.2.8 System should manage many bookings and transactions at the same time without crashing.

7.2.9 The system should load quickly, even with lots of users online.

8. Automated Notifications and Reports:

8.1 Automated Notifications

- **8.1.1:** System must automatically notify customers regarding: - Booking status upon booking confirmation. - Estimated arrival times based on current dispatch schedules. - Delays, including causes and updated arrival times.

8.2 Fleet Performance and Cost Reports

- **8.2.1:** System must generate and display reports on:
 - Fleet performance metrics (e.g., efficiency, usage rates, maintenance records).
 - Operational costs, including fuel, repairs, and other expenses.
- **8.2.2:** Reports should support data export in formats like CSV or Excel for further analysis.

9.1 Vehicle Options:

9.1.1 The system must allow the user to rent a car.

9.1.2 The system must allow the user to rent a wedding car.

9.1.3 The system must allow the user to book a car ride.

9.1.4 The system must allow the user to book a Motorcycle.

9.1.5 The system must allow the user to book a bus trip.

9.1.6 The system must allow the user to book a full bus for a trip.

9.2 Booking and Reservation

9.2.1 The user must book a flight at a specific time and date.

9.2.2 The customer can add a promo code for a discount.

9.2.3 The customer can cancel the flight after booking within a specific time of 10 minutes.

9.2.4 The customer can book according to the economy class.

10. Account Management and control

10.1 Customer Account Management

10.1.1 Customers should have secure access to their accounts with password protection.

10.1.2 Customers should be able to update personal details, such as contact information and payment preferences.

10.1.3 Customers should have the option to view past transactions and ride history.

10.1.4 Customers should be able to save their payment details for future use.

10.2 Admin Control Over Accounts

10.2.1 Admins should be able to add and remove drivers from the system as needed.

10.2.2 Admins should have control over customer accounts, with options to enable, disable, or remove accounts as necessary.

10.2.3 Admins should be able to add or remove vehicles, updating the availability and types offered to customers.

10.2.4 Admins should have tools for monitoring customer and driver activities to ensure compliance and service quality.

Nonfunction

1. Security:

1.1 protection of the employee's personal data.

1.2 ensuring only employees of the same department can see the reports submitted by their colleagues.

2. Usability:

2.1 The employees should easily understand the system layout.

2.2 uploading and reviewing reports should be easily done .

3. Accessibility:

3.1 the system should be accessible from all devices (computers (Mac OS, Windows, Linux))

4. Availability:

4.1 the system should be available for a generous portion of the year (99.999% of the year).

5. Maintainability:

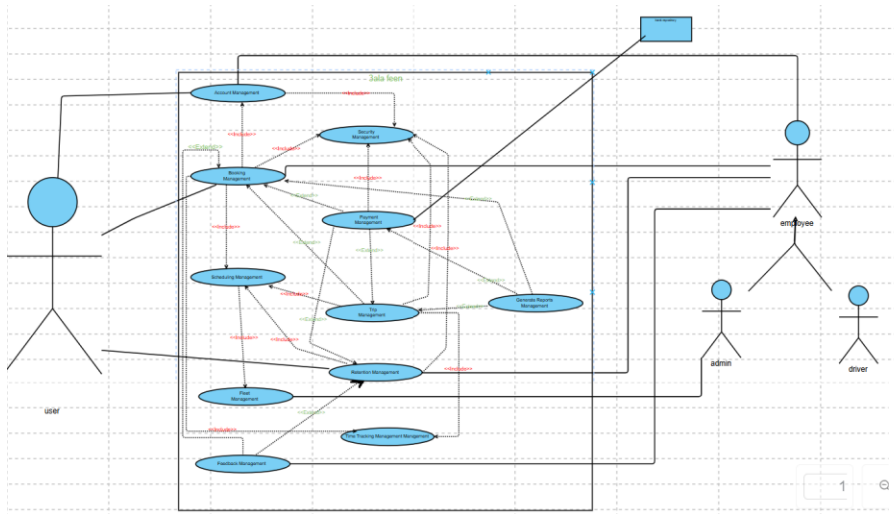
5.1 the system should be easily maintained to ensure low downtimes.

6. Scalability:

6.1 the system should be able to handle high workloads as the company grows through horizontal scaling.

(taken from best project)

USE CASE DIGRAM:



[DRAW LINK](https://online.visual-paradigm.com/login.jsp?r=/share.jsp?id=323731303133302d32)

<https://online.visual-paradigm.com/login.jsp?r=/share.jsp?id=323731303133302d32>

USE CASE Descriptions:

Use Case: Account Management

- **Actors:** User (initiator), Employee
- **Type:** Primary and essential
- **Description:**
 1. User opens "3ala Feen" system and selects the "Sign Up" option.
 2. User is presented with a registration form to fill out.
 3. User enters their personal information, including Full name, Email address and phone number.
 4. User creates a password, following the password policy (minimum 8 characters, including a number and special character).
 5. The system sends a verification email with a link or generates an OTP code sent to the user's email or phone number.
 6. User checks their email or phone inbox for the verification message.
 7. User either clicks the verification link or enters the OTP code in a designated field within the system
 8. The system verifies the link or OTP code and completes the registration process.
 9. Upon successful verification, the user account is activated, and the user is directed to the login page with a confirmation message.
 10. User logs into the system using their email and password.

11. Admin logs into the system using their email, password and hash code.

- **Preconditions:**

1. User must have a valid email address.
2. User must have a device with internet access.
3. User must not have any previous accounts banned or flagged for violations of the system's policies.
4. User must agree to the terms and conditions of the system.
5. User must provide a unique username that is not already in use by another account.
6. User must have access to their email account to complete the verification process.

- **Postcondition:**

1. User account is successfully created and activated.
2. User receives a confirmation email indicating successful account registration and activation.
3. User's personal information is securely stored in the "3ala Feen" system database.
4. User can now log into "3ala Feen" system.
5. User can access additional features such as booking a ride, rent a car and payment methods after logging in.
6. User can receive notifications regarding promotions, offers, and account activity via their registered email.
7. User can update their profile information after registration.
8. User is assigned a unique user ID for tracking and managing his account.

- **Alternate Scenarios:**

1. If the email address entered by the user is already associated with an existing account, the system prompts the user with a message indicating the email is already registered.
2. If the verification email is not received within a certain timeframe, the user can request a new verification email.
3. If a technical error occurs, the user is informed of the problem and advised to try again later.
4. If the registration form is incomplete, the system highlights the missing fields and prompts the user to complete them.
5. If there are network issues during registration, the user is alerted to check their connection.
6. If the username is already taken, the user must select a different username.

Use Case: Fleet Mangement

- **Actors:** admin (initiator), driver
- **Type:** primary and essential
- **Description:**
 1. Admin login in the app
 2. Admin add Hach code to change to admin mode
 3. Admin going to admin page
 4. Admin click on add new car
 5. Admin add car id, car name, care color, car type
 6. The car was added to the fleet.

Precondition:

- The admin should Loge in
- The admin should add Hach code right to open admin page

Postcondition:

- Admin add a car
- Admin remove a car
- Admin add driver
- Admin remove driver
- Admin know Number of Vehicles

Alternate Scenarios:

- The Hach code is not correct, and System will display error message
- The id of car is not correct System will display error message
- The driver's id is not correct System will display error message

Use Case: Payment Managment

- **Actors:** user (initiator), admin, driver, bank repository system
- **Type:** primary and essential
- **Description:**
 1. User logs into the app.
 2. User books a trip and confirms it.
 3. User clicks on "proceed to payment".
 4. The system proceeds the user to the payment page.
 5. The system presents the customer with multiple payment methods and the total fare.
 6. User chooses the payment method he wants.
 7. User confirms the payment by clicking on "Confirm Payment" button.
 8. The system communicates with the Bank Repository System to verify the payment.
 9. The system sends a confirmation message through the app saying, "Successful transaction".
 10. The system sends the user an invoice with the detailed fees and total amount charged.
- **Preconditions:**
 1. User has a valid account in the system.
 2. Bank Repository System is available and functioning.
 3. User has a valid payment method.
- **Postconditions:**
 1. The transaction is successful, and the user receives confirmation.
 2. The transaction details are saved in both the transportation system and the bank repository, so they can be referenced later if needed.
- **Alternate Scenarios:**
 1. The payment is declined by the bank repository system, customer gets notified with the problem.
 2. The user cancelled his booking and wants a refund.
 3. The user wants to retry the payment and choose a different payment method.
 4. The user wants to cancel his payment before it gets processed.

5. The payment is declined due to insufficient amount in the user's chosen payment method and system advise user to try again or choose a different method.

Use Case: Renting Management

- **Actors:** user (initiator), employee
- **Type:** primary and essential
- **Description:**
 1. The users can rent a vehicle from the system.
 2. The user can rent both a public wedding car and private car for personal use.
 3. The user can see vehicle availability and booking dates.
- **Preconditions:**
 1. The user must have a valid account and be logged into the system.
 2. Vehicles must be available for the requested dates.
 3. Rental policies (such as deposit, insurance, and identification) must be met.
- **Postconditions:**
 1. A booking confirmation is provided to the user.
 2. The vehicle is marked as "reserved" for the specified dates.
 3. Notifications may be sent to the employee for rental preparation.
- **Alternate Scenarios:**
- **Vehicle Unavailability:** If the requested vehicle is unavailable, the system suggests alternative dates or vehicles.
- **Multiple Rental Requests:** If the user wants to rent for both wedding and personal use, the system handles these requests separately, ensuring there is no scheduling conflict.
- **Payment Failure:** If payment fails, the system prompts the user to retry or choose another payment method.

Use Case: Scheduling Management

- **Actors:** user (initiator), employee
- **Type:** primary and essential
- **Description:**
 1. The user logs into the system and navigates to the "Booking" section, where they can request a vehicle rental.
 2. The user chooses their desired rental dates and selects the type of vehicle (e.g., wedding car or personal use). They may also specify any specific requirements, such as a particular model or additional services (like a chauffeur).
 3. After entering all necessary details, the user submits the booking request. The system processes the information and moves to the scheduling phase.
 4. The system checks if the requested vehicle and any required personnel (like a driver) are available for the selected dates. This step is automatic, and the user waits for the system's response.
 5. **If Available:** The system confirms the availability and displays a message that the booking is scheduled. The user receives confirmation with details about the vehicle, driver (if applicable), and scheduled dates.
 6. **If Unavailable:** If the selected vehicle or driver is not available, the system presents alternative options. This could include other available dates, different vehicles, or different drivers. The user reviews these options and selects a preferred alternative.
 7. Once the user agrees to the proposed schedule (or selects an alternative), they confirm the booking. The system finalizes the schedule, and the user sees a confirmation screen showing all booking details, including vehicle type, rental dates, pickup/drop-off location, and any special instructions.
 8. The system sends automated notifications to the user's email or mobile device. These notifications include: (Booking confirmation details, Reminders as the rental date approaches and Any changes to the schedule, such as driver assignment or unexpected vehicle maintenance.)
 9. If the user needs to make changes or cancel the booking, they can access the "My Bookings" section and select the scheduled booking. The system allows

users to request modifications, subject to availability, or cancel the booking entirely. The user receives updated notifications based on any changes.

10. After the rental period ends, the user may receive a final notification to confirm the end of the booking and prompt feedback. This closes the scheduling process for that booking.

- **Preconditions:**

1. Vehicles and drivers must be available in the system.
2. The user must have a confirmed booking request.
3. Scheduling policies (e.g., minimum booking notice, conflict resolution rules) must be in place.

- **Postconditions:**

1. The schedule is updated and finalized.
2. Notifications are sent to the user, employee, and driver (if applicable) confirming schedule details.
3. The system logs the schedule details for reporting and auditing purposes.

- **Alternate Scenarios:**

Resource Unavailability: If the requested vehicle or driver is unavailable, the system suggests alternative dates, vehicles, or drivers.

- **Conflict with Existing Schedule:** If a conflict arises, the system prompts the employee or admin to manually adjust the schedule or suggest alternative options to the user.
- **Cancellation or Rescheduling by User:** If the user needs to reschedule or cancel, the system updates availability and sends notifications to relevant actors.

Use Case: Booking Management

- **Actors:** User
- **Type:** Primary
- **Description:**
 1. User selects the booking option from the main menu.
 2. System displays available trips and booking details.
 3. User selects a trip and proceeds with booking.
 4. System verifies user account and payment information.
 5. System confirms the booking and provides details.

- **Cross Reference:** Account Management, Security Management, Payment Management.
- **Preconditions:**
 - The actor is authenticated and authorized.
- **Postconditions:**
 - The booking is created as requested.
- **Alternate Scenarios:**
 - If the Google Maps Repository is down, allow manual entry of location.
 - If there are security or data validation or payment issues, the system prompts correction.

Use Case: Trip Management

- **Actors:** User, Account Management, Scheduling Management
- **Type:** Primary
- **Description:**

This use case allows users to view, manage, and customize their planned trip

1. User accesses the trip management section.
 2. System displays the user's upcoming trips.
 3. Users can view details, add notes, or make changes to trip preferences.
 4. System updates trip information based on user inputs.
- **Cross Reference:** Account Management, Scheduling Management
 - **Preconditions:** User must have an existing booking.
 - **Postconditions:** Trip details are updated and saved in the system.

Alternate Scenarios:

- **Trip Cancellation:** If the user cancels a trip, the system updates the trip status and notifies relevant departments.
- **Changes Unavailable:** If changes are not allowed for a certain trip, the system informs the user.

interaction Scenarios:

Use Case: Account Management

Actor Intentions	System Responsibility
1.User opens "3ala Feen" application selects the "Sign Up" option	2.System displays the registration form.
3.User enters personal information (name, email, phone number)	4.System validates the format of the entered information.
5.User creates a password and confirms it.	6.System checks if the passwords match and meet security requirements.
7.User receives an OTP code and enters it.	8.System sends an error message if the OTP code entered is incorrect or expired.
9. User submits the registration form.	10.The system checks if the email is unique. If so, it sends a verification email to the provided address.
11. User either clicks the verification link or enters the OTP code.	12.System verifies it, activates the user account, and redirects user to the login page with a success message.

Alternative:

Step 6: If the passwords don't meet security requirements or aren't strong enough, the system asks the user to fix them before continuing.

Step 10: If the email is already used, the system tells the user and suggests logging in or resetting the password instead of creating a new account.

Use case: fleet management

Actor Intentions	System Responsibility
1.admin open app	2. System displays login or register
3. admin click to login and put Hach code, phone and opt	4. System display admin page

5.admin click to add driver	6. System display add driver page
7.admin add car info	8. Systeam add car to the fleet
9. admin remove a car	10. Systeam removed car from the fleet
11. Admin add driver	12.driver added to fleet
13.admin click on show number of vehicles	14. The Systeam display number of car
15.admin click get Available Car	16. The Systeam display number of car available

Alternative:

Step 8: if admin add car added before the error message will display this car added before

Step 6: if admin add driver added before the error message will display this driver added before

Use case: Renting Management

Actor Intentions	System Responsibilities
1. Admin accesses the retention management section.	2. System displays user engagement metrics and feedback details.
3. Admin reviews user engagement and feedback statistics.	4. System verifies if the selected user or group is eligible for retention.
5. Admin selects a user or group for retention efforts.	6. System sends personalized offers, feedback requests, or incentives to the selected users.
7. Admin sends personalized retention offers.	8. System logs the retention effort and tracks engagement response.

Alternative Scenarios

- **Step 4:** If the selected user is inactive, the system may mark the user for follow-up.
- **Step 6:** If there is insufficient feedback data, the system notifies the admin.

Use case: Scheduling Managment

Actor Intentions	System Responsibilities
1. Admin accesses the scheduling management section.	2. System displays the existing trip schedule, including resource allocation.
3. Admin views the current trip schedule.	4. System checks for any conflicts or overbooked resources.
5. Admin selects a trip to modify its schedule/resources.	6. System applies the schedule changes and reallocates resources as needed.
7. Admin saves the updated schedule.	8. System confirms and saves the updated schedule.

Alternative Scenarios

- **Step 4:** If a scheduling conflict or overbooking is detected, the system alerts the admin to resolve it.
- **Step 6:** If certain resources are unavailable, the system suggests alternatives or rescheduling options.

Use case: Booking Management

Actor Intentions	System Responsibility
1-User accesses the booking management platform.	2. System presents available trips, dates, and seating options.
3. User selects a trip and desired date.	4. System confirms seat or slot availability.
5. User provides account information for verification.	6. System verifies user account information and eligibility.
7. User selects the payment method and enters details.	8. System calculates total booking price and initiates payment processing.
9. User confirms booking and payment.	10. System authorizes payment, if applicable, and generates a digital confirmation.

11. User receives booking confirmation.	12. System sends confirmation via email or SMS, including QR code or ticket details.
13. User views or downloads digital confirmation.	14. System logs the booking transaction and reserves the selected seats or slots.

Alternative Courses

- **Step 10:** If payment fails, the system prompts the user to retry or select another payment method.
- **Step 4:** If no seats or slots are available, the system prompts the user to select another date or trip.

Use case: Trip Managment

Actor Intentions	System Responsibilities
1. User accesses the trip management section.	2. System displays all upcoming trips for the user.
3. User selects an existing booking to view or modify.	4. System checks if the selected trip is eligible for modifications.
5. User adds notes or changes preferences.	6. System saves any added notes or preferences.
7. User confirms any updates or modifications.	8. System updates the trip information and adjusts any related scheduling.
9. User saves the updated trip details.	10. System confirms and displays the updated trip details.

Alternative Scenarios

- **Step 4:** If modifications are not allowed for the selected trip, the system informs the user.
- **Step 6:** If the user attempts to add invalid information, the system prompts them to correct the input.

Use case: Payment Managment

Actor Intentions	System Responsibility
1. User opens the app and books a ride.	2. System displays available trips and calculates the estimated fare for that trip
3. User reviews the fare details and clicks on "Proceed to Payment".	4. System displays the available payment methods.
5. User selects a payment method and click "Confirm payment".	6. System sends the payment request to the bank repository for processing.
7. User waits for the transaction to be processed.	8. System sends a confirmation message to the user.
9. User receives message.	10. System sends the user an invoice with the detailed fees and total amount charged.

Alternative courses:

- **Step 4:** If the user decides to cancel the payment process, the user should click on "Cancel" button and the system will return the user to the booking page.
- **Step 6:** If the transaction is declined, the system notifies the user of the declined payment and gives the user the option to select a different payment method or to try again.

Subsystems:

Subsystem Name	Subsystem Function	Subsystem Interface
Real-time Tracking	1-This subsystem allows the customer to continually	Interface VehicleTracking { boolean

	<p>track the vehicle using GPS or other technologies</p> <p>2- Sending alerts to drivers or customers about delays or route changes.</p>	<pre>startTrackingVehicle(String vehicleID); boolean updateVehicleLocation(String vehicleID, String newLocation); } Interface AlertSystem { void sendAlert(String vehicleID, Alert newAlert); }</pre>
Route Optimization	<p>1-This subsystem calculates the most efficient routes for deliveries or transportation.</p> <p>2-It gives options to choose the optimal route for the customer/driver) based on Ai algorithm Like (Dijkstra algo, etc).</p>	<pre>Interface RouteCalculation { String calculateOptimalRoute(String startLocation, String endLocation); List<String> getSuggestedRoutes(Stri ng startLocation, String endLocation); } Interface RouteSelection { String selectCostEffectiveRout e(String origin, String destination); String dynamicReroute(String currentLocation, String destination); String optimizeForFuelEfficienc</pre>

		<pre> y(String origin, String destination); String planMultiStopRoute(List <String> stops); } </pre>
Fleet Management	<p>1-The subsystem allows admin knowing the number of cars in the fleet.</p> <p>2-knowing their status, whether they are Busy, if Busy who owns them now.</p> <p>3-Knowing the cars that need maintenance and what is wrong with them.</p> <p>4- can see the cars out of service and available.</p> <p>5-allows the admin to add and remove drivers.</p> <p>6- Allows the admin to add or remove cars.</p>	<pre> Interface FleetOverview { int getNumberOfVehicles(); Car SerchByinfo(String carid) } Interface CarAvailability { Car getAvailableCar(); Car getBusyCar(); String carStatus(String status); String searchByInfo(String carId, String carName, String color, String type); } Interface Car Maintenance { Void AddCarTOMaintenanceLIS T(String CarId ,String reason); </pre>

		<pre> Void Removecarmintanince(car id); } Interface DriverManagement { void addDriver(Driver driver); void removeDriver(String driverId); } Interface CarManagement { void addCar(String carId, String carName, String color, String type); void removeCar(String carId); } Interface MaintenanceTracking { void carNeedsMaintenance(S tring carId, String carBreakdown); } Interface ServiceStatus { Car carOutOfService(); } </pre>
--	--	---

Scheduling Management	<p>1-This subsystem is essential for coordinating transportation operations efficiently.</p> <p>2- It handles planning trips or deliveries with vehicles and drivers</p> <p>3- can adjust or cancel schedules.</p>	<pre> Interface DispatchCoordination { void dispatchVehicle(String vehicleId, String routeId); void cancelDispatch(String vehicleId); } Interface TripPlanning { void scheduleTrip(String tripId, String vehicleId, Date startTime, Date endTime); boolean isVehicleAvailable(String vehicleId, Date time); void assignDriver(String driverId, String vehicleId); String getTripStatus(String tripId); boolean isDriverAvailable(String driverId, Date time); String optimizeRoute(String origin, String destination); } Interface ScheduleAdjustment { void </pre>
------------------------------	--	---

		<pre> rescheduleTrip(String tripId, Date newStartTime, Date newEndTime); void cancelTrip(String tripId); } </pre>
Performance Analytics	<p>1-This subsystem monitors the performance of vehicles, route efficiency, costs.</p> <p>2-Logs the drivers</p>	<pre> Interface PerformanceMonitoring { Report generatePerformanceRe port(Date fromDate, Date toDate); CostReport analyzeOperationalCost s(Date from, Date to); EfficiencyData analyzeTripEfficiency(Str ing tripId); PerformanceData trackVehiclePerformanc e(String vehicleId); } Interface DriverEvaluation { DriverPerformanceData evaluateDriver(String driverId); void logVehicleIssue(String vehicleId, String issueDescription); } </pre>

Retention Management	<p>1-This subsystem allows users to rent a public wedding car or a private car for personal use.</p> <p>2- this subsystem manages vehicle availability, booking dates</p> <p>3- handling more than one rental requests (e.g., for a wedding and personal use).</p>	<pre> Interface PrivateCarRental { void rentPrivateCar(String userId, String carId, String date); void rentWeddingCar(String userId, String carId, String date); RetentionStrategyAnaly sis analyzeRetentionStrateg ies(); } Interface VehicleAvailability { boolean checkCarAvailability(Stri ng carId, String date); } Interface CustomerRetention { RetentionData trackCustomerRetention (String customerId); } </pre>
Payment Management	<p>1-This subsystem enables customers to choose from multiple payment options, including cash, credit cards, and mobile payment apps.</p> <p>2- It effectively manages payment errors and offers discounts as applicable.</p>	<pre> Interface PaymentOptions { boolean setupCreditCardPaymen t(String API_key, String public_key, String secret_key); boolean </pre>

	<p>3-Additionally, it validates transactions by sending detailed payment receipt emails to customers, ensuring they receive complete information about their payments.</p> <p>4-The subsystem also alerts customers regarding payment confirmations, declines, and refunds, keeping them informed throughout the transaction process</p>	<pre>processPayment(String paymentDetails); }</pre> <pre>Interface PaymentErrorHandling { boolean refundPayment(String transactionId); boolean applyDiscount(String userId, String discountCode); }</pre> <pre>Interface TransactionValidation { boolean sendPaymentReceipt(String email, String receiptDetails); }</pre> <pre>Interface CustomerNotifications { void sendPaymentAlert(String userId, String alertType); }</pre>
Security Management	<p>1-This subsystem allows continuously monitors vehicles from inside and outside,</p> <p>2- checks for any security breaches,</p>	<pre>Interface VehicleMonitoring { void monitorVehicle(String vehicleId); }</pre>

	<p>3- sends alerts when unauthorized activities are detected.</p> <p>4- Incorporating a GPS tracking tool for real-time vehicle tracking.</p>	<pre> Interface SecurityBreachDetection { boolean detectSuspiciousActivity (String vehicleId); void reportIncident(String userId, String vehicleId, String description); String encryptData(String plainText); String decryptData(String encryptedText); } Interface AlertSystem { void sendAlert(String message, String[] recipients); } Interface GPSTracking { String getRealTimeLocation(Str ing vehicleId); String getVehicleMovementHis tory(String vehicleId, Date from, Date to);} </pre>
Account Management	<p>1-This subsystem allows users to sign up or login.</p>	<pre> Interface UserAccountManageme nt { void addAccount(User user); void </pre>

	<p>2- Allows admins to sign up or login.</p> <p>3- Allows users to update, create and delete accounts.</p> <p>4. Allows the admin to manage customer accounts, performing different account actions.</p>	<pre> updateAccount(String accountId, User updatedUser); void removeAccount(String accountId); void userLogin(String userEmail, String userPassword); void adminLogin(String userEmail, String userPassword, String adminHashCode); void signup(,String userName, String userPhoneNumber, String userEmail, String userPassword, String otp); } Interface CustomerAccountHandli ng { void manageCustomerAccou nt(String customerId, AccountAction action); } </pre>
--	--	--

Traceability Matrix:

Features	Real time Tracking	Route Optimization	Fleet Management	Dispatching and Scheduling	Performance Analytics	Retention Management	Payment Management	Security Management	Account Management
using GPS or other technologies	✓	✓		✓				✓	
Ai algorithm		✓		✓	✓	✓		✓	
Fleet scheduling and maintenance	✓		✓	✓	✓				
Supporting Technical issues					✓			✓	✓
Tracks and analyzes expenses related to the fleet	✓		✓						
Payment handling and invoicing							✓		✓
Vehicle Tracking and Monitoring	✓							✓	
Compliance Management			✓						✓
handling rental requests						✓			
Managing customer accounts			✓						✓

and admin control over drivers and vehicles									
Optimized route planning to reduce travel time and costs		✓		✓					
Performance reports on fleet operations					✓				
Sending automated alerts for route changes, maintenance needs	✓			✓				✓	
Customer rating for drivers					✓	✓			
Security monitoring and handling security breaches	✓	✓						✓	
Integration with GPS and payment	✓	✓					✓	✓	

Test requirements:

Use case : fleet management (1)

- 1.1 Validate the admin can add a car
- 1.2 Validate the admin cannot add a car with id saved in database
- 1.3 Validate the admin can remove a car in saved in data base
- 1.4 Validate the admin cannot remove a car not in data base
- 1.5 Validate the admin can add a drive
- 1.6 Validate the admin cannot add a drive with id saved in database
- 1.7 Validate the admin can remove a drive
- 1.8 Validate the admin can't remove a car not in data base
- 1.9 Validate the admin add a trip
- 1.10 Validate the admin add a trip in future
- 1.11 Validate the admin can't add a trip in bast date
- 1.12 Validate the admin can't add a trip out of zone
- 1.13 Validate the admin can edit the trip
- 1.14 Validate the admin can't edit the trip to past date
- 1.15 Validate the admin can't edit the trip to out of zone
- 1.16 Validate the admin can get number of vehicles in fleet
- 1.17 Validate the admin should know car Status by search by id
- 1.18 Validate the admin put wrong data in search
- 1.19 Validate the admin should search By Info
- 1.20 Validate the admin can add car to maintenance list
- 1.21 Validate the admin can't add car to maintenance list if it is added before

Use Case: Account Management

1. Validate User Sign-Up

- 1.1 Validate user cannot sign up with valid phone number, email, password, and OTP but an invalid username.

1.2 Validate user cannot sign up with a valid username, email, password, and OTP but an invalid phone number.

1.3 Validate user cannot sign up with a valid username, phone number, password, and OTP but an invalid email.

1.4 Validate user cannot sign up with a valid username, phone number, email, and OTP but an invalid password.

1.5 Validate user cannot sign up with a valid username, phone number, email, and password but an invalid OTP.

1.6 Validate user can successfully sign up with a unique username, valid phone number, valid email, valid password, and valid OTP.

1.7 Validate user cannot sign up with empty credentials.

2. Validate User Login

2.1 Validate user cannot login with valid username and invalid password.

2.2 Validate user cannot login with invalid username and valid password.

2.3 Validate user can login with valid username and valid password

2.4 Validate user can't login with empty credentials.

33. Validate Admin Authentication

3.1 Validate admin cannot login with valid password and email and invalid hash code.

3.2 Validate admin cannot login with valid hash code and email and invalid password.

3.3 Validate admin cannot login with valid password and hash code and invalid email.

3.4 Validate admin can login with valid password, email and hash code.

3.5 Validate admin cannot login with empty credentials.

Use Case: Renting Management

1. Verify the system allows the user to log in and access rental features.
2. Verify that users can view available vehicles based on selected rental type and dates.

3. Verify the system suggests alternatives if the requested vehicle is unavailable.
4. Verify that users can proceed with multiple rental requests without conflicts.
5. Verify the system checks rental policy compliance (e.g., deposit and document submission).
6. Verify that payment processing is functional, and the system handles payment failures appropriately.
7. Verify booking confirmation and vehicle reservation for selected dates

Use Case: Scheduling Management

1. Verify the admin can update the trip schedule.
2. Validate the system alerts for overbooking.
3. Verify the System Prevents Scheduling Conflicts
4. Verify System Sends Notifications for Schedule Changes
5. Verify System Handles Multiple Bookings on a Single Trip
6. Verify the System Calculates Availability Based on Cancellations

Use Case Booking Management:

1. Validate successful booking creation.
2. Validate the integration with Google Maps for location accuracy and validation.
3. Validate error handling for unavailable Google Maps services.
4. Validate that user adds valid information.
5. Validate data security for booking details in transit and storage.
6. Validate that payment processing works as expected.
7. Validate the system handles booking unavailability.
8. Validate that unauthorized users cannot access bookings.

• Use Case Trip Management:

1. Verify that a user can view and modify trip details.
2. Validate that cancellations are processed correctly.
3. Verify Users Can Reschedule Trips

4. Confirm Notifications Are Sent for Changes in Trip Details
5. Verify Trip Details Are Valid and Complete
6. Verify Only Authorized Users Can Modify Trip Details
7. Verify System Allows for Trip Cancellation and Reimbursement Processing
8. Verify Trip Completion Updates User's Trip History

- **Use Case: Payment Managment**

1. Validate that system proceeds user to payment page after booking.
2. Validate that system displays the available payment methods to the user.
3. Validate that system processes the payment through the bank repository system securely.
4. Validate that system sends a confirmation message to user upon successful transaction.
5. Validate that system allows user to choose another payment method or to retry after payment failure.
6. Validate that system allows user to cancel payment during processing.
7. Validate that the system provides the option for the user to request refunds for transactions.
8. Validate that system should send users invoices with the detailed fees and total amount charged.
9. Validate that user can view his invoice history through the system app.
10. Validate that system offer promo codes or discounts for applicable users before payment.

Test Case:

For FLeet management use case (1)

Id	test scenario	test steps	test data	Expect / Actual	p/f
tst1	1.20	1-open page 2- click on login	#valid car id	Pass /Pass	p

		3- put your email, password and Hach code 4- go to admin page 5- click on add to maintenance car and put car data 6- click add			
tst2	1.21	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add to maintenance car and put car data 6- click add	#car id add to maintenance before	pass/ Error cant add car add before	F
tst3	1.17	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on search and put car data 6- click add	#valid car id	Pass/displ ay car info	p
tst4	1.1	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add car and put car data 6- click add	#car123 #honday #black #sedan	Pass/the car added	p
tst5	1.2	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add car and put car data 6- click add	#car123 #honday #black #sedan	PASS/ this car saved before	F

tst6	1.3	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on remove car and put car data 6- click remove	#car123	Pass/the care deleted	p
tst7	1.4	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on remove car and put car data 6- click remove	#car333	Pass/ the care not in data base	f
tst8	1.5	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add driver and put driver data 6- click add	#palstaine #303070388 #3/7/2003 #car123	Pass/the driver adds in data base	pass
tst9	1.6	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on remove driver and put driver data 6- click remove	#palstaine #303070388 #3/7/2003 #car123	Pass/ Error can't add user added before	f
tst10	1.7	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on remove driver and put driver data 6- click remove	#303070388	Pass/the driver removed	p

tst11	1.8	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on remove driver and put driver data 6- click remove	#303070399	Pass/ this driver saved before	p
tst12	1.14	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add trip and put data 6- click remove	#cairo #alx #1/1/2000 #giza # Damanhur	Pass/ the date in past time	f
tst13	1.10	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add trip and put data 6- click remove	#cairo #alx #1/1/2025 #giza # Damanhur	Pass/ will add the trip	p
tst12	1.14	1-open page 2- click on login 3- put your email, password and Hach code 4- go to admin page 5- click on add trip and put data 6- click remove	#cairo #alx #1/1/2025 #dubai # Damanhur	Pass/ one point out of zoon	f

For account management use case (2)

ID	test scenario	test steps	test data	Expected / Actual	p/f
----	---------------	------------	-----------	-------------------	-----

T1	1.1	1.Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username. 4. Enter phone number. 5. Enter email. 6.Enter password. 7.Enter OTP code.	Username: valid username Phone number: valid phone number Email: valid email Password: valid password OTP: invalid otp	Error message "Incorrect or expired OTP" / User can't create his account.	F
T2	1.2	1.Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username. 4. Enter phone number. 5. Enter email. 6.Enter password. 7.Enter OTP code.	Username: valid username Phone number: invalid phone number Email: valid email Password: valid password OTP: valid otp	Error message "Invalid phone number" / User can't create his account.	F
T3	1.3	1. Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username. 4. Enter phone number. 5. Enter email. 6.Enter password.	Username: valid username Phone number: valid phone number Email: invalid email Password:	Error message "Invalid email format" / User can't create his account.	F

		7.Enter OTP code.	valid password OTP: valid otp		
T4	1.4	1. Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username. 4. Enter phone number. 5. Enter email. 6.Enter password. 7.Enter OTP code.	Username: valid username Phone number: valid phone number Email: valid email Password: invalid password OTP: valid otp	Error message "Invalid password" / User can't create his account.	F
T5	1.5	1. Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username. 4. Enter phone number. 5. Enter email. 6.Enter password. 7.Cheks for OTP code to enter it.	Username: valid username Phone: valid phone number Email: valid email Password: valid password OTP: invalid otp	Error message "Invalid OTP" / User can't create his account. .	F
T6	1.6	1. Open "3ala Feen" app 2.Selects the "Sign Up" option. 3.Enter username.	Username: valid username Phone: valid phone number	User successfully signed up. / User has an account.	P

		4. Enter phone number. 5. Enter email. 6. Enter password. 7. Checks for OTP code to enter it.	Email: valid email Password: valid password OTP: valid otp		
T7	1.7	1. Open "3ala Feen" app 2. Selects the "Sign Up" option. 3. Enter username. 4. Enter phone number. 5. Enter email. 6. Enter password. 7. Enter OTP code.	Username: empty Phone number: empty Email: empty Password: empty OTP: empty	Error message "Credentials are required" / User can't create his account	F
T8	2.1	1. Open "3ala Feen" app 2. Selects the "Login" 3. Enter email. 4. Enter password	Username: valid username Password: invalid password	Error message "Invalid username or password". / User can't login.	F
T9	2.2	1. Open "3ala Feen" app 2. Selects the "Login" 3. Enter email. 4. Enter password	Username: invalid username Password: valid password	Error message "Invalid username or password". / User can't login.	F
T10	2.3	1. Open "3ala Feen" app 2. Selects the "Login" 3. Enter email. 4. Enter password	Username: valid username Password: valid password	User successfully logged in. / User can access his account	P

T11	2.4	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password	Username: empty Password: empty	Error message "Username and password are required". / User can't login	F
T12	3.1	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password 5.Enter hash code	Username: valid username Password: valid password Hash code: invalid hash code	Error message "Invalid hash code". / Admin can't login.	F
T13	3.2	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password 5.Enter hash code	Username: valid username Password: invalid password Hash code: valid hash code	Error message "Invalid username or password". / Admin can't login.	F
T14	3.3	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password 5.Enter hash code	Username: invalid username Password: valid password Hash code: valid hash code	Error message "Invalid username or password". / Admin can't login. .	F

T15	3.4	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password 5.Enter hash code	Username: valid username Password: valid password Hash code: valid hash code	Admin successfully logged in. / Admin can access his account.	P
T16	4.5	1.Open "3ala Feen" app 2.Selects the "Login" 3.Enter email. 4.Enter password 5.Enter hash code	Username: empty Password: empty Hash code: empty	Error message "Username, password and hash code are required" / Admin can't login.	F

For Booking Management use case (3)

Id	test scenario	Test steps	test data	Expected	A c t u a l	pa ss/ fail
TC-01	Attempt to book a trip with valid details and confirm booking.	1. Access booking platform. 2. Select trip and date. 3. Enter valid account and payment details. 4. Confirm payment.	Trip ID: T100, Date: 2024-11-15, Card: 1234-5678-9012-3456, Exp: 12/25, CVV: 123	Booking is successfully completed, confirmation is displayed, and ticket/QR code is generated.		

TC-02	Attempt booking with invalid payment details and check for error.	1. Access booking platform. 2. Select trip. 3. Enter invalid payment details. 4. Confirm payment.	Card Number: 0000-0000-0000-0000	System displays error message, prompting user to retry payment or use a different payment method.		
TC-03	Try booking an unavailable trip and verify the alternative suggestion.	1. Access booking platform. 2. Select an unavailable trip. 3. Attempt to proceed with booking.	Trip ID: T101 (unavailable)	System shows that trip is unavailable and suggests alternative trips, dates, or seats for user selection.		
TC-04	Validate Google Maps integration	1. Access booking system. 2. Enter pickup and drop-off locations. 3. Verify map displays location correctly.	Pickup: Cairo, Drop-off: Alexandria	Google Maps accurately displays and validates the entered locations.		
TC-05	Validate error handling for Google Maps unavailability	1. Simulate Google Maps service unavailability. 2. Attempt to enter pickup and drop-off locations.	N/A	System displays an error message and suggests trying again later.		
TC-06	Validate data security in transit	1. Begin booking process. 2. Capture and monitor network traffic. 3. Complete booking.	Booking details: Pickup, Drop-off, Dates	Booking data is encrypted during transmission		

TC-07	Validate data security in storage	1. Complete booking. 2. Check database to verify data encryption at rest.	Booking details: Pickup, Drop-off, Dates	Booking data is stored securely with encryption.		
TC-08	Validate handling of booking unavailability	1. Select fully booked vehicle and dates. 2. Attempt booking.	Vehicle: ID123, Date: 2024-12-01 to 2024-12-05	System displays message that vehicle is unavailable and suggests alternatives.		
TC-09	Validate access control for unauthorized users	1. Attempt to access booking features without logging in.	N/A	System prevents access and redirects to login page.		

For Renting Management use case (4)

Id	test scenario	Test steps	test data	Expected	A c t u a l	p a s s / f a i l
TC-02	Verify login and access to rental features	1. Go to login page. 2. Enter valid username and password. 3. Submit login form.	Username: user1, Password: password123	User is logged in and has access to rental features.		
TC-03	Verify available vehicles display correctly	1. Log in to the system. 2. Select rental type (e.g.,	Rental Type: Wedding, Date: 2024-11-05 to 2024-11-10	System displays list of available		

		wedding or personal use). 3. Choose rental dates.		vehicles for selected type and dates.		
TC-04	Verify alternative suggestion for unavailable vehicle	1. Log in and select a vehicle that is already booked for the selected dates. 2. Attempt to proceed with booking.	Vehicle: ID456, Date: 2024-11-15 to 2024-11-18	System displays message indicating vehicle unavailability and suggests alternative dates or vehicles.		
TC-05	Verify multiple rental requests without conflicts	1. Log in and initiate a wedding car rental for a specific date. 2. Initiate another personal car rental for a different date. 3. Submit both requests.	Rental 1: Wedding, Date: 2024-11-20 Rental 2: Personal, Date: 2024-11-25	System processes both rentals separately without conflicts, ensuring no scheduling overlap.		
TC-06	Verify rental policy compliance	1. Log in and initiate a rental. 2. Attempt to proceed without meeting policy requirements (e.g., missing deposit or document upload).	Rental Type: Personal, Missing Deposit	System displays an error message and prompts user to complete required policy items.		
TC-07	Verify booking confirmation and vehicle reservation	1. Complete booking details and payment. 2. Confirm booking.	Rental Date: 2024-11-25 to 2024-11-30	System confirms booking and reserves vehicle for		

				specified dates.		
TC-08	Verify reservation notification to employee	1. Complete booking. 2. Check employee notifications.	Rental ID: 789	System sends a notification to the employee for rental preparation.		
TC-09	Book a vehicle that is already rented out	1. Log in as a user. 2. Search for a specific vehicle. 3. Attempt to book the vehicle during dates it is already rented.	Vehicle ID: 78901, Dates: 2024-12-01 to 2024-12-03	System shows a message that the vehicle is unavailable and suggests alternatives.		
TC-10	Book multiple rentals with no schedule conflicts	1. Log in as a user. 2. Book two different vehicles for overlapping periods. 3. Confirm each booking.	Vehicle ID: 12345 (Wedding Car), Dates: 2024-12-15 to 2024-12-17 Vehicle ID: 23456 (Personal Car), Dates: 2024-12-15 to 2024-12-17	System successfully books both rentals without conflicts and confirms each booking.		

For Payment management use case (5)

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
PT001	Validate that the system proceeds the user to the payment page after booking.	1. Book a ride using the transportation app. 2. Click on "Proceed to Payment" button.	User Email: validEmail Password: validPass Booking details:	System proceeds the user to payment page successfully		

			Pickup, Drop-off, Dates			
PT002	Validate that the system displays the available payment methods to the user.	1. Navigate to the payment page after booking. 2. observe available payment methods.	User Email: validEmail Password: validPass	The system should display a list of available payment methods.		
PT003	Validate that the system processes the payment through the bank repository system securely.	1. User selects a payment method. 2. User enter payment details and clicks on "Confirm Payment" button.	Valid Payment Method: credit card Card Details: 4111 1111 1111 1111 / Exp: 12/25 / CVV: 123	The system should process the payment securely and confirm the transaction.		
PT004	Validate that the system sends a confirmation message to the user upon successful transaction.	1. User completes the payment process. 2. User checks for confirmation message.	User Email: validEmail	The system should send a confirmation message through the app.		
PT005	Validate that the system allows the user to choose another payment method or to retry after payment failure.	1. System processes the payment. 2. Payment failures (e.g., insufficient funds)	Payment Method: Invalid card details	The system should display options to retry or select an alternative payment method.		
PT006	Validate that the system allows the user to cancel payment during processing.	1. User starts payment process. 2. User clicks on "Cancel" button.	User Email: validEmail Password: validPass	The system should abort the payment process and return to the previous screen.		
PT007	Validate that the system provides the	1. User navigates to transaction history.	User Email: validEmail Password:	The system should allow the user to		

	option for the user to refund transactions.	2. User selects a completed transaction and clicks on "Request Refund".	validPass Transaction ID: 123456	submit a refund request for the selected transaction.		
PT008	Validate that the system sends users invoices with detailed fees and total amount charged.	1. User completes a payment transaction. 2. User checks his email for the invoice.	User Email: validEmail Transaction ID: 232323	The system sends an invoice to the user detailing all fees and the total amount charged.		
PT009	Validate that the user can view their invoice history through the system app.	1. User proceed to the invoice history section in app. 2. User views all past invoices.	User Email: validEmail Password: validPass User past invoices	The system should display a complete history of the user's invoices.		
PT010	Validate that the system offers promo codes or discounts for applicable users before payment.	1. User books a ride and proceeds to payment page. 2. User enters a promo code and applies it.	User Email: validEmail Password: validPass Valid Promo Code: SUMMER24	The system should apply the discount to the total fare and display the updated amount.		

For Trip management use case (6)

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
TC-01	View trip details and add a note	1. Log in to the system.	Trip ID: 12345, Note: "Bring	Trip details are displayed, and the		

	successfully	2. Navigate to "My Trips" section. 3. Select a trip to view details. 4. Add a note to the trip and save changes.	extra documents."	note is added successfully to the trip.		
TC-02	Attempt to cancel a trip and verify status update	1. Log in to the system. 2. Go to "My Trips" section. 3. Select a trip and click on "Cancel Trip". 4. Confirm the cancellation.	Trip ID: 12345	Trip is successfully canceled, and its status is updated to "Canceled."		
TC-03	Try to change trip details for a non-modifiable trip and check for error message	1. Log in to the system. 2. Navigate to "My Trips". 3. Select a trip that is marked as "Non-modifiable".	Trip ID: 67890 (Non-modifiable)	System prevents modification and displays an error message: "This trip cannot be modified."		

		4. Attempt to edit trip details.				
TC-04	Update trip details during an ongoing trip	1. Log in to the system. 2. Go to "My Trips" section. 3. Select an ongoing trip that is modifiable. 4. Update trip details (e.g., change destination or add additional notes). 5. Save the changes.	Trip ID: 34567, New Destination: "Cairo Museum", Note: "Extend trip by one day"	Trip details are successfully updated, and the system reflects the new destination and note.		
TC-05	Reschedule a trip to a new date and time	1. Log in to the system as a user. 2. Go to "My Trips". 3. Select an eligible trip for rescheduling. 4. Choose a new date and time. 5. Confirm the	Trip ID: 56789, New Date: "2024-12-05", New Time: "10:00 AM"	System updates the trip with the new date and time, and the user sees confirmation of the change.		

		rescheduling.				
TC-06	Notify users of trip rescheduling or cancellation	<ol style="list-style-type: none"> 1. Log in as an admin. 2. Reschedule or cancel a trip. 3. Confirm the update. 4. Check user notifications. 	Trip ID: 67890, Action: "Rescheduled to 2024-12-10"	All users associated with the trip receive a notification about the rescheduling or cancellation.		
TC-07	Check trip details for accuracy and completeness	<ol style="list-style-type: none"> 1. Log in as a user. 2. View details of a specific trip. 3. Verify that all fields (e.g., location, date, time) are complete. 	Trip ID: 78901	All trip details, including location, date, time, and additional notes, are accurately displayed.		
TC-08	Unauthorized user attempts to modify trip details	<ol style="list-style-type: none"> 1. Log in as a regular user. 2. Attempt to access and edit a trip that they didn't create. 3. Check for access 	Trip ID: 45678 (created by another user)	System restricts access and displays a message: "You do not have permission to modify this trip."		

		restriction .				
TC-09	Cancel a trip and process reimbursement if applicable	<ol style="list-style-type: none"> 1. Log in as a user. 2. Go to "My Trips". 3. Select an eligible trip and choose "Cancel". 4. Confirm the cancellation and check for reimbursement. 	Trip ID: 12345	Trip is canceled successfully, and the user receives a reimbursement notification if eligible.		
TC-10	Verify trip is moved to "Completed Trips" after end date	<ol style="list-style-type: none"> 1. Log in as a user. 2. Check "My Trips" after the trip's scheduled end date. 3. Verify trip appears in "Completed Trips". 	Trip ID: 67890, Scheduled End Date: Past date	Trip is successfully moved to "Completed Trips" section.		

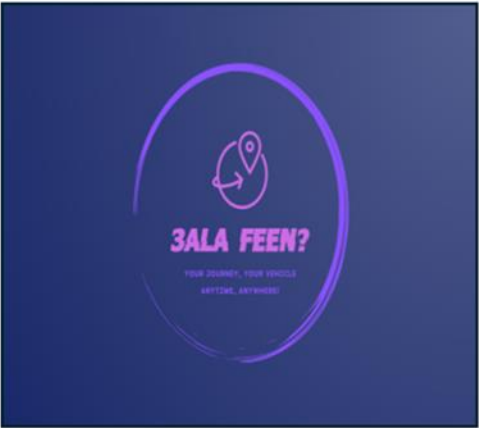
For Scheduling management use case (6)

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
TC-01	Attempt to overbook a trip and check for system warning	1. Log in as a user and attempt to book a trip that is already at maximum capacity. 2. Confirm the booking.	Trip ID: 56789, Date: 2024-12-01 to 2024-12-10, Capacity: Full	System displays an alert or warning indicating that the trip is overbooked and prevents further bookings.		
TC-02	Attempt to create overlapping trip schedules and check for alert	1. Log in as admin. 2. Go to "Scheduling Management". 3. Schedule a new trip that overlaps with an existing trip.	Trip ID: 78901, New Schedule: "Start Date: 2024-12-05, End Date: 2024-12-07" Existing Trip Schedule: "2024-12-06 to 2024-12-08"	System displays a conflict alert, preventing the overlap.		
TC-03	Notify users of a schedule change	1. Log in as admin. 2. Update the schedule for an	Trip ID: 56789, New Start Date: "2024-12-15"	System sends notifications to all users booked on		

		<p>upcoming trip.</p> <p>3. Confirm the update.</p> <p>4. Check notifications for users.</p>		the trip about the schedule change.		
TC-04	Allow multiple users to book a single trip without exceeding capacity	<p>1. Log in as multiple users.</p> <p>2. Each user books the same trip until reaching max capacity.</p> <p>3. Verify booking success for each user.</p>	Trip ID: 56789, User ID: user123	System reflects the updated availability after cancellation, freeing up a spot for new bookings.		
TC-05	View trip schedule summary	<p>1. Log in as User.</p> <p>2. Navigate to "Scheduled trips".</p> <p>3. Select a trip and view schedule summary details.</p>	Trip ID: 56789	System displays a summary of the trip schedule, including dates, times, and booked users.		
TC-06	Modify trip schedule to overlap with an	<p>1. Log in as user.</p> <p>2. Go to "Scheduling</p>	Trip ID: 34567, New Date: Overlapping with	System blocks the update, displaying		

	existing trip	Management". 3. Modify a trip's schedule to overlap with another existing trip. 4. Save changes.	Trip ID: 45678	a conflict alert.		
TC-07	Verify trip status updates after rescheduling	1. Log in as user 2. Reschedule a trip. 3. Check if the trip's status updates accordingly.	Trip ID: 67890, Action: Reschedule to 2024-12-10	The trip status is updated to "Rescheduled", and users are notified.		
TC-08	Send notification for last-minute changes to trip schedule	1. Log in as admin. 2. Change a trip's schedule less than 24 hours before the start. 3. Check user notifications.	Trip ID: 78901, New Date: 1 day before start	System sends immediate notifications to all users associated with the trip		

GUI



from

to

Date

Round trip ☐



rent a car



Book a full trip



Book a ride

Contact us

3alafeen@3alafeen.com



Add a car



 carid


 carname


 car color


 car type


add


Add a trip



 from

 to

 date

 point

add

Add a driver



 from

 id

 date of birth

 carid

add

Other services ▾

Remove driver

Remove car

edit trip

edit car details

run systeam

stop systeam

more services ▾

get number of vehicls

number of vehicls
avilabl

generate report

name

phone number

email

otp sent to your phone

confirm



put car id to remove

remove



put driver id to remove

remove



☐ credit ☐ cash

enter credit number

enddate put discount code apply

cvc  your payment is secure

pay leave