A logo on a white background

Description automatically generatedA red and blue logo

Description automatically generated

UniRoutes

A diagram of a company

Description automatically generated

ERD:

A diagram of a system

Description automatically generated

A diagram of a software company

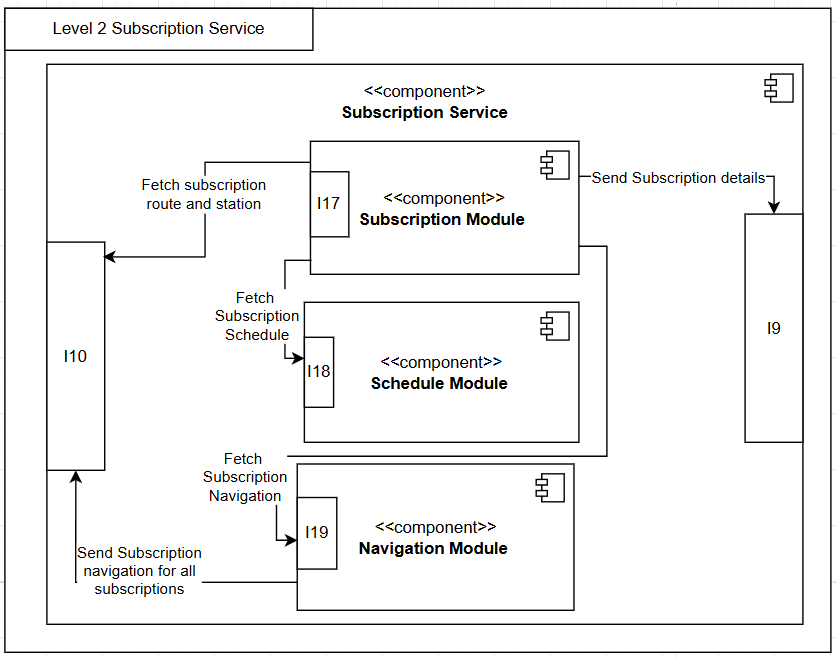
Description automatically generated

A diagram of a payment service

Description automatically generated

A screenshot of a computer

Description automatically generated



A diagram of a service

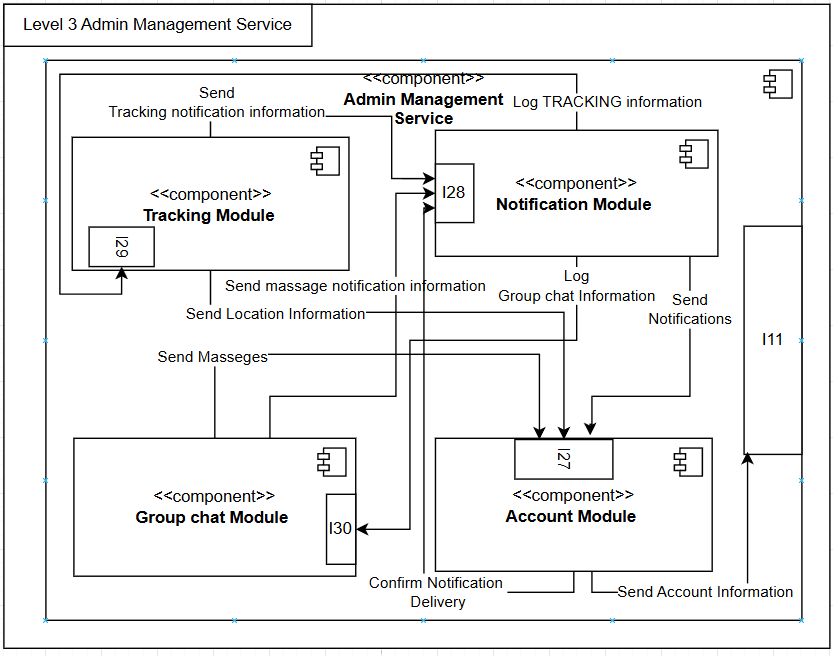
Description automatically generated

A diagram of a bus management service

Description automatically generated

A diagram of a service

Description automatically generated



**Level 0 interface description**

**I1 (Payment Interface):**

* Manages payment processing and integration with the external banking system. It ensures payment confirmation and facilitates payment submissions for subscriptions and other services

**I2 (Student Interaction Interface):**

* Serves as the central hub for Students to access core functionalities such as subscription management, navigation management, group chat, tracking, account, schedule, and notification services.

**I3 (Admin Interaction Interface):**

* Allows administrative staff to manage core functionalities related to the transportation system. This includes notification management, navigation management, group chat oversight, account, tracking, and schedule management.

**I4 (University Admin Interface):**

* Provides university administrators with tools to manage staff, bus company partnerships, and subscriptions.

**I5 (Transportation Admin Interface):**

* Designed for transportation administrators to handle route and station management, as well as bus operations.

**Level 1 interface Description**

**I6 (Payment Service Interface):**

* Connects the Payment Service with other system components, facilitating user transactions for subscriptions, fines, and payment confirmations.

**I7 (Admin Operational Management Interface):**

* Provides administrative tools to manage operational aspects of subscriptions and transportation. This interface ensures admins can perform updates and monitor activities effectively. While also acting as the central interface for admins to manage their accounts also including notification management, navigation management, group chat oversight, account, tracking, and schedule management

**I8 (Accounts Service Interface):**

* Acts as the central interface for managing user accounts and authentication. It bridges other services like subscriptions and payments with user account information. Also enabling them to handle their group chats tracking and Notifications.

**I9 (Subscription Service Interface):**

* Focused on subscription-related actions such as browsing, purchasing, canceling, or updating subscriptions. This interface ensures seamless user access to subscription services.

**I10 (Admin Subscription Service Interface):**

* Provides administrators with full control and monitoring capabilities over all active subscriptions. This interface allows admins to manage, update, and oversee subscription details effectively.

A diagram of a company

Description automatically generated

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Payment M.** | **Subscription M.** | **Buscompany M.** | **Staff M.** | **Schedule M.** | **Bus Mang.** | **Route/Station M.** | **Navigation M.** | **Tracking Mang** | **Account Mang** | **Group Chat M.** | **Notification M** |
| **Admin** |  |  |  |  | 🗸 |  |  | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 |
| **Uni Admin** |  | 🗸 | 🗸 | 🗸 |  |  |  | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 |
| **Transport**  **Admin** |  |  |  |  |  | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 |
| **User** | 🗸 | 🗸 |  |  | 🗸 |  |  | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 |
| **Payment** | 🗸 | 🗸 |  |  |  |  |  |  |  |  |  | 🗸 |
| **Account** |  | 🗸 |  |  | 🗸 |  |  |  | 🗸 |  | 🗸 | 🗸 |
| **Tracking** |  | 🗸 |  |  |  | 🗸 |  | 🗸 | 🗸 |  |  | 🗸 |
| **Notification** | 🗸 | 🗸 |  |  | 🗸 | 🗸 |  | 🗸 | 🗸 |  | 🗸 | 🗸 |
| **Group Chat** |  | 🗸 |  |  | 🗸 |  |  | 🗸 | 🗸 | 🗸 | 🗸 | 🗸 |
| **Navigation** |  | 🗸 |  |  |  |  |  | 🗸 | 🗸 |  | 🗸 | 🗸 |
| **Schedule** |  | 🗸 |  |  | 🗸 |  |  | 🗸 |  |  | 🗸 | 🗸 |
| **Subscription** |  | 🗸 |  |  | 🗸 |  | 🗸 | 🗸 |  |  | 🗸 | 🗸 |
| **Operations** |  | 🗸 |  |  | 🗸 | 🗸 |  | 🗸 | 🗸 |  | 🗸 | 🗸 |

This is the Mapping Module Matrix

Columns are the use cases and the rows are the main classes

Belal Fathy

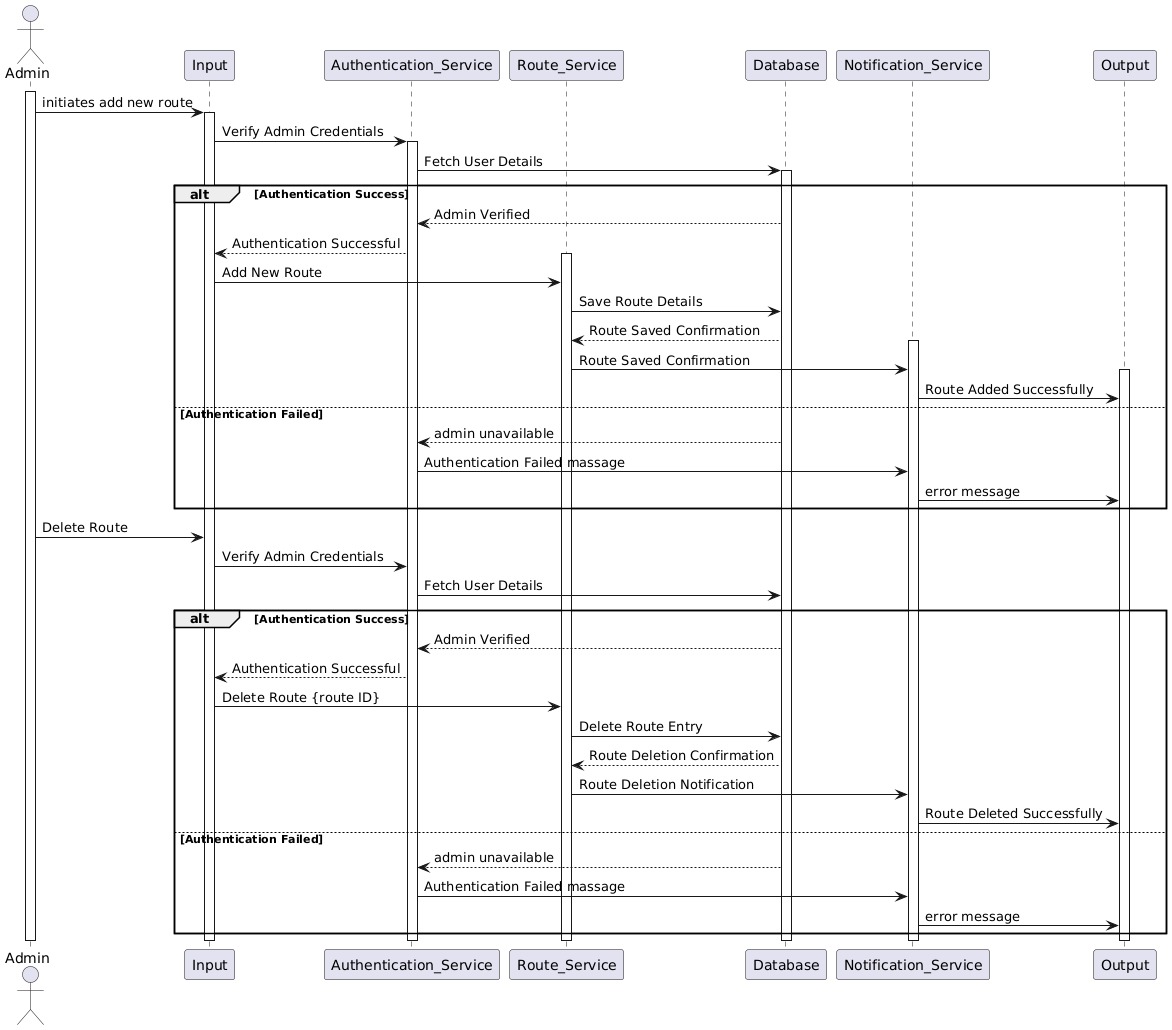
Use case: Subscription Management

A screenshot of a computer program

Description automatically generated22101311

Mahmoud Eid

Use case: Route Management

22100680

Ahmed Islam Farouk 22101008

Use Case : Payment Management

| **Use Case** | **Controller** | **Payment Controller** | **Domain** | **Data Table** | **Display Screens** | **Other UC** |
| --- | --- | --- | --- | --- | --- | --- |
| Payment | UserInput | PaymentGateway | User | Transaction Data | User Input Screen | Retry Payment/Alternative Options |
|  | PaymentOptions | PaymentService |  |  | Payment Options Screen |  |
|  |  | NotificationService |  |  | Notification Screen |  |
|  |  |  |  |  | User Output Screen |  |

| **Step** | **Message Name** | **Owner Class Name** |
| --- | --- | --- |
| 1 | InitiatePayment() | UserInput |
| 2 | SelectPaymentMethod() | PaymentOptions |
| 3 | SendPaymentDetails() | PaymentGateway |
| 4 | ProcessPayment() | PaymentService |
| 5 | SaveTransaction() | Database |
| 6 | NotifyUserOfPaymentSuccess() | NotificationService |
| 7 | NotifyUserOfPaymentFailure() | NotificationService |
| 8 | DisplayNotification() | UserOutput |
| 9 | RetryPaymentOrAlternativeOption() | PaymentOptions |

A screenshot of a computer screen

Description automatically generated

Abdallah Basem Zain 22100848 Use case: Navigation Management

|  |  |  |
| --- | --- | --- |
| **Domain** | **Database Table** | **Display Screens or Report** |
| User | Accounts table | Customer App Screen |
|  | Subscriptions table | Registration Screen |
|  | Routes table | Map Screen |

The system validates the user account and retrieves the list of subscriptions.

The system fetches and displays the relevant routes based on the user subscriptions.

The user can one see his location on the map If user is not logging on.

The user can review a map showing their subscribed routes.

The user can view detailed information about that route by clicking on a specific route.

Messages:

Get account cardinalities ()

Return cardinalities ()

get subscription details ()

return details ()

Get routes ()

Return routes ()

Return null ()

Ask location ()

Assign location ()

A screenshot of a computer screen

Description automatically generated

Eyad Metwally 22100757

A screenshot of a computer screen

Description automatically generatedUse Case: Notification Management

Mazen Ahmed Samir 22100369

A screenshot of a computer

Description automatically generatedUse Case: Bus Management

Belal Fathy 22101311

Use Case: Subscription Management

A diagram of a diagram

Description automatically generatedLevel 0

Level 1

A diagram of a software flow

Description automatically generated with medium confidence

Mahmoud Eid 22100680

A diagram of a account management system

Description automatically generatedUse Case: Account Management

A diagram of a user account

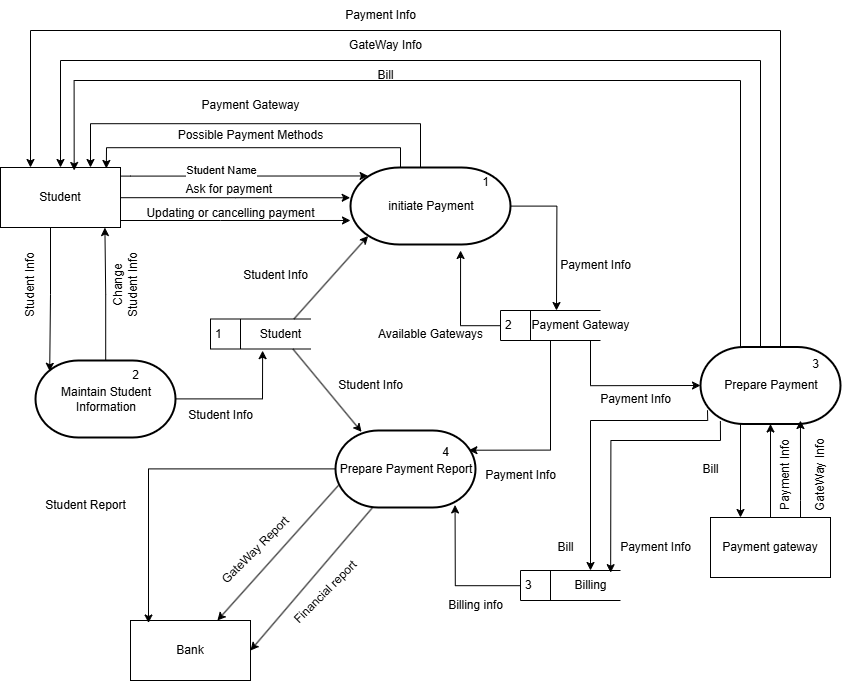
Description automatically generated

Ahmed Islam Farouk 22101008

Use Case: Payment Management

A diagram of a diagram

Description automatically generatedLevel 0

Level 1

Abdallah Basem 22100848

Use Case: Navigation Management

A diagram of a user login

Description automatically generatedLevel 0

A diagram of a diagram

Description automatically generatedLevel 1

Eyad Metwally 22100757

A diagram of a diagram

Description automatically generatedUse case: Notification

Mazen Ahmed Samir 22100369

A diagram of a diagram

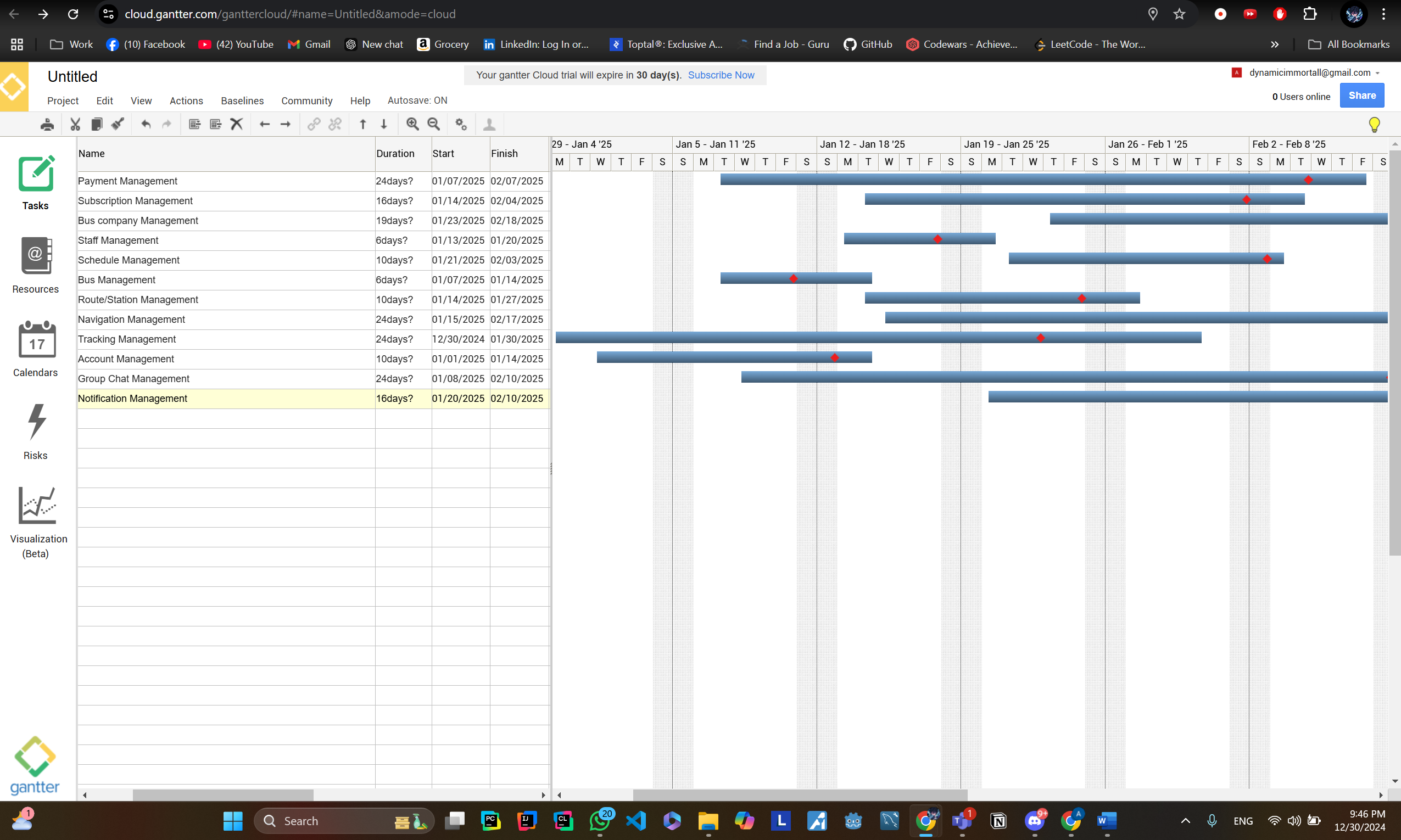
Description automatically generatedUse Case: Staff Management

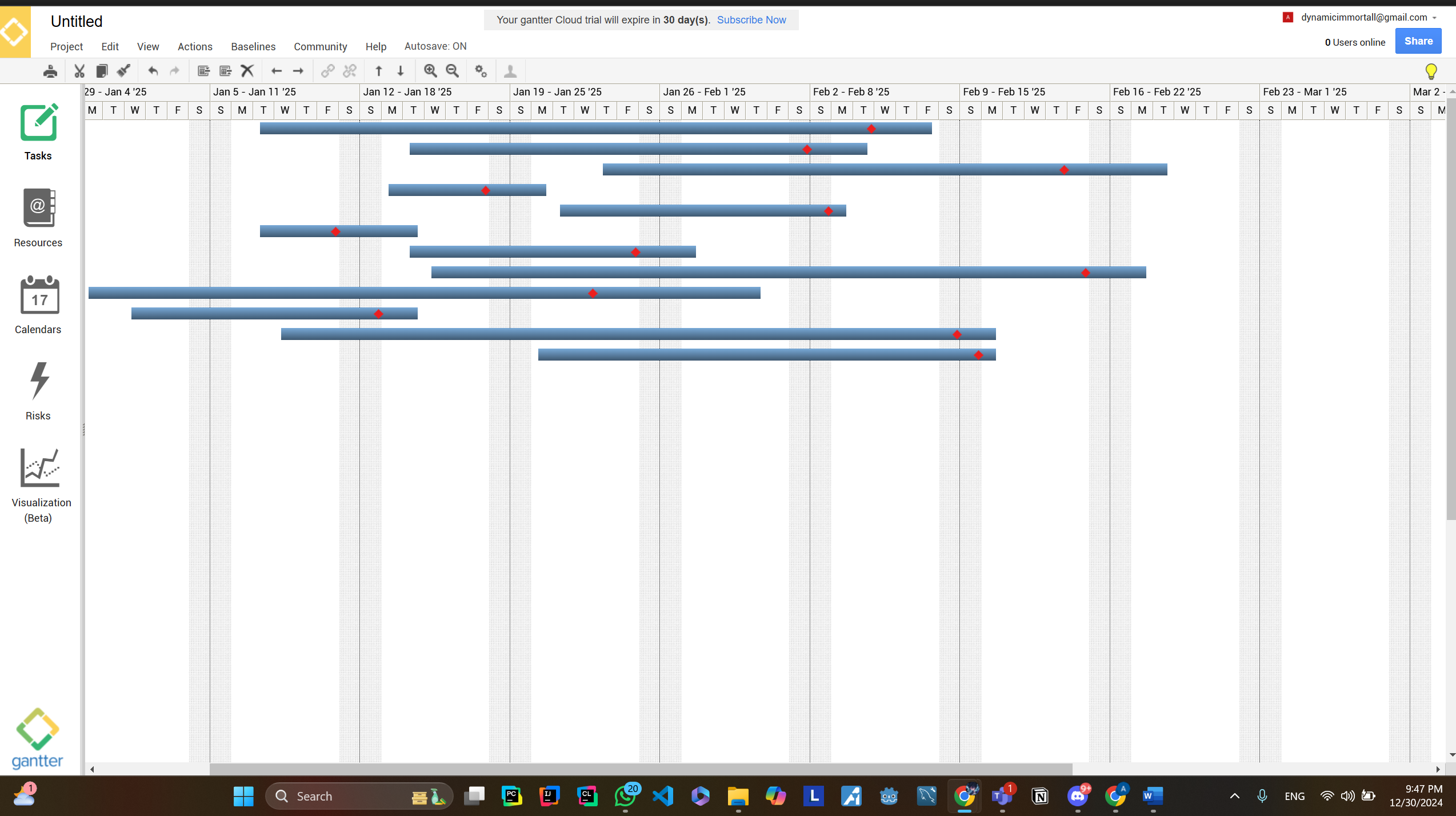
A diagram of a company

Description automatically generated

|  |  |
| --- | --- |
| **Use Case** | **Estimation(Fibonacci)** |
| **Payment Management** | 12 |
| **Subscription Management** | 8 |
| **Bus company Management** | 3 |
| **Staff Management** | 3 |
| **Schedule Management** | 5 |
| **Bus Management** | 3 |
| **Route/Station Management** | 5 |
| **Navigation Management** | 12 |
| **Tracking Management** | 12 |
| **Account Management** | 5 |
| **Group Chat Management** | 12 |
| **Notification Management** | 8 |

Use Case Estimation Each point is 2 days

Gantt chart



UniRoutes Test Plan

This test plan covers:

1. Unit Testing: Testing individual classes and methods.
2. Integration Testing: Testing interactions between subsystems.
3. System Testing: Testing the complete system functionality.
4. Acceptance Testing: Verifying that business requirements are met.

Modules to Test:

1. Bus Company Management

Add, update, and delete bus companies.

1. Driver and Staff Management

Add, remove, and update driver and supervisor profiles.

Assign staff to routes and chat groups.

1. Bus Route and Stops Management

Manage routes, stops, and schedules.

1. Subscription Management

Reserve and manage bookings.

1. Notifications and Alerts

Manage and send notifications for delays, cancellations, and emergencies.

1. Real-Time Tracking

Display real-time bus locations.

1. Emergency and Safety Features

Alert dispatchers with emergency notifications.

1. Feedback System

Collect feedback from users post-travel.

1. Payment Management

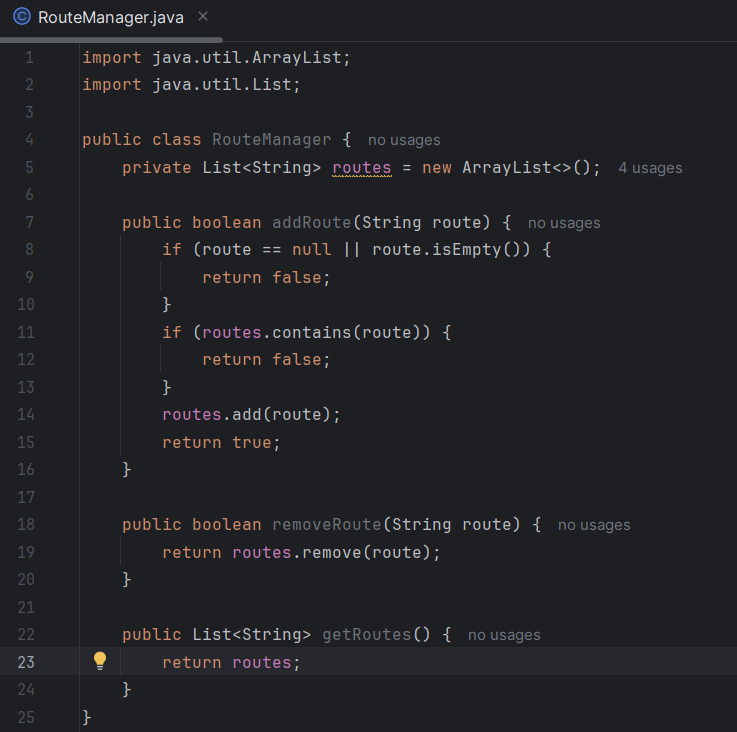
Process payments and handle refunds.

Testing Strategies

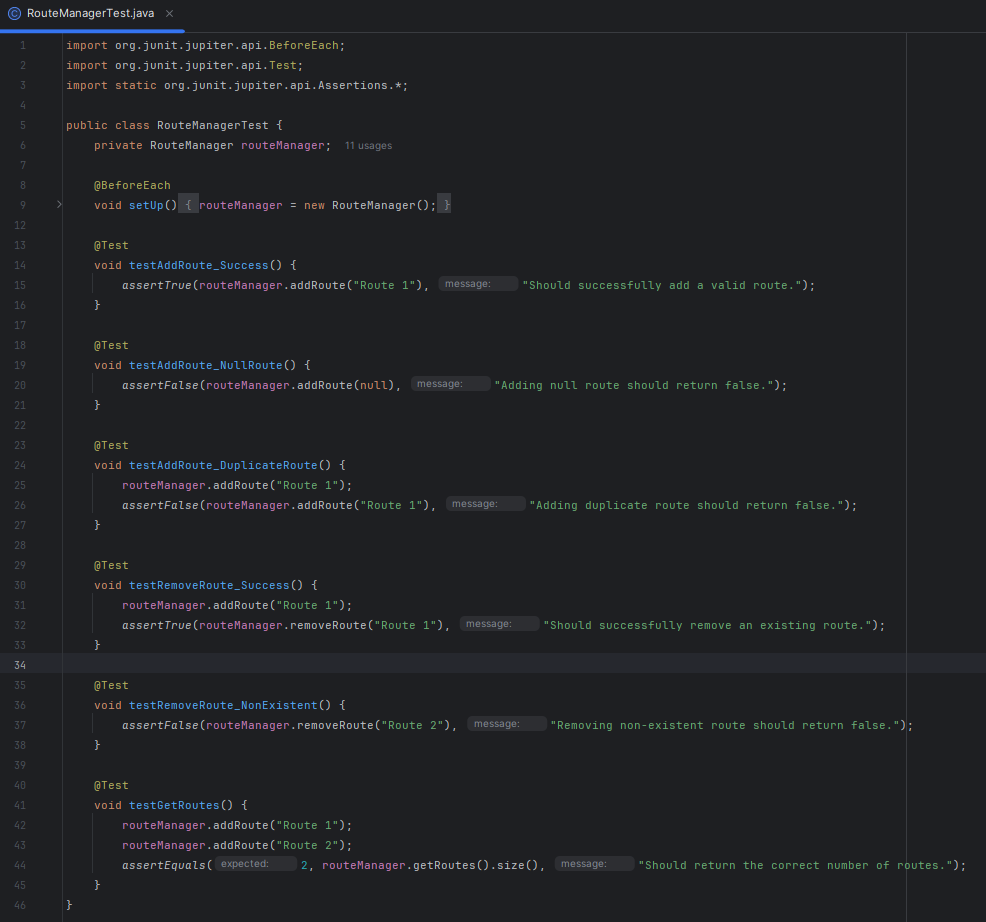
1. Unit Testing

Tools: JUnit, TestNG.  
Examples:

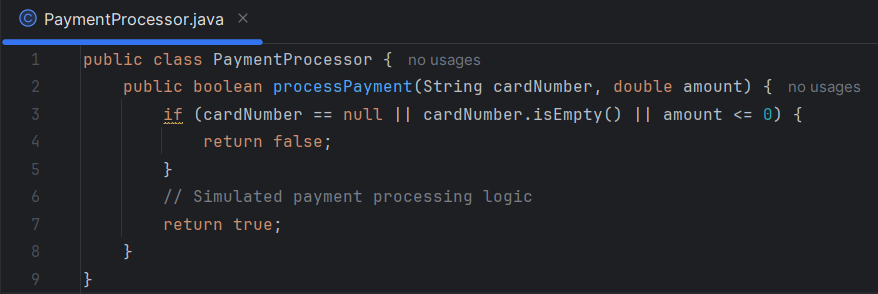
* Class: RouteManager
  + Method: addRoute()
    - Test Case 1: Provide valid route details; expect success.
    - Test Case 2: Provide a null object; expect failure.
  + Method: removeRoute()
    - Test Case 1: Remove an existing route; expect success.
    - Test Case 2: Attempt to remove a non-existent route; expect failure.
  + Code:
    - RouteManger:



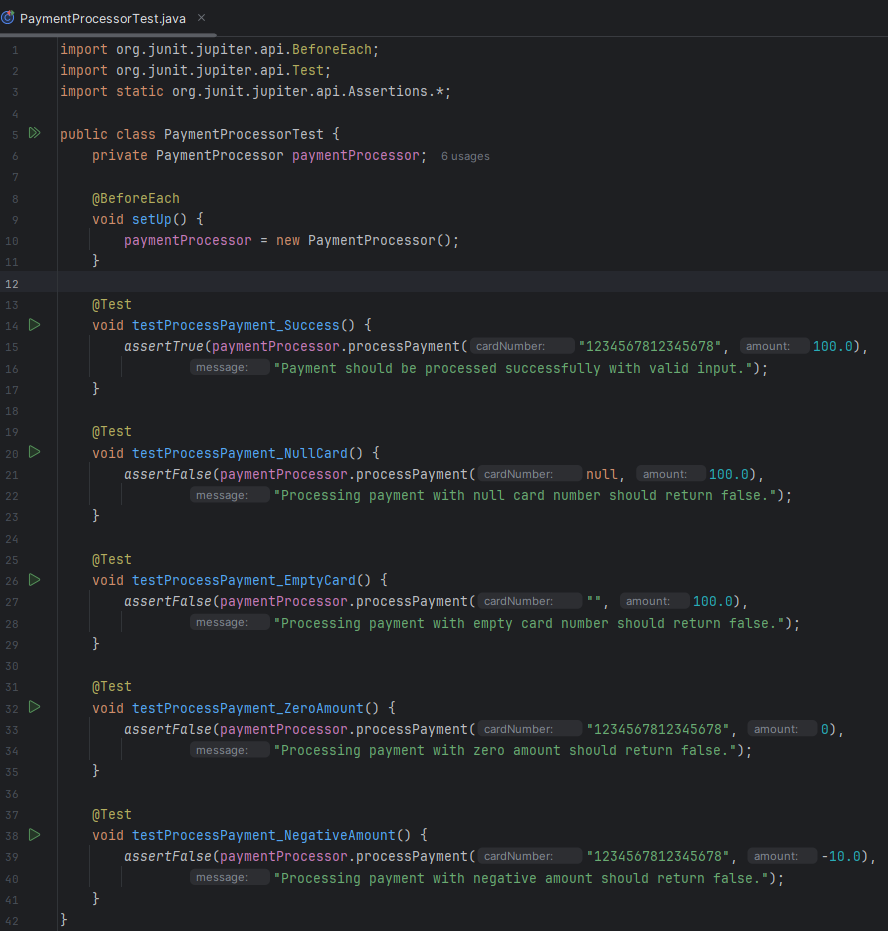
* + - RouteManagerTest:



* Class: PaymentProcessor
  + Method: processPayment()
    - Test Case 1: Process payment with valid details; expect success.
    - Test Case 2: Insufficient funds; expect failure and error message.
  + Code:
    - PaymentProcessor:



* + - PaymentProcessorTest:



2. Integration Testing  
Tools: Mockito, JUnit.

* Subsystems:
  1. GPS System and Notification System
  2. Booking System and Payment Gateway

3. System Testing

1. Scenario: Student booking workflow
2. Scenario: Emergency Alert

4. Acceptance Testing

Objective: Ensure the system meets business requirements.  
Actors: Students, admins, and supervisors.

To verify that the system meets business requirements, we will conduct thorough acceptance testing based on predefined user stories and requirements. This involves simulating real-world scenarios, such as route management by admins and feedback submission by students, and validating the results against the expected outcomes. Key business processes like creating and updating routes, as well as collecting and analyzing user feedback, will be tested to ensure they align with organizational goals and provide value to end-users. Continuous feedback from stakeholders will guide iterative improvements and ensure compliance with business needs.

**UniRoutes Test Plan**

**1. Bus Company Management**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U1 | AddCompany | Name: "XYZ Buses" | Company "XYZ Buses" added successfully. |
|  |  | Address: "123 Main St" |  |
| U2 | UpdateCompany | CompanyID: 101 | Details updated for CompanyID 101. |
|  |  | New Name: "ABC Transport" |  |
| U3 | DeleteCompany | CompanyID: 102 | CompanyID 102 removed successfully. |
| U4 | GetCompanies | None | List of all bus companies displayed. |

**2. Driver and Staff Management**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U5 | AddDriver | Name: "John Doe" | Driver "John Doe" added successfully. |
|  |  | LicenseID: "D12345" |  |
| U6 | UpdateDriver | DriverID: 201 | DriverID 201 details updated successfully. |
|  |  | New Phone: "9876543210" |  |
| U7 | DeleteDriver | DriverID: 202 | DriverID 202 removed successfully. |
| U8 | GetDrivers | None | List of all drivers displayed. |
| U9 | AddSupervisor | Name: "Alice Smith" | Supervisor "Alice Smith" added successfully. |
|  |  | Contact: "1234567890" |  |
| U10 | UpdateSupervisor | SupervisorID: 301 | SupervisorID 301 details updated successfully. |
|  |  | New Email: "[alice@example.com](mailto:alice@example.com)" |  |
| U11 | DeleteSupervisor | SupervisorID: 302 | SupervisorID 302 removed successfully. |
| U12 | GetSupervisors | None | List of all supervisors displayed. |

**3. Bus Route and Stops Management**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U13 | AddRoute | Name: "Route A" | Route "Route A" added successfully. |
|  |  | Stops: ["Station 1", "Station 2"] |  |
| U14 | UpdateRoute | RouteID: 401 | RouteID 401 updated with new stops. |
|  |  | New Stops: ["Station 3"] |  |
| U15 | DeleteRoute | RouteID: 402 | RouteID 402 removed successfully. |
| U16 | GetRoutes | None | List of all routes displayed. |
| U17 | AddStop | RouteID: 401, Stop: "Station 4" | Stop "Station 4" added to RouteID 401. |
| U18 | UpdateStop | StopID: 501 | StopID 501 details updated. |
|  |  | New Name: "Main Plaza" |  |
| U19 | DeleteStop | StopID: 502 | StopID 502 removed successfully. |
| U20 | GetStops | RouteID: 401 | List of stops for RouteID 401 displayed. |

**4. User Roles and Permissions**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U21 | AssignRole | UserID: 601, Role: "Supervisor" | Role "Supervisor" assigned to UserID 601. |
| U22 | UpdateRole | UserID: 602, Role: "Admin" | Role updated to "Admin" for UserID 602. |
| U23 | RemoveRole | UserID: 603 | Role removed for UserID 603. |
| U24 | GetRoles | None | List of user roles displayed. |

**5. Advanced Subscription Management**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U25 | ReserveSeat | UserID: 701, RouteID: 401, Seat: 10 | Seat 10 on RouteID 401 reserved for UserID 701. |
| U26 | CancelReservation | ReservationID: 801 | ReservationID 801 cancelled successfully. |
| U27 | GetReservations | UserID: 701 | List of reservations for UserID 701 displayed. |
| U28 | SendReminder | ReservationID: 801 | Reminder sent for ReservationID 801. |

**6. Notifications and Alerts**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U29 | SendGlobalAlert | Message: "System maintenance" | Alert sent to all users successfully. |
| U30 | SendRouteAlert | RouteID: 401, Message: "Delay" | Alert sent to users on RouteID 401. |
| U31 | SendBusAlert | BusID: 301, Message: "Breakdown" | Alert sent to users of BusID 301. |
| U32 | GetAlerts | UserID: 701 | List of alerts received by UserID 701. |

**7. Real-Time Bus Tracking**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U33 | GetBusLocation | BusID: 301 | Real-time location of BusID 301 displayed. |
| U34 | GetRouteETAs | RouteID: 401 | ETAs for RouteID 401 stops displayed. |

**8. Emergency and Safety Features**

**Unit Tests**

| **Unit Test ID** | **Method Name** | **Input Example** | **Expected Output** |
| --- | --- | --- | --- |
| U35 | SendEmergencyAlert | UserID: 701, Location: "Station 1" | Emergency alert sent successfully. |
| U36 | DeployAssistance | AlertID: 901 | Assistance deployed to location in AlertID 901. |

Execution Plan

**1. Unit Tests**

* + Unit Test U1
  + Unit Test U2
  + Unit Test U3
  + Unit Test U4
  + Unit Test U5
  + Unit Test U6
  + Unit Test U7
  + Unit Test U8
  + Unit Test U9
  + Unit Test U10
  + Unit Test U11
  + Unit Test U12
  + Unit Test U13
  + Unit Test U14
  + Unit Test U15
  + Unit Test U16
  + Unit Test U17
  + Unit Test U18
  + Unit Test U19
  + Unit Test U20
  + Unit Test U21
  + Unit Test U22
  + Unit Test U23
  + Unit Test U24
  + Unit Test U25
  + Unit Test U26
  + Unit Test U27
  + Unit Test U28
  + Unit Test U29
  + Unit Test U30
  + Unit Test U31
  + Unit Test U32
  + Unit Test U33
  + Unit Test U34
  + Unit Test U35
  + Unit Test U36

**2. Integration Testing**

* + Integration Testing for U1, U6
  + Integration Testing for U2, U3, U6, U1
  + Integration Testing for U5, U6, U2, U1
  + Integration Testing for U4, U6, U1

**3. Regression Testing**

* + Regression Testing for U6
  + Regression Testing for U1
  + Regression Testing for U2
  + Regression Testing for U3
  + Regression Testing for U4
  + Regression Testing for U5
  + Regression Testing for U1, U6
  + Regression Testing for U6, U2, U3, U1
  + Regression Testing for U6, U5, U2, U1
  + Regression Testing for U6, U4, U1

**4. Resources**

* + Testers: 3 testers.
  + Tools: Junit, Selenium.