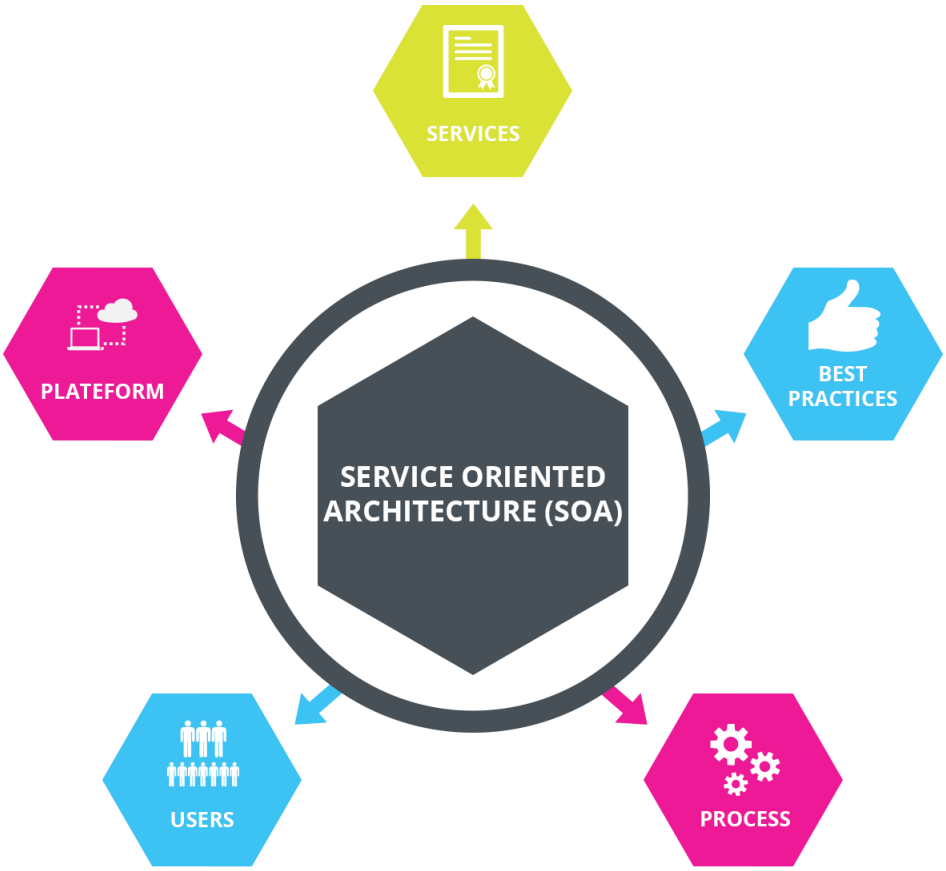
SERVICE-ORIENTED ARCHITECTURE (SOA)

SERVICE MODELLING



ABSTRACT

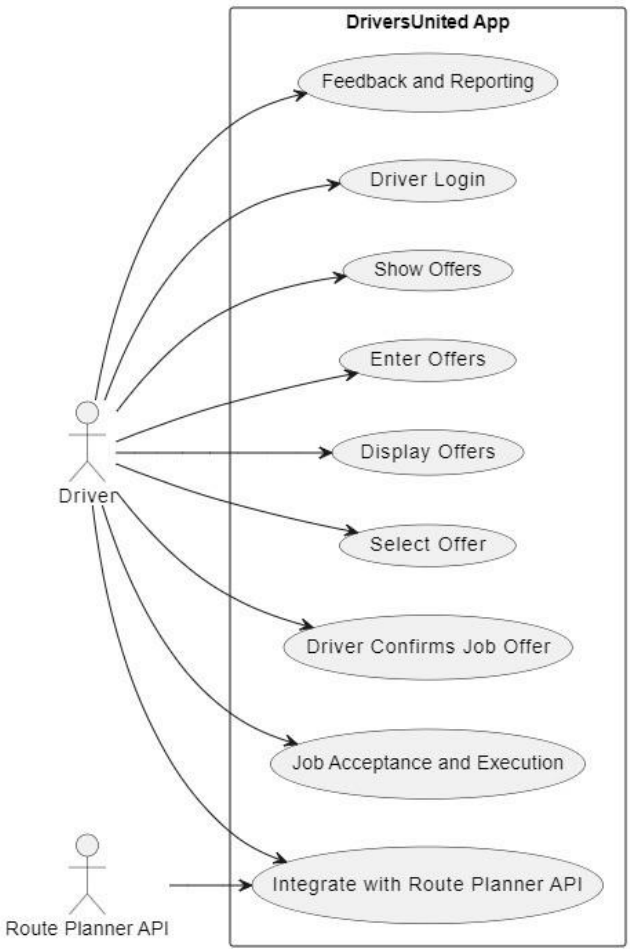
DriversUnited is a comprehensive system designed to empower delivery drivers associated with various delivery platforms such as UberEats, Deliveroo, and DoorDash. The primary objective is to enhance drivers' ability to compare job offers across platforms, thereby increasing their bargaining power and fostering a more competitive market landscape.
The DriversUnited service comprises a user-friendly app interfacing with a robust backend system. This system orchestrates seamless interactions between drivers, and external route planning APIs to optimize job selection and execution.

To realize these functionalities, we propose a comprehensive design comprising:

- **Use Case Diagram**: Illustrating the primary actors, their interactions, and the system's functionalities.
- **Sequence Diagrams**: Detailing the chronological flow of interactions between system components and external entities.
- **Component Diagram**: Visualizing the modular components and their interconnections within the DriversUnited system.
- **Interfaces with Detailed Operation Signatures**: Clearly defining the operations and parameters for seamless communication between system components.
- **Class Diagram as Conceptual Data Model**: Representing the essential data structures and relationships within the system.
- **Textual Descriptions of Preconditions and Effects:** Outlines the conditions that must be met before certain operations can be performed, as well as the outcomes or effects of those operations.

Our solution adheres to principles of well-formedness, completeness, correctness, consistency, readability, and documentation, ensuring a coherent and effective design that meets the specified requirements while allowing for scalability and adaptability to future enhancements.

1. USE CASE DIAGRAM:



Use Case Diagram Description:

Actors:

- **Driver:** Represents the registered driver who interacts with the DriversUnited app.
- **Route Planner API:** An external actor representing the API used for route planning to estimate travel time.

System:

- **DriversUnited App:** Represents the application that facilitates interactions between the driver and the delivery service platforms.

Use Cases:

- **Driver Login:** The driver logs into the DriversUnited app.

- **Show Offers:** The driver views job offers from the connected DeliveryService platforms.
- **Enter Offers:** The driver enters job offers from DeliveryService platforms they find attractive into the DriversUnited app.
- **Integrate with Route Planner API:** The DriversUnited app integrates with the external Route Planner API to fetch job offers and obtain detailed payment information, route, and estimates of distance and time.
- **Display Offers:** Job offers are displayed on the DriversUnited app with basic details.
- **Select Offer:** The driver selects a job offer that looks appealing in the DriversUnited app.
- **Driver Confirms Job Offer:** The driver, making an informed decision, confirms the job offer within the DriversUnited app.
- **Job Acceptance and Execution:** Once the job is confirmed, the driver uses the connected delivery service app to formally accept the job and start the delivery process.
- **Feedback and Reporting:** After completing the job, the driver can provide feedback or report any discrepancies via the DriversUnited app.
- **Integrate with Route Planner API:** The external Route Planner API interacts with the DriversUnited app to provide route planning information.

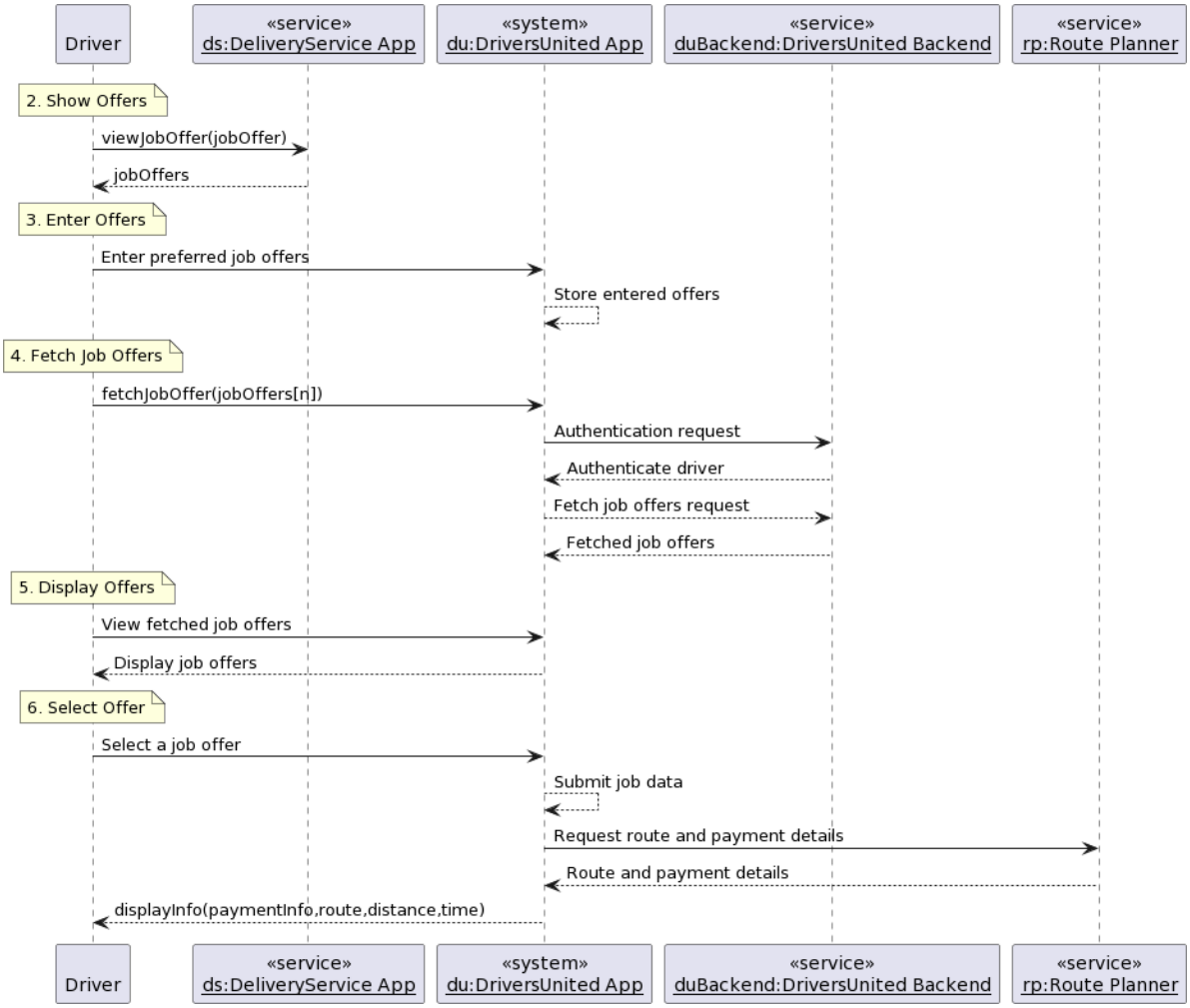
Relationships:

- **Driver to DriversUnited App:** The driver interacts with the DriversUnited app to perform various actions.
- **RoutePlannerAPI to Integrate with Route Planner API:** The external Route Planner API interacts with the DriversUnited app to provide route planning information.

This use case diagram outlines the interactions between the actors and the use cases within the DriversUnited app, emphasizing the integration with the Route Planner API for fetching job offers and route planning.

Use Case Diagram Bot Conversation Link:
https://chat.openai.com/share/24aaa270-cf6c-4226-befb-b87a7e168524

2. SEQUENCE DIAGRAM:



1. Show Offers:

- The process begins with the driver using the DeliveryService App to view job offers.
- The driver initiates the action by calling the `viewJobOffer(jobOffer)` method.
- The DeliveryService App responds by providing the available job offers to the driver (`jobOffers`).

2. Enter Offers:

- The driver then enters preferred job offers into the DriversUnited App.
- The DriversUnited App stores the entered offers for future reference.

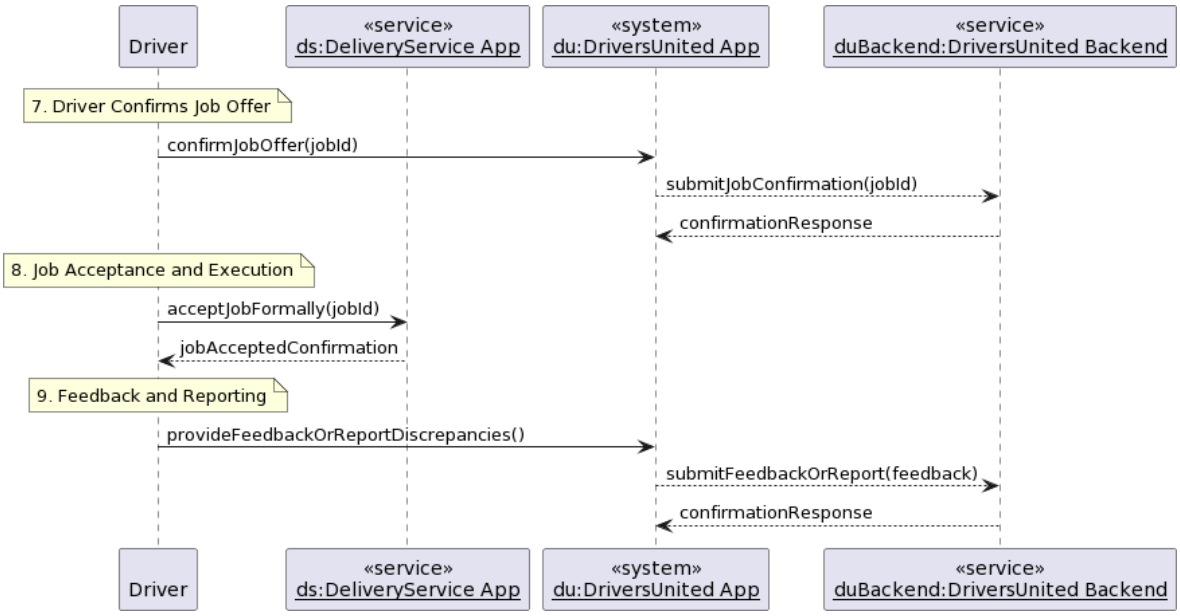
3. Fetch Job Offers:

- The driver, through the DriversUnited App, initiates the process of fetching job offers by calling `fetchJobOffer(jobOffers[n])`.
- The DriversUnited App, in turn, sends an authentication request to the DriversUnited Backend.
- The DriversUnited Backend authenticates the driver and responds with the fetched job offers.

- 4. Display Offers:
- The DriversUnited App displays the fetched job offers to the driver.

5. Select Offer:

- The driver selects a specific job offer using the DriversUnited App.
- The DriversUnited App submits the job data, triggering a request for route and payment details.
- The request is sent to the Route Planner.
- The Route Planner responds with detailed information, including payment details, route information, distance, and time required.
- The DriversUnited App then displays this information to the driver using `displayInfo(paymentInfo, route, distance, time)`.



7. Driver Confirms Job Offer:

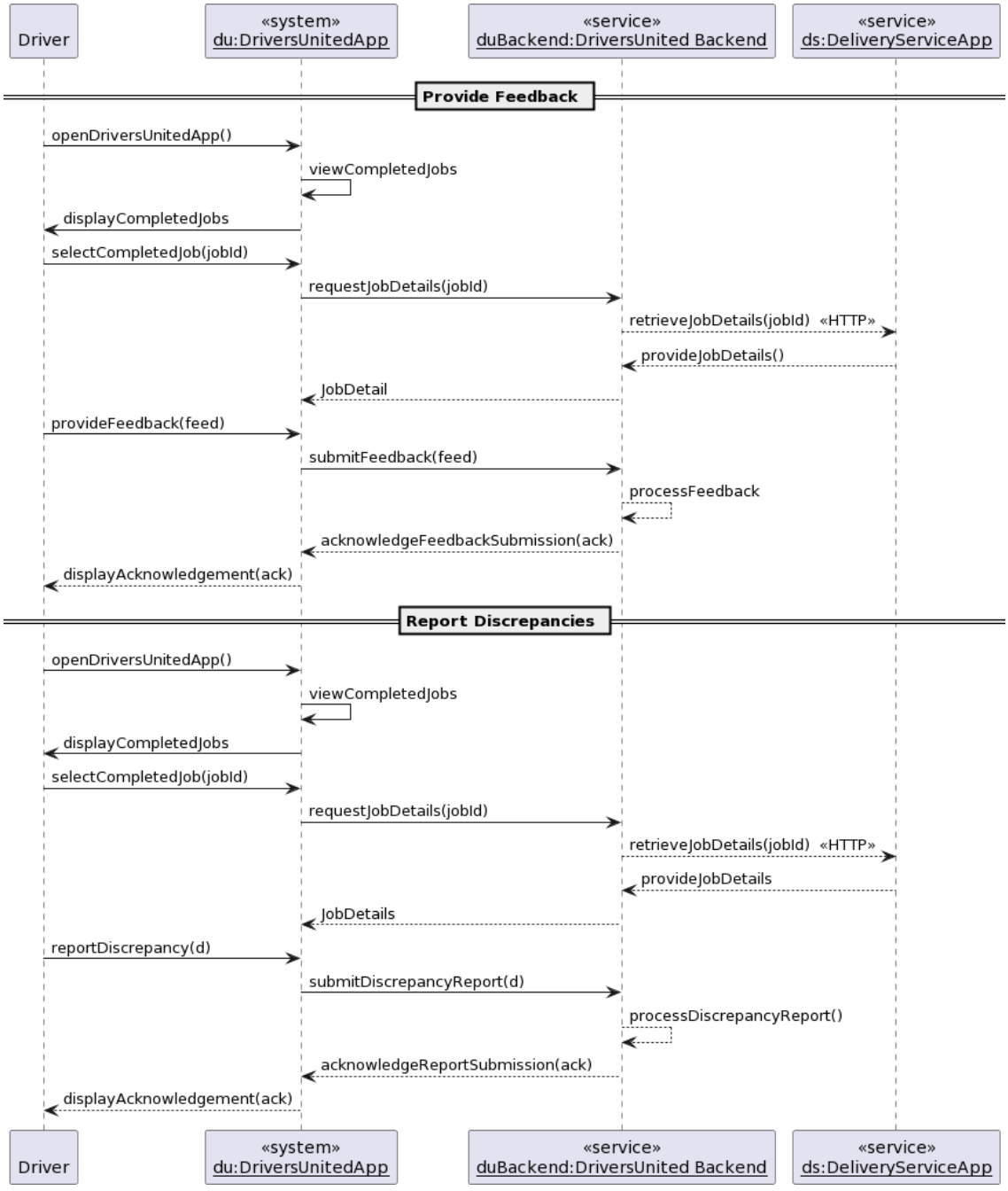
- The driver, having reviewed the job offer, confirms it using the DriversUnited App by calling the `confirmJobOffer(jobId)` method.
- The DriversUnited App, upon receiving the confirmation, submits a job confirmation request (`submitJobConfirmation(jobId)`) to the DriversUnited Backend.
- The DriversUnited Backend processes the confirmation and responds with a confirmation message (`confirmationResponse`).

8. Job Acceptance and Execution:

- After receiving the confirmation from the DriversUnited Backend, the driver formally accepts the job and initiates the execution process.
- The driver uses the DeliveryService App to call the `acceptJobFormally(jobId)` method, indicating the formal acceptance of the job.
- The DeliveryService App responds with a confirmation message (`jobAcceptedConfirmation`) to the driver.

9. Feedback and Reporting:

- After completing the job, the driver has the option to provide feedback or report any discrepancies.
- The driver uses the DriversUnited App to call the `provideFeedbackOrReportDiscrepancies()` method.
- The DriversUnited App submits the feedback or report to the DriversUnited Backend using the `submitFeedbackOrReport(feedback)` method.
- The DriversUnited Backend processes the feedback or report and responds with a confirmation message (`confirmationResponse`).



Provide Feedback Scenario:

1. Driver Views Completed Jobs:

- The driver opens the DriversUnited App.
- The app displays the option to view completed jobs.
- The driver selects the completed jobs option.

2. Select Completed Job:

- The driver chooses a specific completed job from the list.

3. Retrieve Job Details:

- The DriversUnited App sends a request to the BackendService for details of the selected completed job.
- The BackendService interacts with the DeliveryServiceApp to retrieve job details.
- The DeliveryServiceApp provides the job details to the BackendService.

4. Provide Feedback:

- The DriversUnited App allows the driver to provide feedback on the completed job.
- The driver submits feedback through the app.

5. Submit Feedback to BackendService:

- The DriversUnited App sends the feedback to the BackendService using the `Submit Feedback` action.
- The BackendService processes the feedback.

6. Acknowledge Feedback Submission:

- The BackendService sends an acknowledgment to the driver, confirming the submission of feedback.
- The DriversUnited App displays the acknowledgment to the driver.

Report Discrepancies Scenario:

1. Driver Views Completed Jobs:

- Similar to the feedback scenario, the driver opens the DriversUnited App, views completed jobs, and selects a specific job.

2. Retrieve Job Details:

- The DriversUnited App sends a request to the BackendService for details of the selected completed job.
- The BackendService interacts with the DeliveryServiceApp to retrieve job details.
- The DeliveryServiceApp provides the job details to the BackendService.

3. Report Discrepancy:

- The DriversUnited App provides the option to report discrepancies related to the completed job.

- The driver submits a discrepancy report through the app.

4. Submit Report to BackendService:

- The DriversUnited App sends the discrepancy report to the BackendService using the `Submit Discrepancy Report` action.
- The BackendService processes the report.

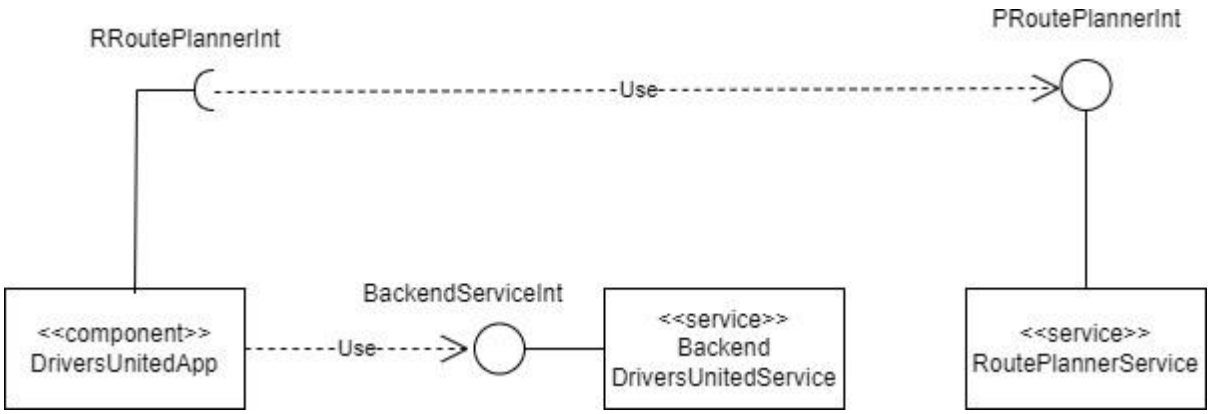
5. Acknowledge Report Submission:

- The BackendService sends an acknowledgment to the driver, confirming the submission of the discrepancy report.
- The DriversUnited App displays the acknowledgment to the driver.

Sequence Diagram Bot Conversation Link:

https://chat.openai.com/share/24aaa270-cf6c-4226-befb-b87a7e168524

3. COMPONENT DIAGRAM:



DriversUnitedApp Component:

- Name: DriversUnitedApp
- Purpose: Provides a UI for drivers to compare job offers and choose the most convenient or profitable job offer.
- Required Interface: DriversUnitedApp requires an interface for getting detailed payment information and estimate of the distance and time required.
- Dependencies: Relies on Backend DriversUnitedService for authentication and fetching similar job offers entered by the driver.

Backend DriversUnited Service:

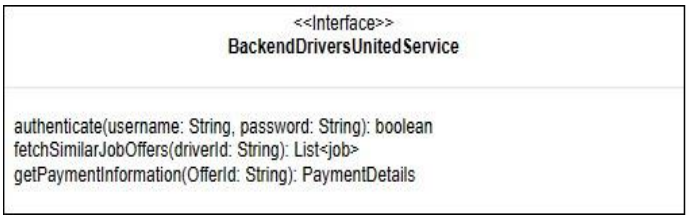
- Name: BackendDriversUnitedService
- Purpose: Authenticates DriversUnitedApp, provides job offers

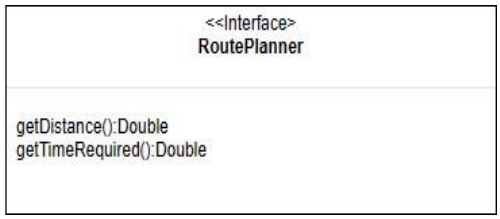
- Provider Interface: BackendDriversUnitedService provides an interface for retrieving jobs according to driver's interest.

Route Planner Service:

- Name: RoutePlannerService
- Purpose: To get information about the job offer selected
- Provider Interface: The RoutePlannerService furnishes comprehensive payment particulars along with route specifics, including estimated distance and time requirements.

4. INTERFACES WITH DETAILED OPERATION SIGNATURES:





The above daigram represents the interfaces with detailed operation signatures:

Backend DrivesUnited Service:

- Verifies the driver's identity with the backend DriversUnited assistance.
- Gathers work offers that are comparable to the ones the driver entered.
- Obtains comprehensive payment details for a particular work offer.

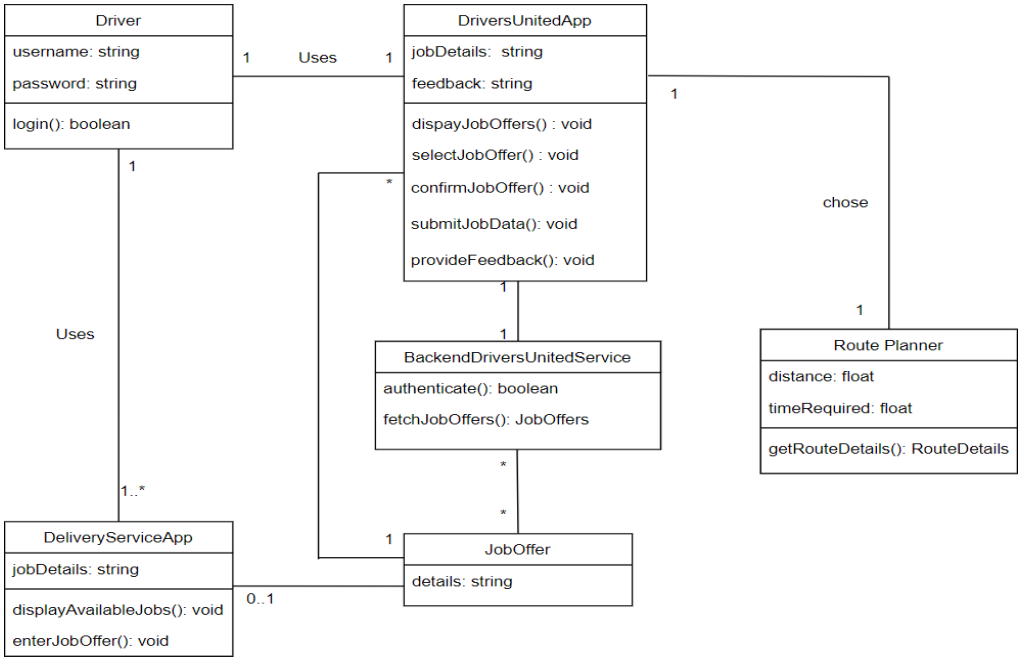
Route Planner:

- Returns a route and an estimate of the time and distance needed for a chosen job offer using an appropriate route planner.

Interface Diagram Bot Conversation Link:

https://chat.openai.com/share/e7387a4a-1afe-42e4-a817-137ec81b7d50

5. CLASS DIAGRAM AS CONCEPTUAL DATA MODEL:



1. Driver:

Attributes:

username: A string which represents the driver's username.

password: A string that serves as the driver's password.

Methods:

login(): A method that enables system login for the driver.

2. Delivery Service App:

Attributes:

jobDetails: A string which holds the detailed information related to a job offer.

Methods:

displayAvailableJobs (): This function displays a delivery service application's current job listing.

enterJobOffer(): Enables the driver to enter employment offers from the delivery service application that they find appealing.

3. Drivers United App:

Attributes:

jobDetails: This attribute holds the detailed information about a specific job offer. It includes essential details such as display, select and confirm job offer.
feedback: This attribute stores the feedback provided by the driver after completing a job.
Methods:
displayJobOffers(): This function shows job openings on the Drivers United app.
selectJobOffer(): Which enables the driver to choose a job offer from the list that is shown.
confirmJobOffer(): Confirms the selected job offer.
submitJobData(): Sends job data to Drivers United's backend service.
provideFeedback(): Allows the driver to provide feedback or report discrepancies.

4. Backend Drivers United Service:
Methods:
authenticate(): Provides the backend service with the Drivers United application's authentication.
fetchJobOffers (): Retrieves the most relevant employment offers that resemble the ones the driver entered.

5. Route Planner:
Attributes:
distance: Represents the distance associated with a route. IT quantifies the length between the starting point and the destination, typically measured in units such as kilometres or miles.
timeRequired: Denotes the time required to traverse a route. It indicates the duration from the starting point to the destination, usually measured in units such as hours or minutes.
Methods:
getPayment Information (): Returns comprehensive payment details for a chosen job offer.
getRoute Details (): Determines and provides the distance and time needed to complete a chosen job offer.

6. Job Offer:
Attributes:
details: A string which represents the specification of the job offers.

Relationships between the components:

- Each driver is assigned with one or more Delivery Service App instances.
- One Drivers United App instance is connected to each Driver.
- Drivers United App instance is associated with one Backend Drivers United Service instance and one Route Planner instance.
- Delivery Service App and Job Offer instances have a one-to-many relationship, meaning that a single delivery service application may have several job offers.
- Likewise, Drivers United App and Job Offer instances have a one-to-many relationship, meaning that the Drivers United application may show more than one job offer.

- Multiple Job Offer instances are connected to the Backend Drivers United Service, suggesting that it supplies job offers to the Drivers United application.

Class Diagram Bot Conversation Link:
https://chat.openai.com/share/a75054e1-0010-49f9-af95-6304483724a0

6. Textual descriptions of the preconditions and effects of operations for the DriversUnited app and its backend service are as follows:

Operations	Precondition	Effects of Operation
Driver login	The driver must have a registered account with both the DeliveryService app and the DriversUnited app. The credentials provided must match the records stored in the respective databases.	Upon successful login, the driver gains access to the features and functionalities of both the DeliveryService and DriversUnited apps.
Show offers	The DeliveryService app must retrieve available job offers from its backend server. The driver's device must have an active internet connection.	The driver can view a list of available job offers provided by the DeliveryService app.
Enter offers	The driver must have access to the list of job offers provided by the DeliveryService app.	The driver selects job offers deemed attractive and enters them into the DriversUnited app for comparison and further processing.
Fetch job offers	The DriversUnited app must have received job offers entered by the driver. The app must be authenticated with the backend DriversUnited service.	The DriversUnited app queries the backend service to fetch current job offers similar to the ones entered by the driver.
Display offers	The DriversUnited app must have successfully fetched job	The job offers are displayed on the DriversUnited app's interface, presenting basic

	offers from the backend service.	details such as job description, location, and payment.
Select offer	The driver must have accessed the list of job offers displayed on the DriversUnited app.	The driver selects a job offer from the list, triggering the app to submit the job data to its backend service and a suitable route planner to estimate payment details, distance, and time required for delivery.
Driver confirms job offer	The driver must have selected a job offer and received detailed payment information, along with the estimated route and delivery parameters.	The driver makes an informed decision and confirms the job offer through the DriversUnited app, indicating readiness to proceed with the delivery.
Job acceptance and execution	The driver must have confirmed the job offer through the DriversUnited app. The delivery service app must receive confirmation from the DriversUnited backend service.	The driver formally accepts the job offer using the delivery service app and begins the delivery process according to the provided details.
Feedback and reporting	The driver must have completed the delivery process using the delivery service app.	After completing the job, the driver can provide feedback or report any discrepancies via the DriversUnited app, which will be logged and processed by the backend service for further action or improvement.

These preconditions and effects outline the necessary conditions and outcomes for each operation involved in the DriversUnited app and its backend service, ensuring smooth functionality and user interaction throughout the service modelling process.