CTIS 222 Object Oriented Analysis and Design Spring 2019-2020 **Homework # 3 (3rd Iteration of the Project)** **(15% of the total contribution)** **Due Date: 13 June 2020 Saturday 23:55 (Final Exam Date)**			
Name Surname:	Muhammad Mustafa	Syed Abdullah Hassan	
Student ID:	21801112	21801072	

A) Project Title: BILKENT UBER

B) Project Description:

Travelling between campuses, going to Bilkent Centre has been a regular concern for students, especially those who live on-campus. This situation is worsened by the decrease in the number of buses. The solution to this is to enable people with cars to connect with those who don't. Enabling this coordination would result in easier and more convenient travel throughout the University.

SCOPE:

- This system is for Bilkent University students and faculty. The system would work inside the campus and accommodate students and faculty present on campus and outside the campus.

USERS:

- The users would be Bilkent Students and Faculty.

HIGH-LEVEL REQUIREMENTS:

- The system shall enable car-owners to plan trips with their counterparts before-hand and set up pick-up points for easier conveyance.
- The system shall match destination and starting locations and match riders with drivers.
- The system shall suggest people with those who have similar schedules.
- The system shall require monthly subscription from riders.
- The system shall reward drivers with VPs (Virtual Points), the accumulation of which would lead to monthly winners getting gift cards and other rewards.

- Upon reaching certain milestones, the system shall reward drivers with additional gift cards.

Updated B) Project Description:

Travelling between campuses, going to Bilkent Center has been a regular concern for students, especially those who live on-campus. This situation is worsened by the decrease in the number of buses. The solution to this is to enable people with cars to connect with those who do not have cars. Providing this feasible alternative would result in easier and more convenient travel throughout the University.

SCOPE:

- This system is for Bilkent University students and faculty. The system would work inside and outside the campus but only accommodate students and faculty of Bilkent University.

USERS:

- The users would be Bilkent University's Students and Faculty.

MULTIPLICITIES:

- One driver can select one or more rider to give a ride to.
- One rider can only select one driver to recieve a ride from.
- The system(app) shall manage multiple riders and drivers at one time.
- Many drivers and riders can use the system(app) at any given time.

BUSINESS RULES:

- All drivers and riders must belong to Bilkent University.

- A driver with the maximum points collected monthly shall recieve virtual gift coupons.
- A rider must be subscribed to the system.
- A driver and rider shall keep location services on throughout the duration of their rides.

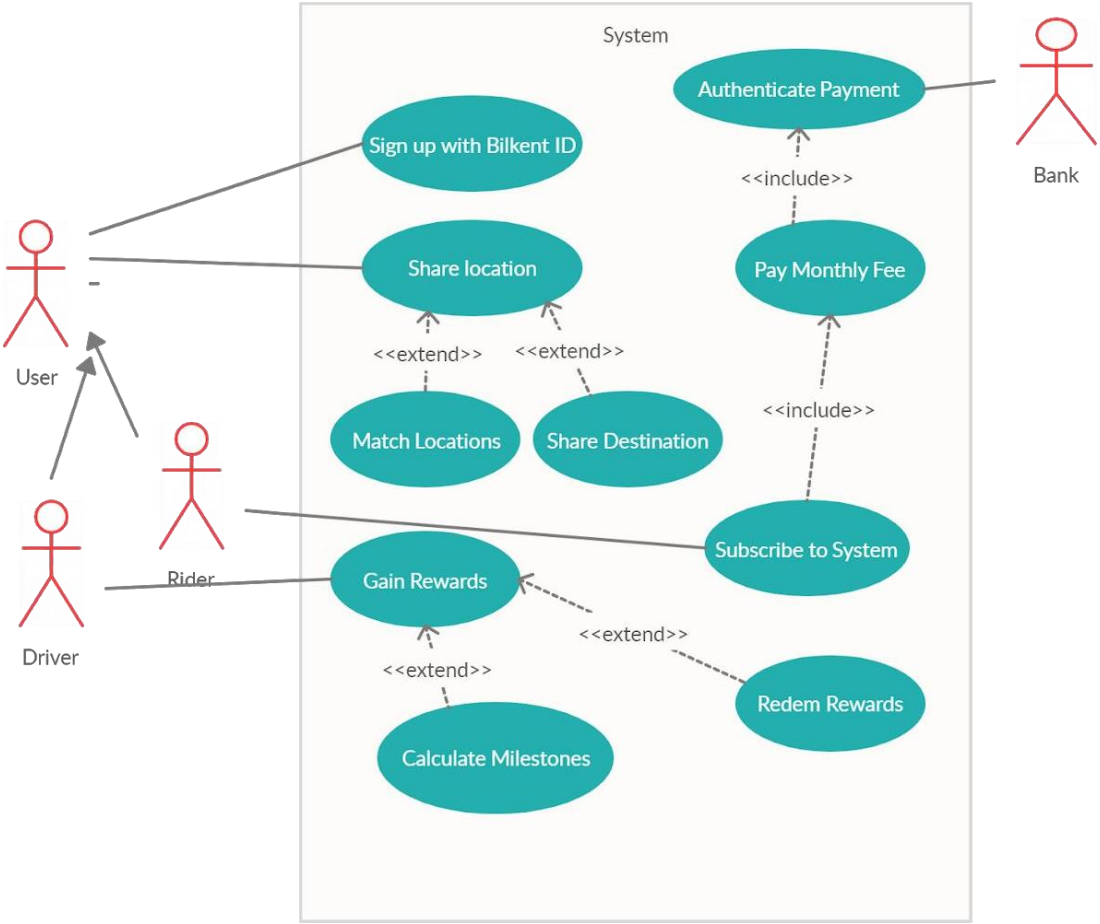
HIGH-LEVEL REQUIREMENTS:

- The system shall enable car-owners to plan trips with their counterparts before-hand and set up pick-up points for easier conveyance.
- The Bilkent email address will be necessary to confirm that the user is either from faculty or a student from Bilkent.
- The system shall match available riders to available drivers, when the drivers search for nearby riders, and vice versa.
- The system shall record ride times, save frequently visited destinations for later suggestions.
- The system shall require monthly subscription from riders.
- The system shall reward drivers with VPs (Virtual Points), the accumulation of which would lead to monthly winners getting gift cards and other rewards.
- Upon reaching certain milestones, the system shall reward drivers with additional gift cards.
- There is an assumption that the algorithm to calculate the reward points will stay constant.

C) Identify actors and use cases for your system and show them on a UML Use Case Diagram (System Boundary, use cases,

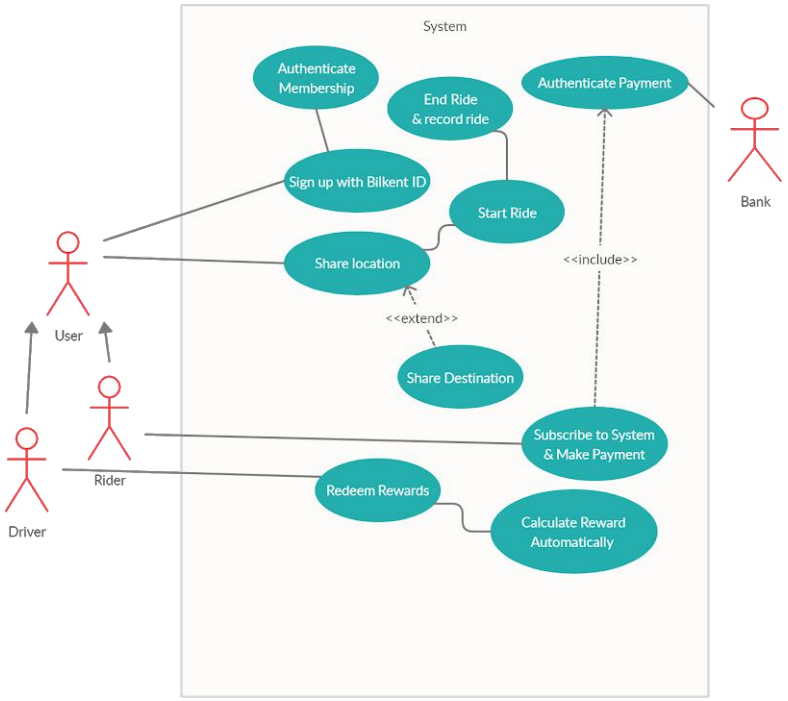
actors, associations, <<include>>, <<extend>>, generalization).

Draw a UML Use Case Diagram of your project.



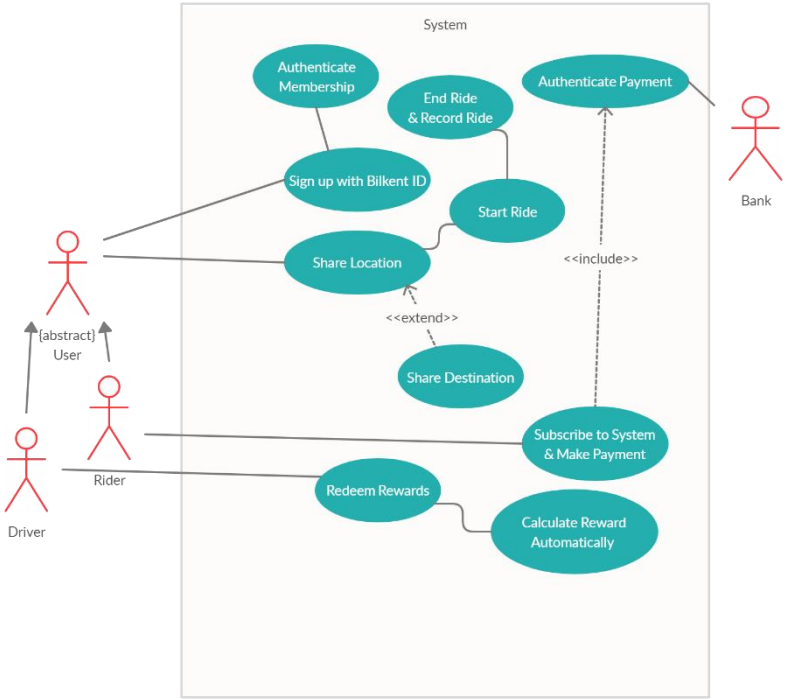
Updated C) Identify actors and use cases for your system and show them on a UML Use Case Diagram (System Boundary, use

cases, actors, associations, <<include>>, <<extend>>, generalization).



Updated 2.0 C) Identify actors and use cases for your system and show them on a UML Use Case Diagram (System Boundary, use

cases, actors, associations, <<include>>, <<extend>>, generalization).



D) After finalizing your use case diagram, elaborate (describe) <u>ALL use cases</u> of your system by making use of the <u>Use Case Template</u> (UC Description) on Moodle.

ID:	**UC-1**
Title:	**Sign up with Bilkent ID**

Description:	**Enable the user to sign up with their Bilkent Information**
Primary Actor:	**User (Driver, Rider)**
Preconditions:	**User must be part of Bilkent Staff or a Bilkent Student**
Post-conditions:	**Permanently signed up for the system**
Main Success Scenario: (Main Flow)	**1. User must be part of Bilkent Staff or a Bilkent Student** **2. User enters their Bilkent ID and WebMail** **3. The system checks whether the user is already present in the system or not?** **4. The system adds the user to the system.** **5. The user is successfully signed up for the app**
Extensions: (Alternative flow)	**1a. If the user's info does not include Bilkent ID, system rejects user**
Frequency of Use:	**Once per user**
Priority:	**High**

ID:	**UC-2**
Title:	**Subscribe to System**
Description:	**Riders (people without cars) choose subscription services in order to use the app**
Primary Actor:	**Riders**
Preconditions:	**User should be a rider**
Post-conditions:	**Rider is able to share location and use features of the app**
Main Success Scenario: (Main Flow)	**1. The user must be a rider to select a subscription type.** **2. The user has to select from weekly, monthly, yearly subscription** **3. The user enters credit card information** **4. The information is authenticated and the payment deducted** **5. The app features are now enabled to the rider**
Extensions: (Alternative flow)	**2a. The information given by the user could not be authenticated and is asked to resend the information**

Frequency of Use:	**Once per rider**
Priority:	**High**

ID:	**UC-3**
Title:	**Search for Riders**
Description:	**Drivers will search for Riders nearby**
Primary Actor:	**Driver**
Preconditions:	**Locations of rider and driver needs to be shared**
Post-conditions:	**Location of nearest riders is shared with drivers**
Main Success Scenario: (Main Flow)	**1. Locations are received from both rider and driver** **2. Driver searches for the nearest riders** **3. Driver is updated with the meeting point location**

Extensions: **(Alternative flow)**	**3a. When the nearest meeting point is not located, rider and driver would have to wait for the next match.**
Frequency of Use:	**Frequently, almost everyday**
Priority:	**High**

ID:	UC-4
Title:	Authenticate Payment
Description:	Payment details are checked by the bank
Primary Actor:	Bank
Preconditions:	Card details are entered by user
Post-conditions:	Payment will be confirmed or rejected
Main Success Scenario: (Main Flow)	1. Bank receives payment details of users 2. The details are checked 3. If the details are true, the payment is deducted 4. If the details are incorrect, a resend notification is sent back to the user
Extensions: (Alternative flow)	4a. Payment is deducted
Frequency of Use:	Once per user
Priority:	High

Explanation of fields

A) **Title:** Enter the goal of the use case – preferably as a short, active verb phrase.

B) **Description:** Describe the goal and context of this use case. This is usually an expanded version of what you entered in the "Title" field.

C) **Primary Actor:** A person or a software/hardware system that interacts with your system to achieve the goal of this use case.

D) **Precondition:** Describe the state the system is in before the first event/task in this use case.

E) **Post-condition:** Describe the state the system is in after all the events/tasks in this use case have taken place.

F) **Main Success Scenario:** This field contains the flow of events/tasks from preconditions to post-conditions, when nothing goes wrong.

G) **Extensions:** Describe all the other scenarios for this use case – including exceptions and error cases.

H) The other fields are self-explanatory.

ID:	UC-5
Title:	Calculate Rewards
Description:	Rewards are calculated according to the rides provided
Primary Actor:	System
Preconditions:	The user should be a driver
Post-conditions:	The rewards are displayed to the user
Main Success Scenario: (Main Flow)	1. The system confirms the user is a driver 2. The ride stats are checked and the rewards are calculated accordingly 3. The rewards are then displayed to the user
Extensions: (Alternative flow)	
Frequency of Use:	Every time the user presses calculate rewards
Priority:	medium

Explanation of fields

- **Title:** Enter the goal of the use case – preferably as a short, active verb phrase.
- **Description:** Describe the goal and context of this use case. This is usually an expanded version of what you entered in the "Title" field.

- **Primary Actor:** A person or a software/hardware system that interacts with your system to achieve the goal of this use case.
- **Precondition:** Describe the state the system is in before the first event/task in this use case.
- **Post-condition:** Describe the state the system is in after all the events/tasks in this use case have taken place.
- **Main Success Scenario:** This field contains the flow of events/tasks from preconditions to post-conditions, when nothing goes wrong.
- **Extensions:** Describe all the other scenarios for this use case – including exceptions and error cases.
- The other fields are self-explanatory.

Updated D) After finalizing your use case diagram, elaborate (describe) ALL use cases of your system by making use of the Use Case Template (UC Description) on Moodle.

ID:	**UC-1**
Title:	**Sign up with Bilkent ID**
Description:	**Enable the user to sign up with their Bilkent Information**
Primary Actor:	**User (Driver, Rider)**
Preconditions:	**User must be part of Bilkent Staff or a Bilkent Student**
Post-conditions:	**Permanently signed up for the system**
Main Success Scenario: (Main Flow)	**1. User must be part of Bilkent Staff or a Bilkent Student** **2. User enters their Bilkent ID and WebMail** **3. The system checks whether the user is already present in the system or not?** **4. The system adds the user to the system.** **5. The user is successfully signed up for the app**
Extensions: (Alternative flow)	**1a. If the user's info does not include Bilkent ID, system rejects user**
Frequency of Use:	**Once per user**

Priority:	**High**

ID:	**UC-2**
Title:	**Subscribe to System and make payment**
Description:	**Riders (people without cars) choose subscription services in order to use the app**
Primary Actor:	**Riders**
Preconditions:	**User should be a rider**
Post-conditions:	**Rider is able to share location and use features of the app**
Main Success Scenario: (Main Flow)	**1. The user must be a rider to select a subscription type.** **2. The user has to select from weekly, monthly, yearly subscription** **3. The user enters credit card information** **4. The information is authenticated and the payment deducted** **5. The app features are now enabled to the rider**
Extensions: (Alternative flow)	**2a. The information given by the user could not be authenticated and is asked to resend the information**
Frequency of Use:	**Once per rider**
Priority:	**High**

ID:	**UC-3**
Title:	**Search for Riders**
Description:	**Drivers will search for Riders nearby**
Primary Actor:	**Driver**
Preconditions:	**Locations of rider and driver needs to be shared**
Post-conditions:	**Location of nearest riders is shared with drivers**
Main Success Scenario: (Main Flow)	**1. Locations are received from both rider and driver** **2. Driver searches for the nearest riders** **3. Driver is updated with the meeting point location**
Extensions: (Alternative flow)	**3a. When the nearest meeting point is not located, rider and driver would have to wait for the next match.**
Frequency of Use:	**Frequently, almost everyday**
Priority:	**High**

ID:	4
Title:	Authenticate Payment
Description:	Payment details are checked by the bank
Primary Actor:	Bank
Preconditions:	Card details are entered by user
Post-conditions:	Payment will be confirmed or rejected
Main Success Scenario: (Main Flow)	1. Bank receives payment details of users 2. The details are checked 3. If the details are true, the payment is deducted 4. If the details are incorrect, a resend notification is sent back to the user

Extensions: (Alternative flow)	4a. Payment is deducted
Frequency of Use:	Once per user
Priority:	High

Explanation of fields

A. **Title:** Enter the goal of the use case – preferably as a short, active verb phrase.
B. **Description:** Describe the goal and context of this use case. This is usually an expanded version of what you entered in the "Title" field.
C. **Primary Actor:** A person or a software/hardware system that interacts with your system to achieve the goal of this use case.
D. **Precondition:** Describe the state the system is in before the first event/task in this use case.
E. **Post-condition:** Describe the state the system is in after all the events/tasks in this use case have taken place.
F. **Main Success Scenario:** This field contains the flow of events/tasks from preconditions to post-conditions when nothing goes wrong.
G. **Extensions:** Describe all the other scenarios for this use case – including exceptions and error cases.
H. The other fields are self-explanatory.

ID:	5
Title:	Calculate Rewards
Description:	Rewards are calculated according to the rides provided
Primary Actor:	System
Preconditions:	The user should be a driver
Post-conditions:	The rewards are displayed to the user
Main Success Scenario: (Main Flow)	1. The system confirms the user is a driver 2. The ride stats are checked and the rewards are calculated accordingly 3. The rewards are then displayed to the user
Extensions: (Alternative flow)	
Frequency of Use:	Every time the user presses calculate rewards
Priority:	medium

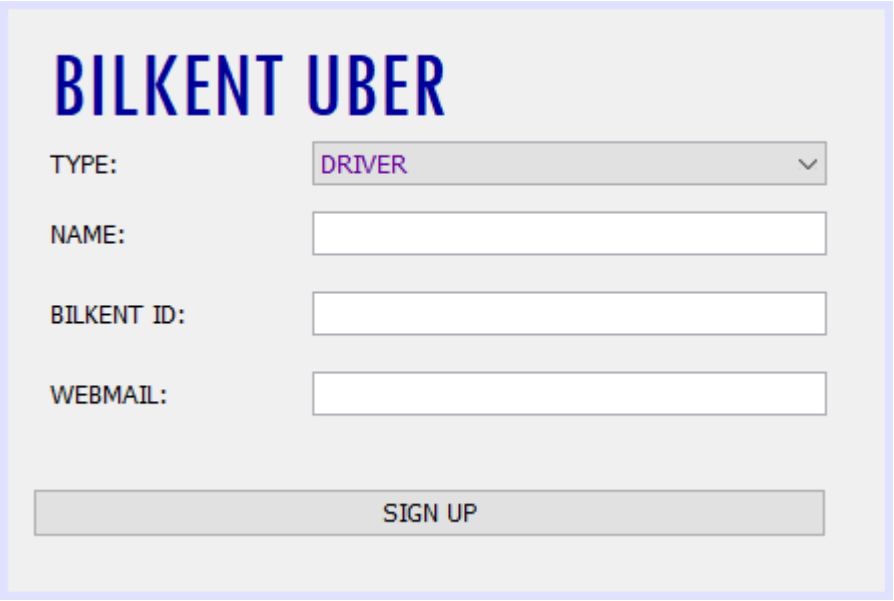
ID:	**6**
Title:	**Record Ride**
Description:	**The duration of the ride is recorded by the system**
Primary Actor:	**System, Driver**

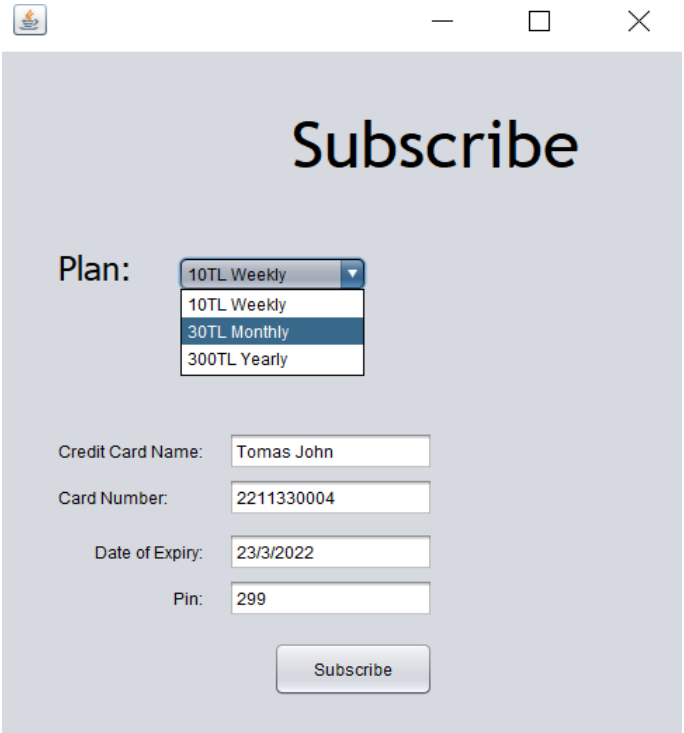
Preconditions:	**Ride needs to be started by driver**
Post-conditions:	**Ride needs to be ended by driver**
Main Success Scenario: **(Main Flow)**	**1. Driver starts the ride, only possible when rider and driver have the same location.** **2. The system logs in start time of the ride** **3. The system logs in end time of the ride, and calculates the ride time.**
Extensions: **(Alternative flow)**	**4a. If the ride is not started, the system sends a notification at intervals to driver when the rider and driver have a shared location (meaning they are in the vehicle)**
Frequency of Use:	**Duration of each ride**
Priority:	**High**

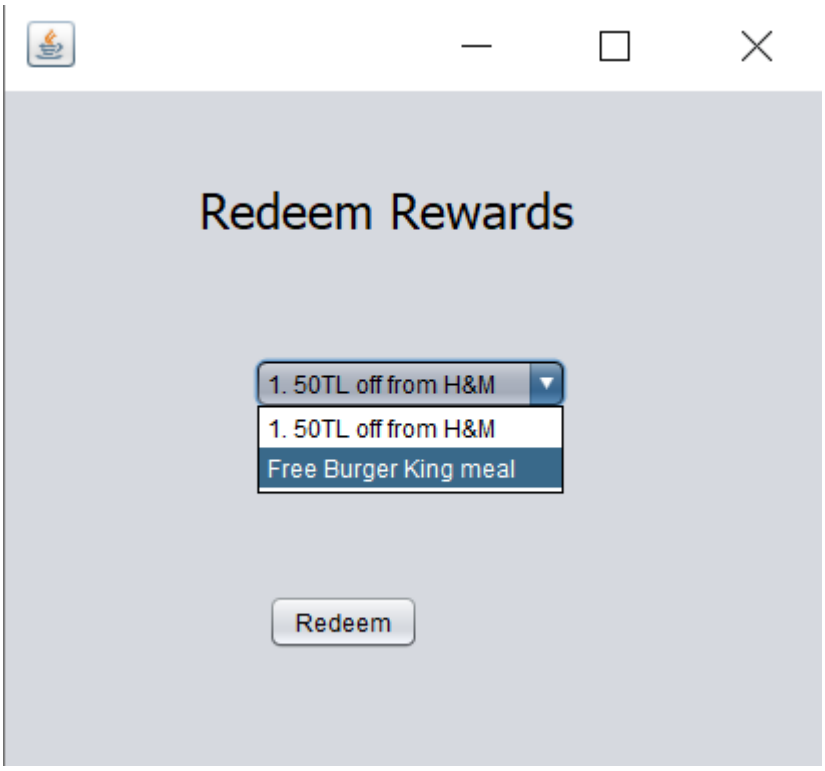
Explanation of fields

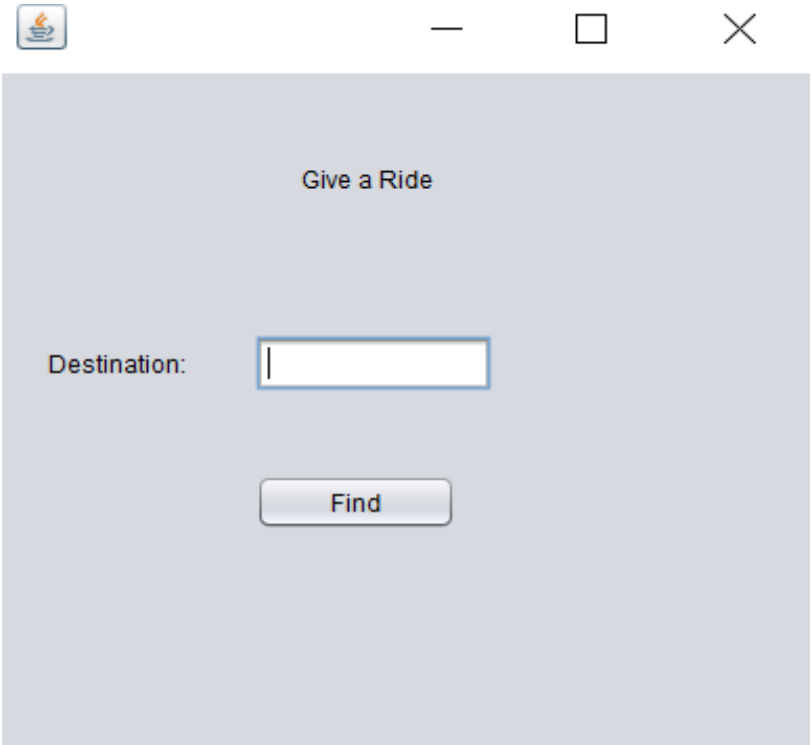
- **Title:** Enter the goal of the use case – preferably as a short, active verb phrase.
- **Description:** Describe the goal and context of this use case. This is usually an expanded version of what you entered in the "Title" field.
- **Primary Actor:** A person or a software/hardware system that interacts with your system to achieve the goal of this use case.
- **Precondition:** Describe the state the system is in before the first event/task in this use case.
- **Post-condition:** Describe the state the system is in after all the events/tasks in this use case have taken place.
- **Main Success Scenario:** This field contains the flow of events/tasks from preconditions to post-conditions, when nothing goes wrong.
- **Extensions:** Describe all the other scenarios for this use case – including exceptions and error cases.
- The other fields are self-explanatory.

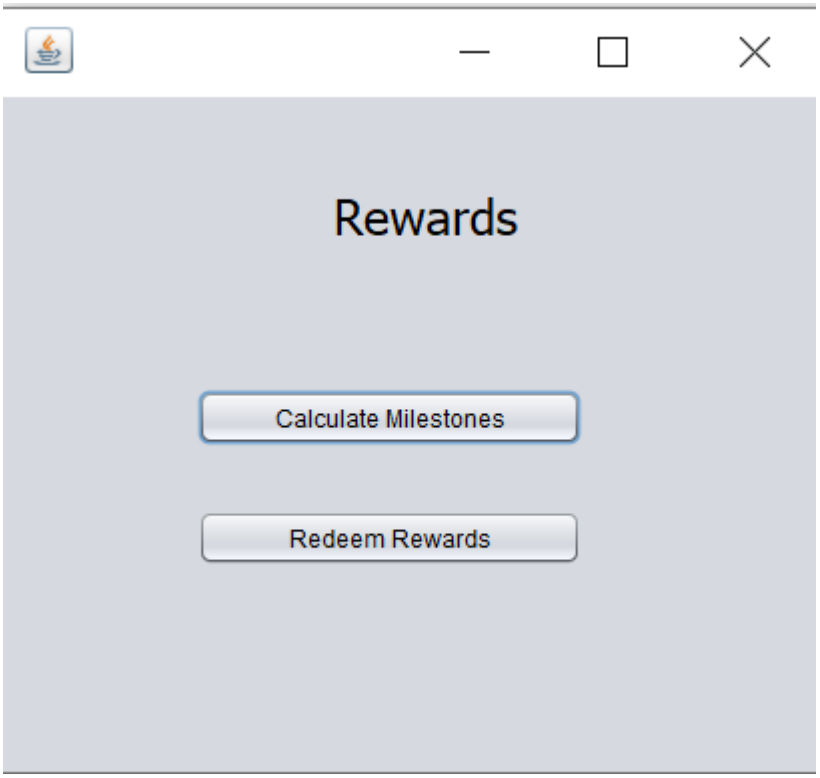
E) Draw a Mock-up for each use case. You may have more (or sometimes less) mock-ups than the number of use cases.

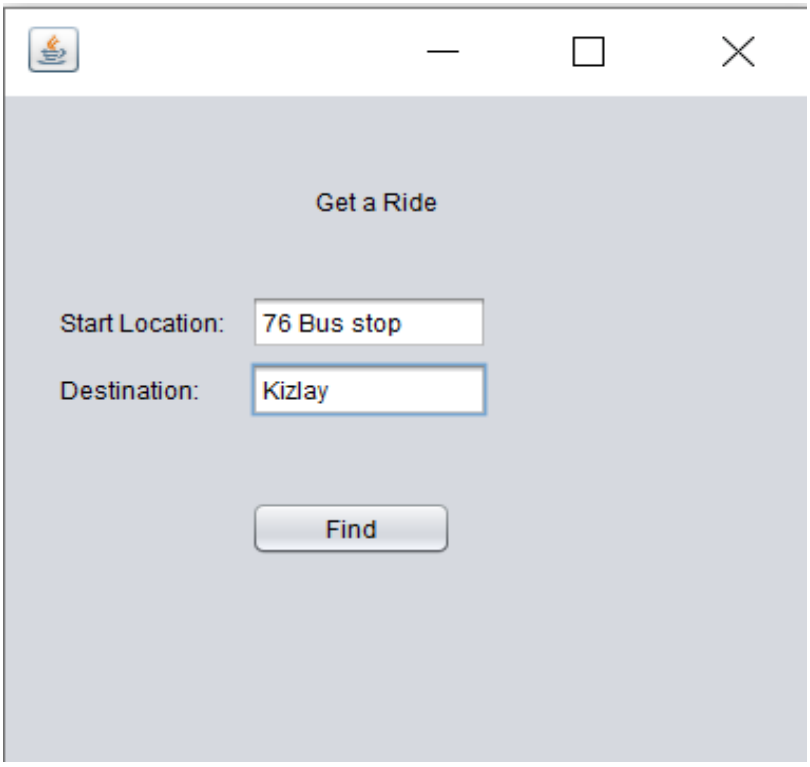



Subscribe
Plan:
10TL Weekly
10TL Weekly
30TL Monthly
300TL Yearly
Credit Card Name:
Tomas John
Card Number:
2211330004
Date of Expiry:
23/3/2022
Pin:
299
Subscribe


Redeem Rewards
1. 50TL off from H&M
1. 50TL off from H&M
Free Burger King meal
Redeem

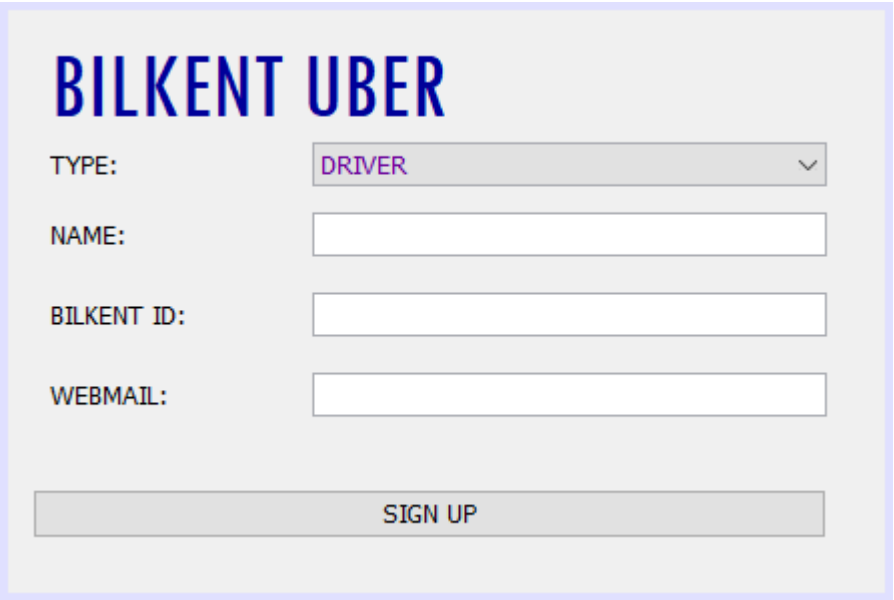

Give a Ride
Destination:
Find


Rewards
Calculate Milestones
Redeem Rewards


Get a Ride
Start Location:
76 Bus stop
Destination:
Kizlay
Find

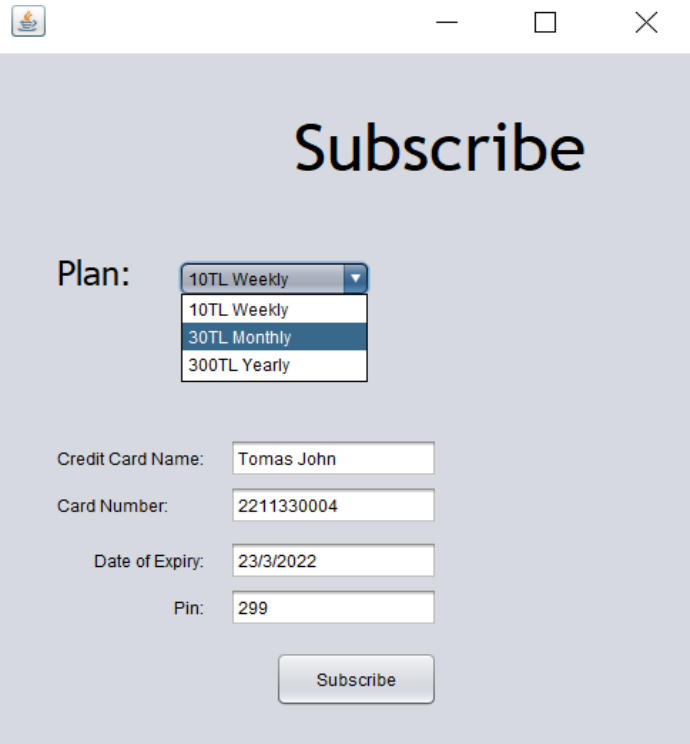
Updated E) Draw a Mock-up for each use case. You may have more (or sometimes less) mock-ups than the number of use cases.

NetBeans is used to make the GUI

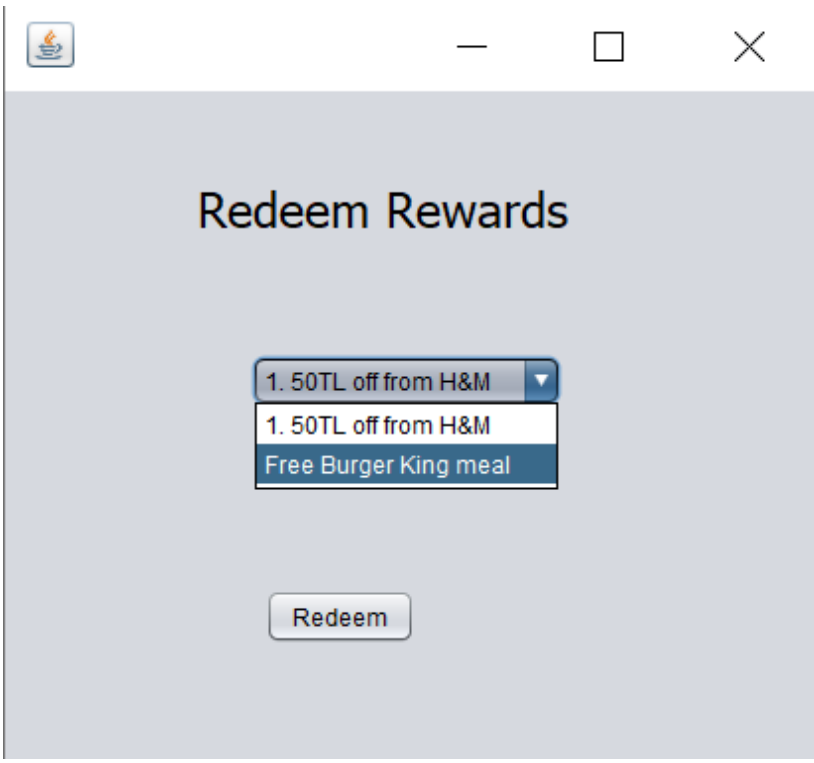


HOMESCREEN

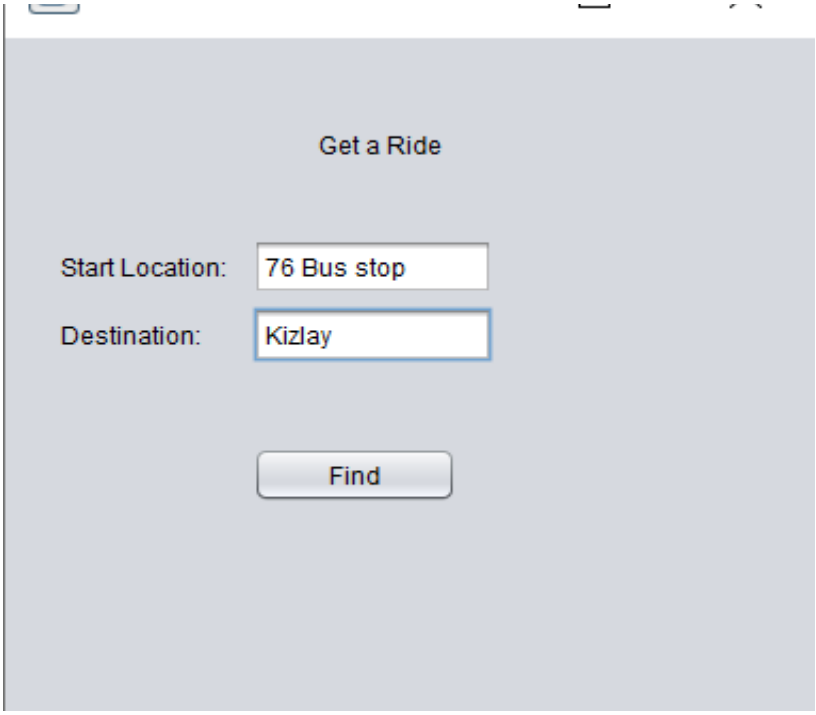
SUBSCRIBE

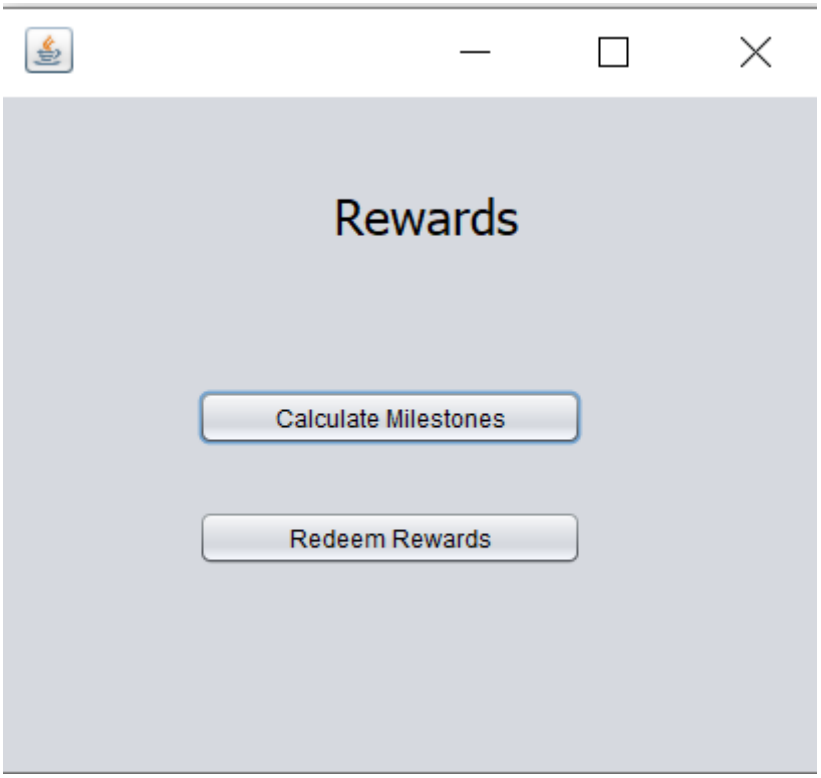


REDEEM REWARDS

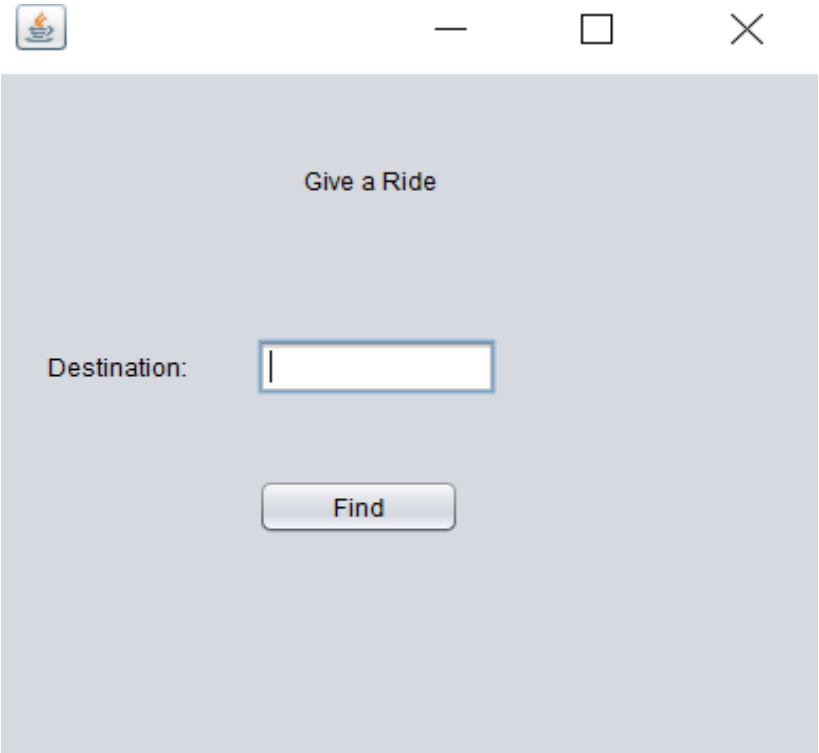


RIDE FINDER

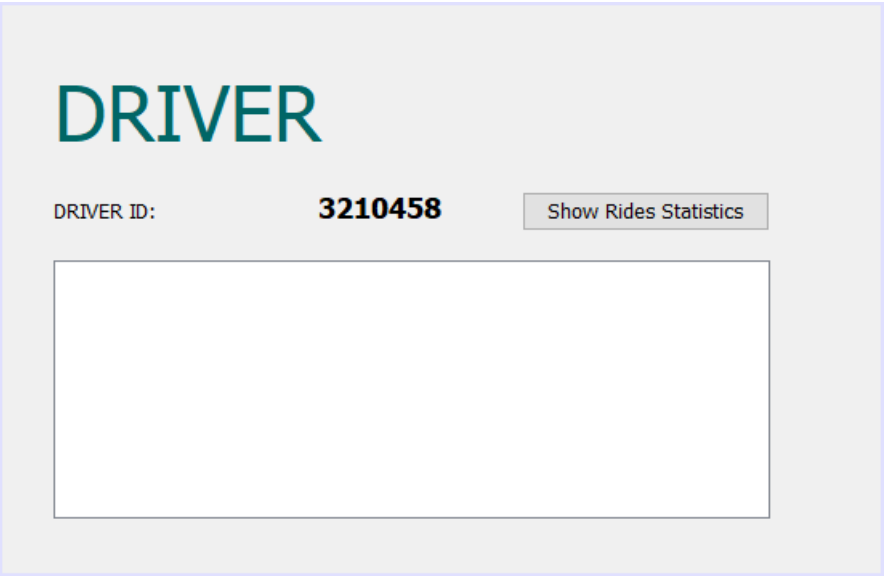




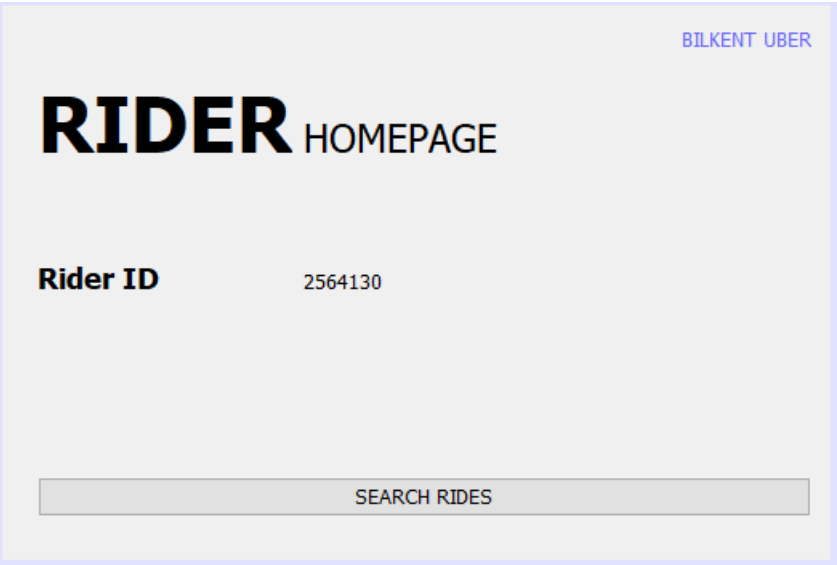
REWARDS



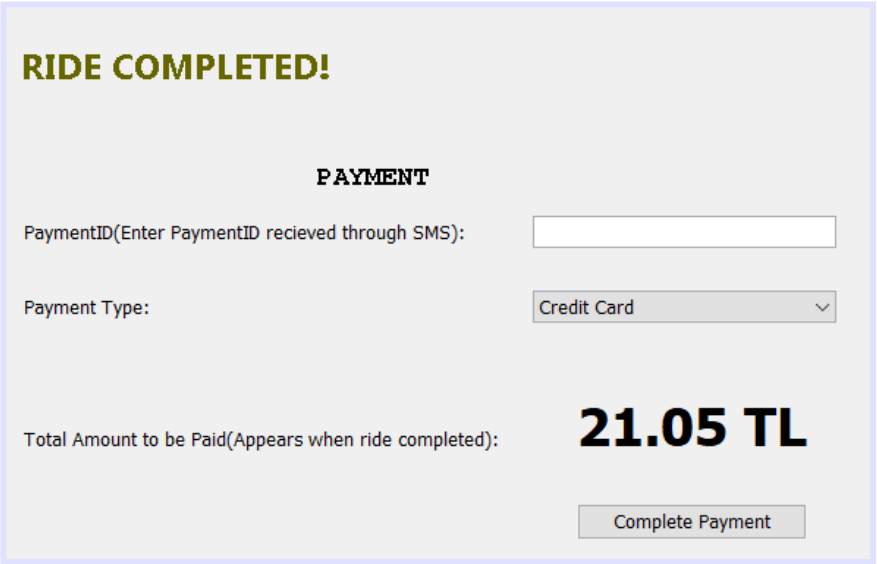
RIDE GIVER



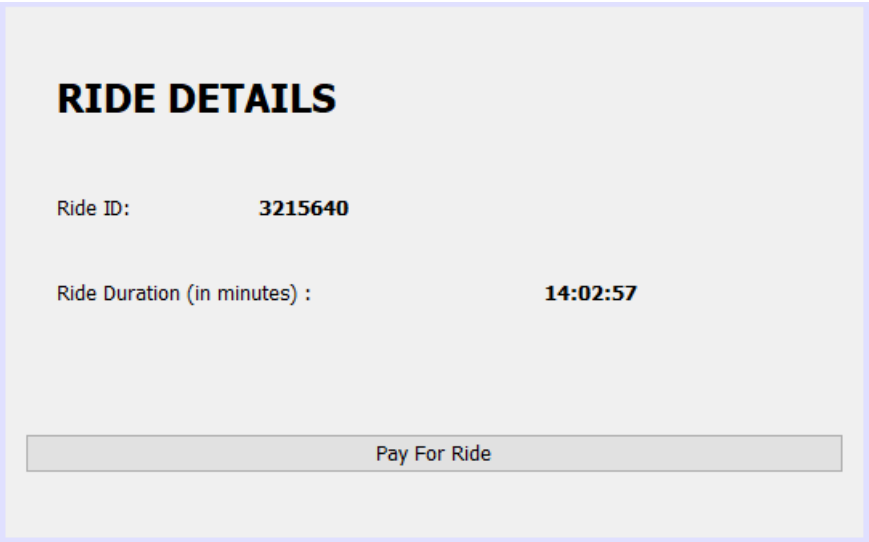
DRIVER HOME PAGE



Rider Home Page



Payment Screen



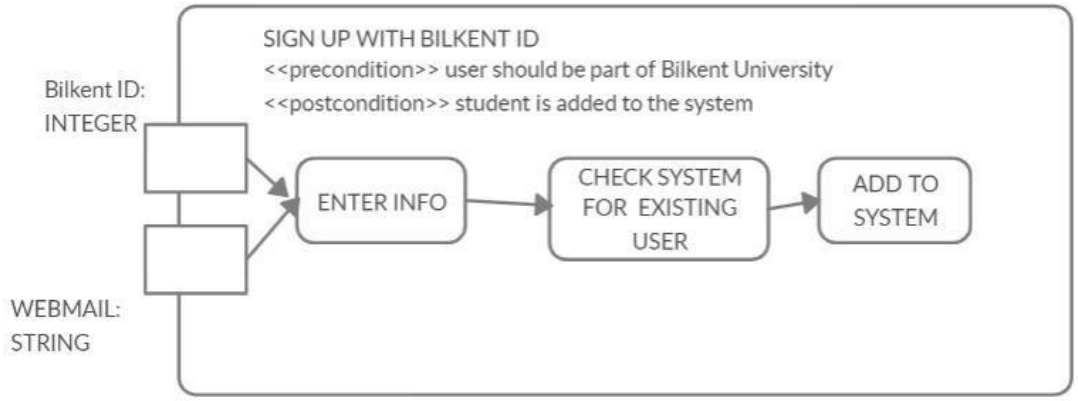
RIDE DETAILS

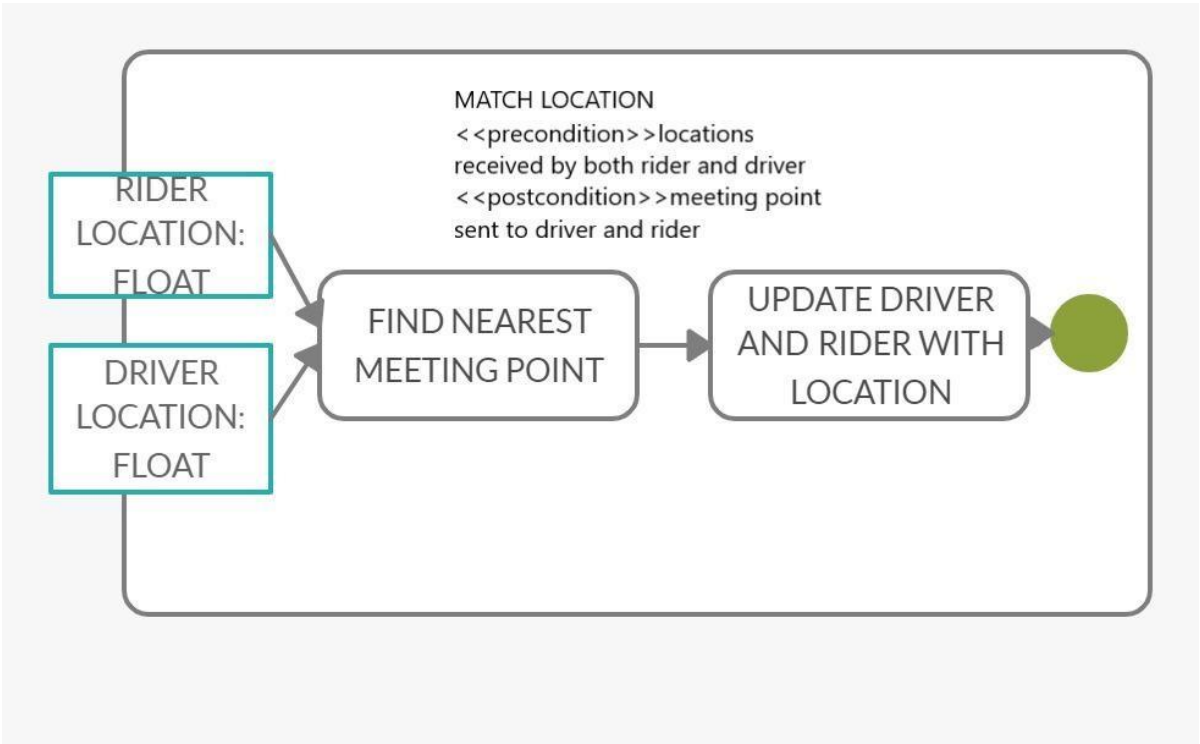
F) UML Activity Diagram

After *finalizing* your UC templates (descriptions), model 3 most important (core) UCs using UML Activity Diagram.

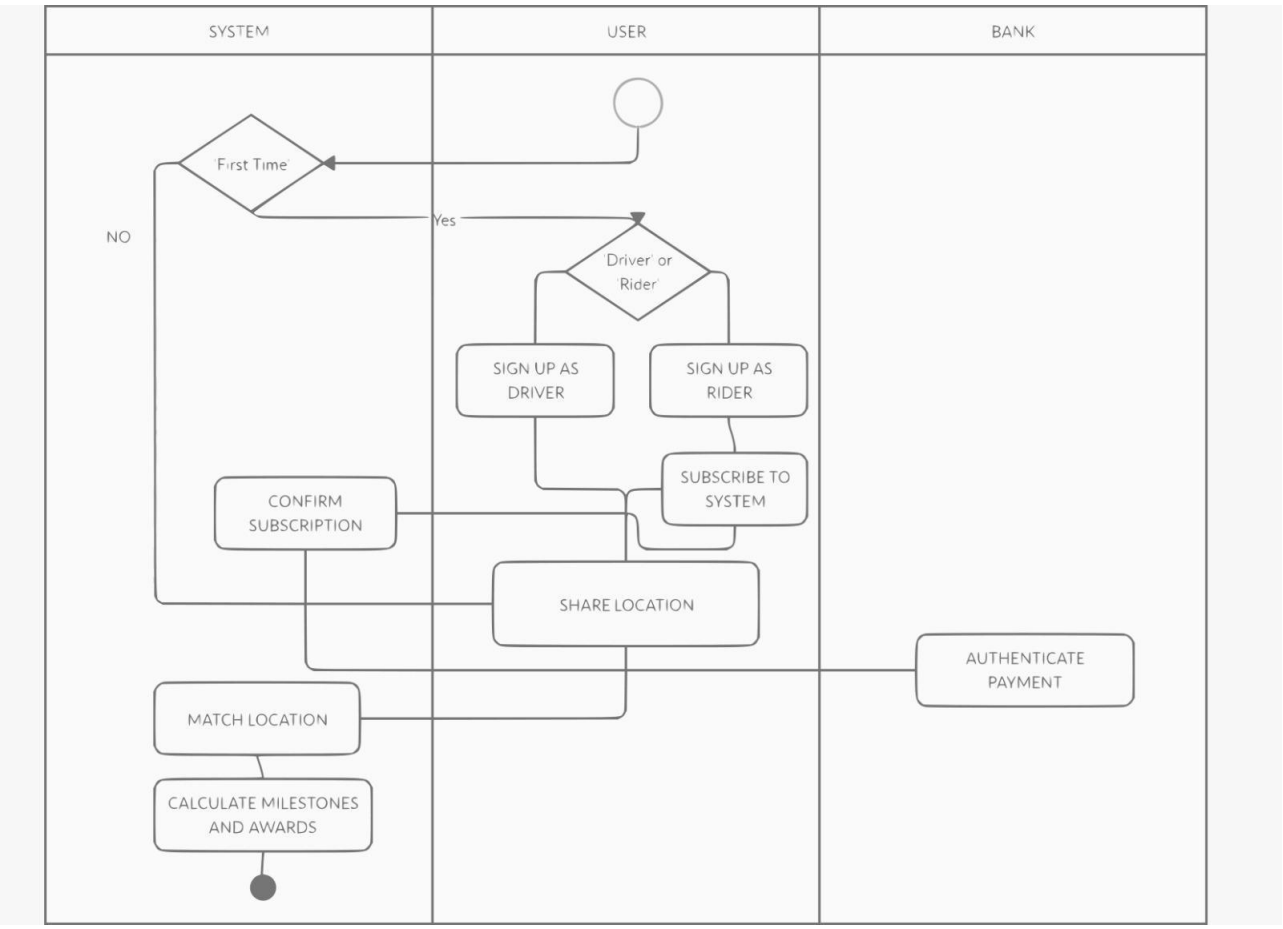
While filling in the UC templates in the Section D, you must have already identified the priority of each UC. Herein, you can use this priority value mentioned in these templates to determine which UC's activity diagram you should draw.

You ALSO need to model alternative (and exceptional) flows of the chosen UCs.

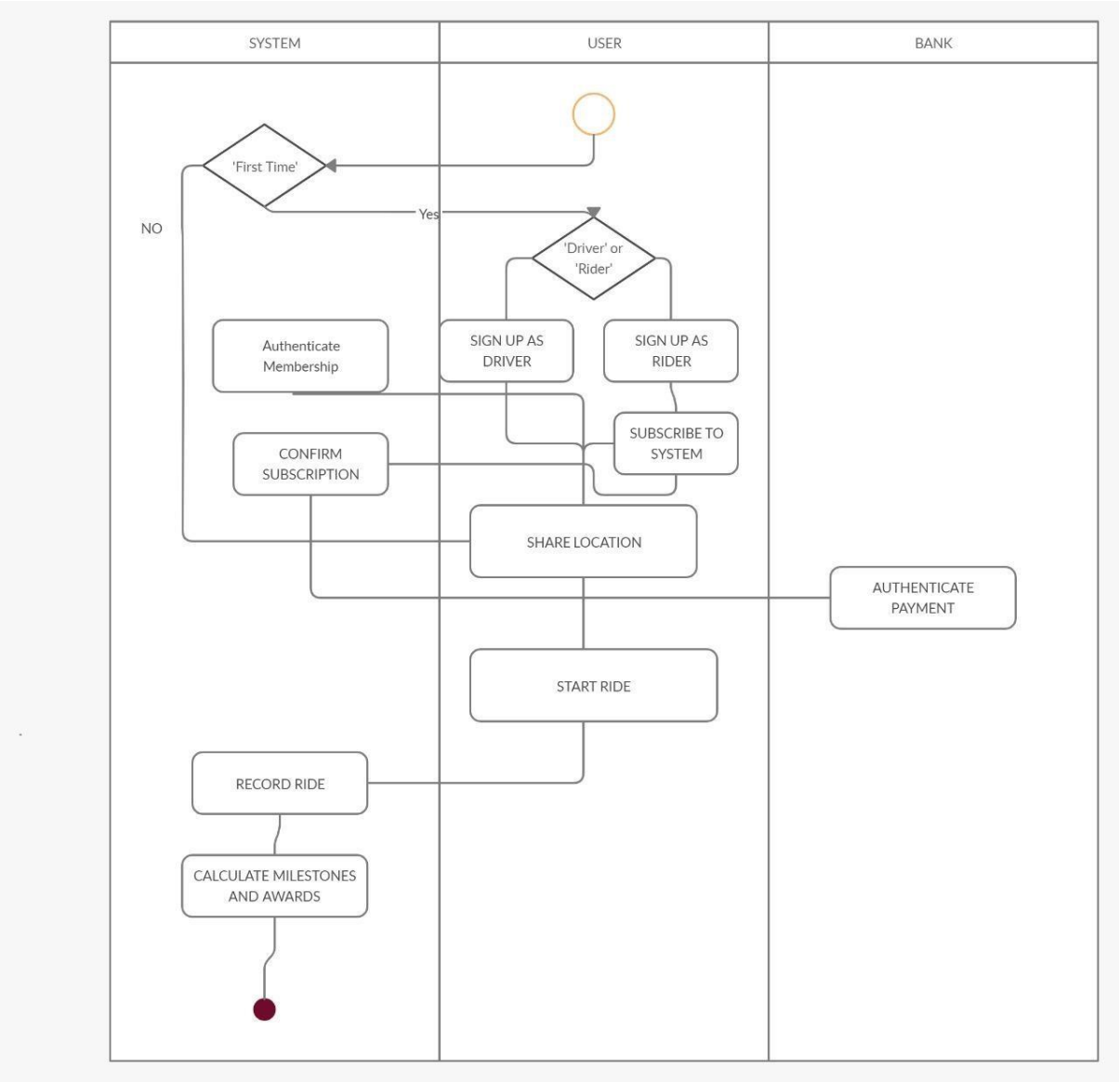

SIGN UP WITH BILKENT ID
<<precondition>> user should be part of Bilkent University
<<postcondition>> student is added to the system
Bilkent ID:
INTEGER
WEBMAIL:
STRING
ENTER INFO
CHECK SYSTEM FOR EXISTING USER
ADD TO SYSTEM


MATCH LOCATION
<<precondition>>locations received by both rider and driver
<<postcondition>>meeting point sent to driver and rider
RIDER LOCATION: FLOAT
DRIVER LOCATION: FLOAT
FIND NEAREST MEETING POINT
UPDATE DRIVER AND RIDER WITH LOCATION

Extra:


SYSTEM
USER
BANK
'First Time'
Yes
NO
'Driver' or 'Rider'
SIGN UP AS DRIVER
SIGN UP AS RIDER
SUBSCRIBE TO SYSTEM
CONFIRM SUBSCRIPTION
SHARE LOCATION
AUTHENTICATE PAYMENT
MATCH LOCATION
CALCULATE MILESTONES AND AWARDS

Updated F) UML Activity Diagram



F) List of Domain Classes

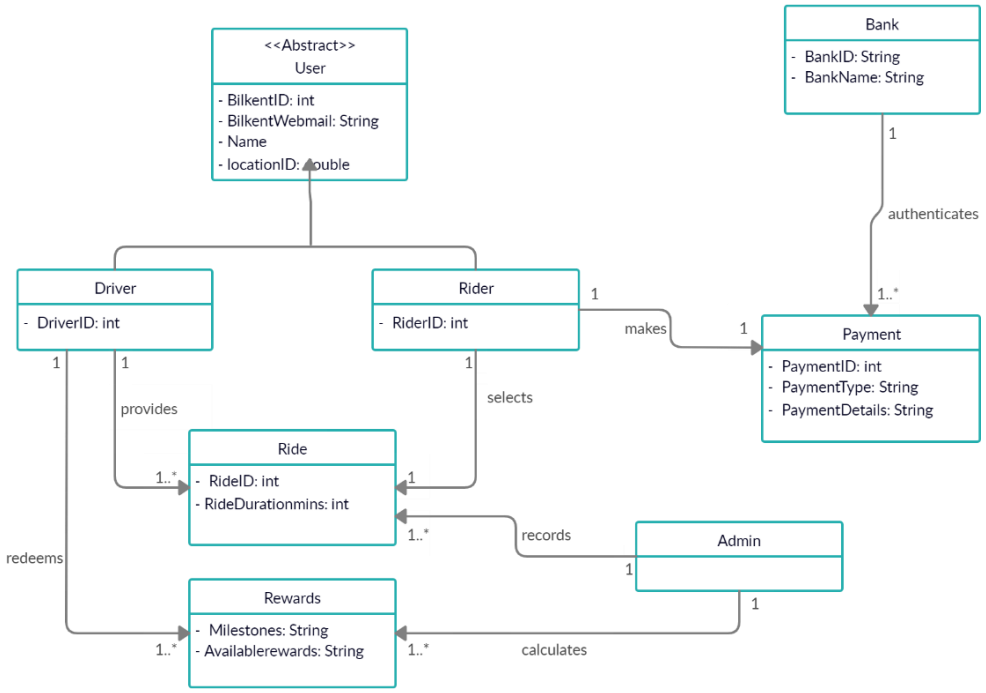
- Driver
- Rider
- Admin
- Bank

Updated G) List of Domain Classes

- Driver
- Rider
- Admin
- Bank
- Ride

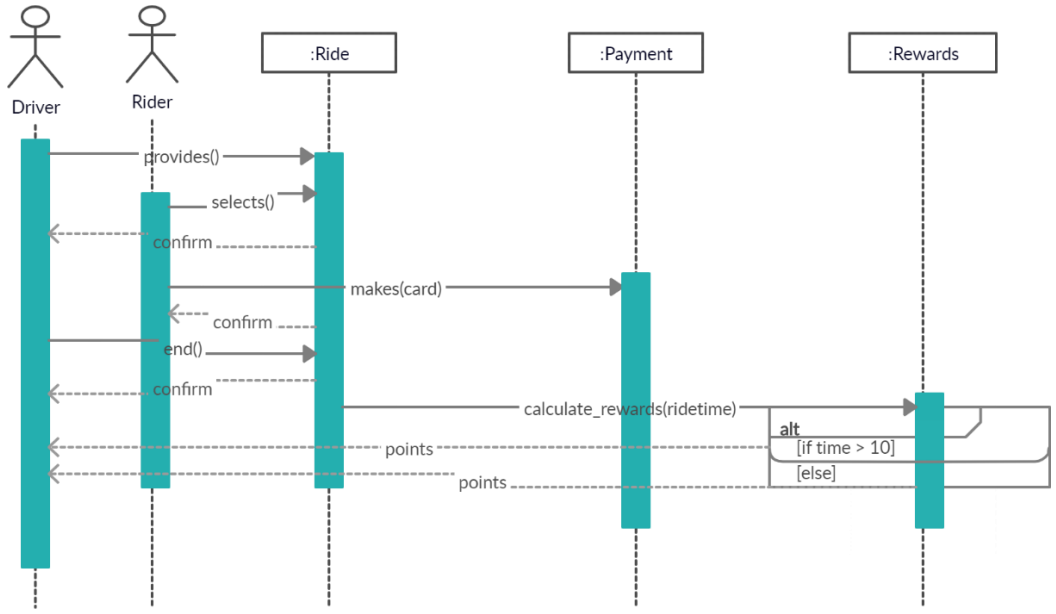
H) UML Class Diagram (New Section)

Draw the UML Class diagram (class names, attributes, multiplicities, navigabilities, role names, method names (optional but it will ease your job during the design)) to model your domain classes in your project. Domain classes should NOT include any system, persistence storage, GUI, and design pattern classes.



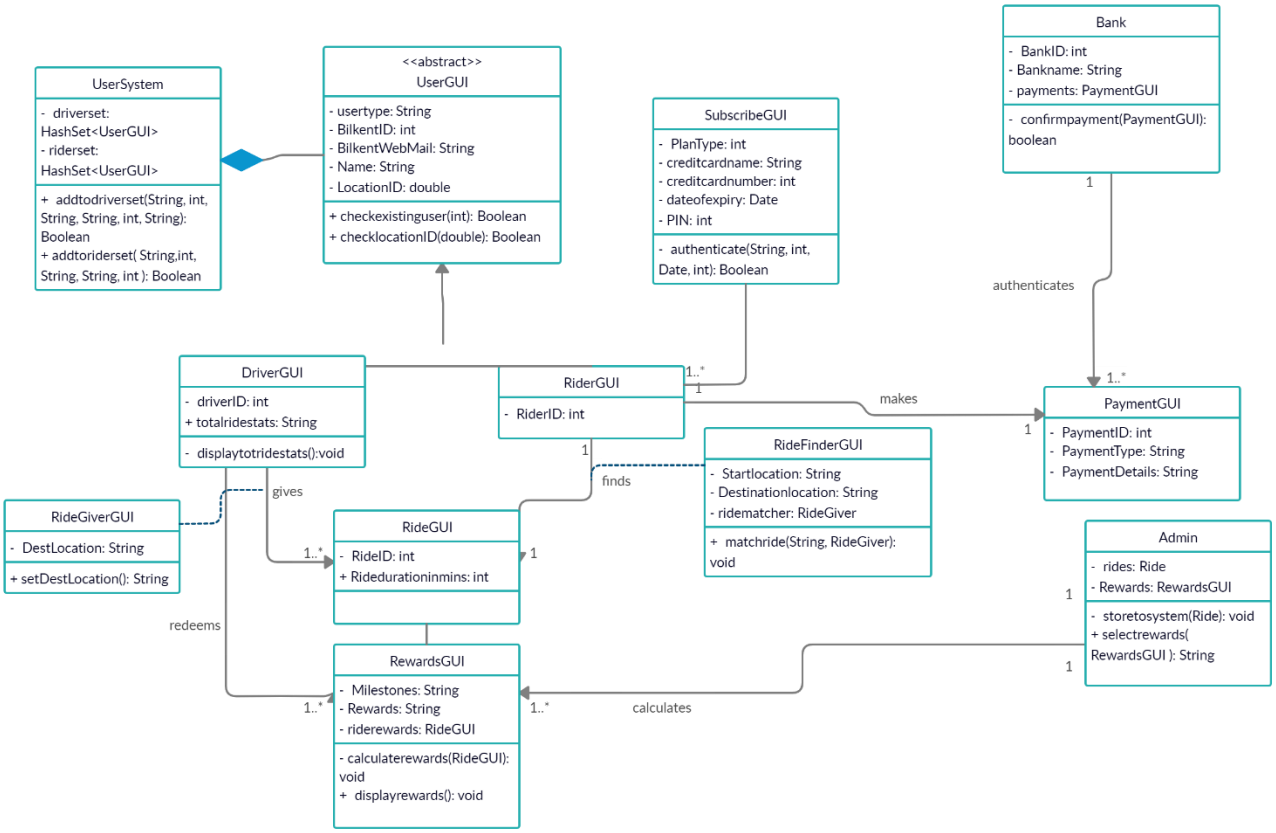
I) UML Sequence Diagram (New Section)

Model the interaction of <u>3 most important (core) UCs</u> with UML Sequence diagrams (SD). (You need to have at least 1 SD for each UC.)

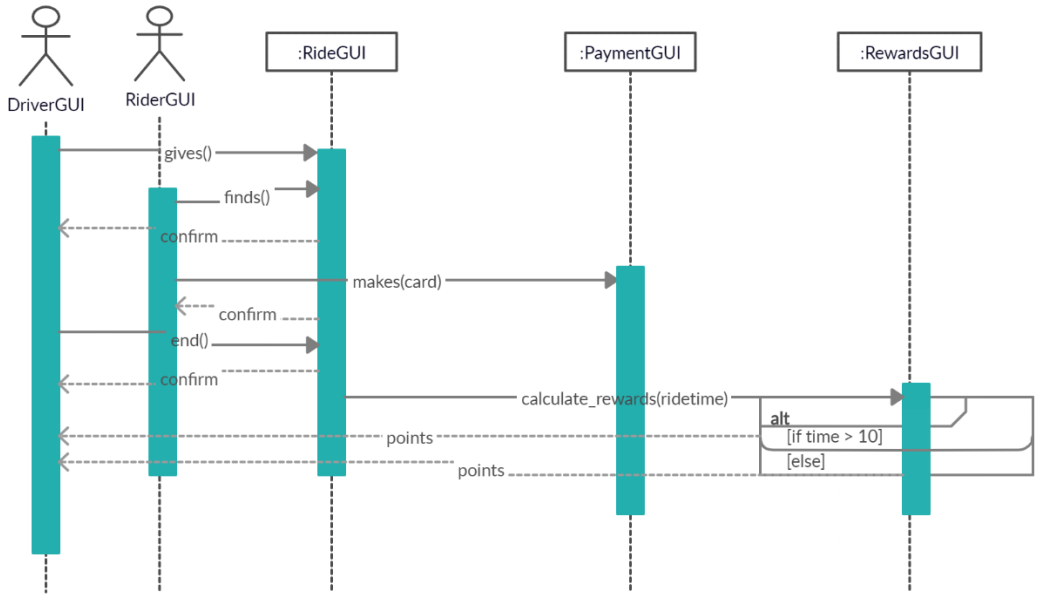


Updated H) UML (Design) Class Diagram (New Section)

Model the detailed-design of your application by making use of UML Class Diagram.



Updated I) UML (Design-Level) Sequence Diagram (New Section) Update the selected UML Sequence diagrams (the same SDs) based on your detailed UML Class Diagram.



J) Write down the names of the OOD principles, guidelines, ideas, and patterns (SOLID, GRASP, MVC etc.,) you used while modeling the detailed design of the project. You need to <u>justify your design decisions</u>. (New Section)

1	High Cohesion	UserGUI, RideGUI, PaymentGUI, SubscribeGUI, RideFinderGUI, RideGiverGUI, RewardsGUI	This allows for focused and strongly related classes. Breaking larger classes into these smaller ones increases cohesion.
2	Information Expert	UserGUI	UserGUI has the majority of the knowledge of its child classes and therefore decreases repetition.
3	Single Responsibility principle	RideGUI,RideFinderGUI, RideGiverGUI, RewardsGUI, PaymentGUI, SubscribeGUI	The mentioned classes have been given a single responsibility each so as to follow the single responsibility principles which enables easier understanding and proper division of labor between the classes
4	Open/ Close Principle	UserGUI	UserGUI class is made abstract following the Open Closed principle so that any new additions can be made in the parent class, not affecting the child classes i.e. DriverGUI and Rider.
5	Dependency	UserGUI	This class is made

	Inversion Principle		**abstract so that private methods and attributes can be protected and instead modifications can be made to the abstract class without interfering with the methods of the child classes.**