MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Bike Rescue

Group KH2L	
Group members	Ngô Thành Lộc – SE130721 (Leader) Đặng Văn Kiên – SE130494 Nguyễn Thanh Hiếu – SE140062 Chu Nguyên Linh – SE130515
Supervisor	Lại Đức Hùng
Ext. Supervisor	N/A

– **Ho Chi Minh City, 22 July, 2020** –

Introduction

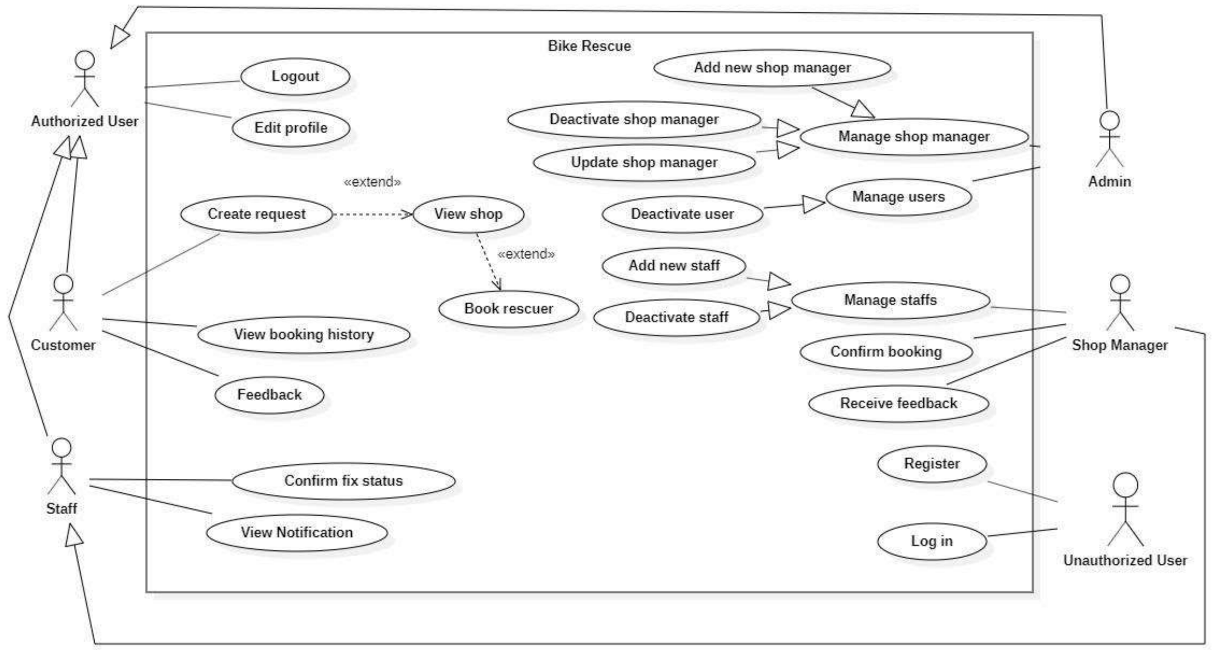
1. Project Information

- Project name: **Bike Rescue**
- Project Code: **BR**
- Start Date: **July 20th, 2020**
- End Date: **--**

2. Introduction

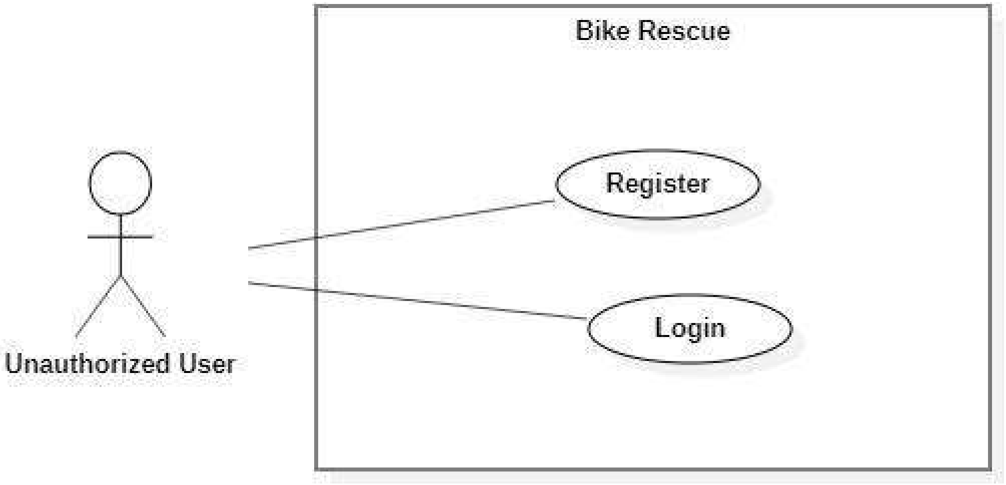
Today society develops, the demand for material facilities of people is increasing. The same goes for travel, the need for travel is increasing. Therefore, the means of transport can be damaged at any time. If you are lucky, you will see store repair is there, but sometimes the car has to drive a few kilometers, even a few hundred kilometers is not available. For that reason, our team has the idea to create an app to help road users not worry about vehicle damage anywhere. When you have broken down the car, just open the app, locate your location and book, we will have a car repaired on the spot if the damage is minor. Heavy will be rescued by a rescue vehicle, it will take your vehicle and you to the garage for repairs.

I. Use Case Diagram

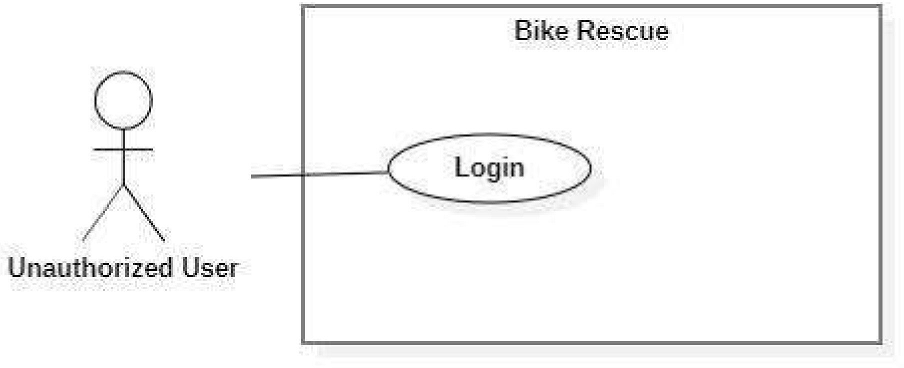


II. Use case detail

<Unauthorized User> Overview User Case



<Unauthorized User> Login



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	Login		
Author:	**HieuNT**		
Date:	**10/6/2020**	**Priority**	Normal

Actor:

- Unauthorized User

Summary:

- This use case allows unauthorized users to login the system.

Goal:

- Unauthorized users log in the system with an authorized role.

Triggers:

- Unauthorized users click on the “Login” button.

Precondition:

- Unauthorized users have an account.

Postcondition:

- Success: Unauthorized users are authorized with a specific role in the system.
- Fail: System shows error messages.

Main Success Scenarios:

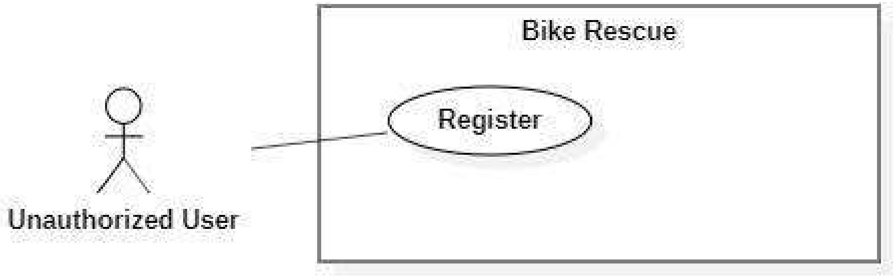
Step	Actor Action	System Response
1	Unauthorized users fill out data fields to the “Login” screen.	“Login” screen is shown with following labels and fields: - Username: text input. - Password: text input.
2	Unauthorized users click on the “Login” button.	System validates inputted data and shows the “Home” page. [Exception 1, 2]

Alternative Flows: N/A
Exceptions:

No	Actor Action	System Response
1	Unauthorized user inputs wrong username or password.	System shows error message: “Username or password wrong. Please try again.”
2	Network problem.	System shows error message: “Fail to login. Please check your connection”.

Relationships: N/A

<Unauthorized User> Register



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	1.0
Use Case Name.	**Register**		
Author	**KienDV**		
Date	10/06/2020	**Priority**	Nomal

Actor:

- Unauthorized user

Summary:

- This use case allows unauthorized users to register.

Goal:

- Unauthorized users signed up successfully.

Triggers:

- Unauthorized users click on the “Register” button.

Preconditions:

- Unauthorized users do not have an account.

Post conditions:

- Success: Unauthorized users will become a user in the system.
- Fail: System shows error messages.

Main Success Scenarios:

Step	**Actor Action**	**System Response**
1	Unauthorized users click on the “Register” button on the app.	“Register” screen is shown with following labels and fields: - Username: text, required. - Password: text, required.

		- Fullname: text, required. - Address: text, optional. - Phone: integer, 8-11 digits, required - Date of Birth: datetime, required
2	Unauthorized users fill out data fields and click on the "Sign Up" button.	System validates inputted data and shows "Login" screen [Exception 1, 2, 3, 4]

Alternative Flows: N/A

Exceptions:

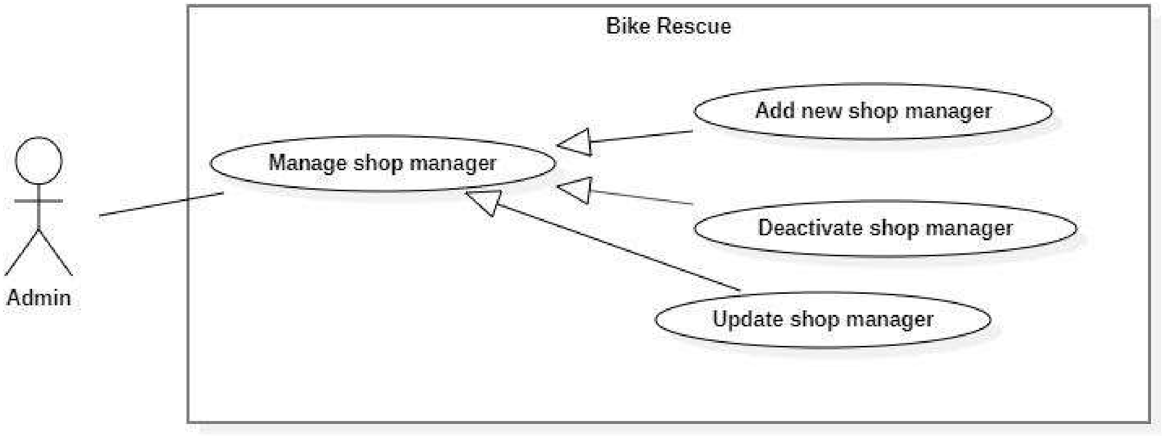
No	Actor Action	System Response
1	- Username is empty. - Phone is empty. - Password is empty. - Full Name is empty.	System shows error message "Please input full field." and highlights the empty fields.
2	Username already existed.	System shows an error message "Username already existed!" and highlights the Username field.
3	Unauthorized users click on the "Cancel" button.	System shows the "Login" screen.
4	Network problem.	System shows an error message: "Please check your connection."

Relationships: N/A

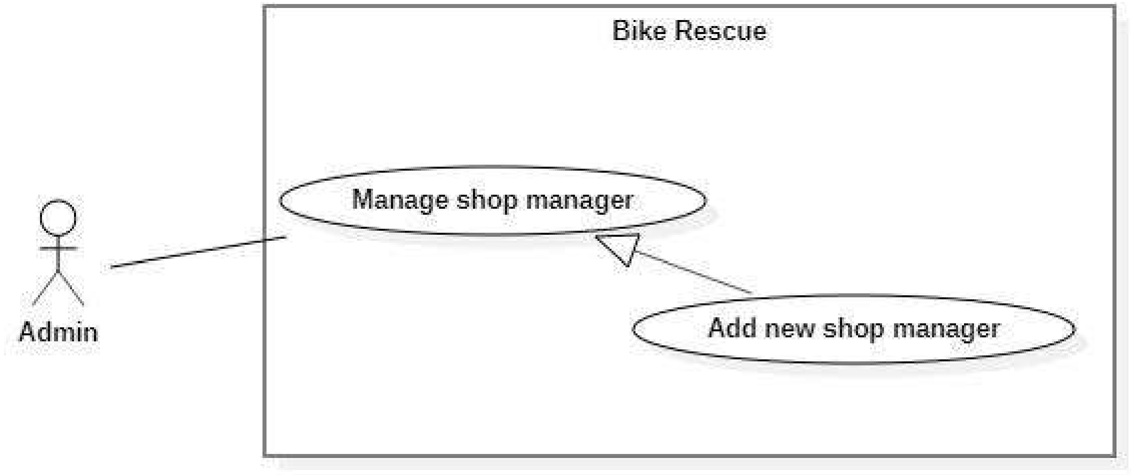
Business Rules:

- Users can log in with given Username and Password

<Admin> Overview User Case



<Admin> Add New Shop Manager



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	1.0
Use Case Name.	**Add new shop manager**		
Author	KienDV		
Date	10/06/2020	**Priority**	Nomal

Actor:

- Admin

Summary:

- This use case allows the admin to add a new shop manager.

Goal:

- Admin add a new shop manager successfully.

Triggers:

- Admin clicks on the “Add Manager” button.

Preconditions:

- Authorized users must login the system with Admin role.

Post conditions:

- Success: A new shop manager will be added.
- Fail: System shows error messages.

Main Success Scenarios:

Step	**Actor Action**	**System Response**

1	Admin clicks on the “Add Manager” button on the “Admin” screen.	“Add New Shop Manager” screen is shown with following labels and fields: - Username: text, required - Password: text, required - Role: select option, list Role - Fullname: text, required. - Address: text, optional. - Phone: integer, 8-11 digits, required.
2	Admin fills out data fields and clicks on the “Confirm” button.	System validates inputted data and shows “Admin” screen [Exception 1, 2, 3, 4]

Alternative Flows: N/A

Exceptions:

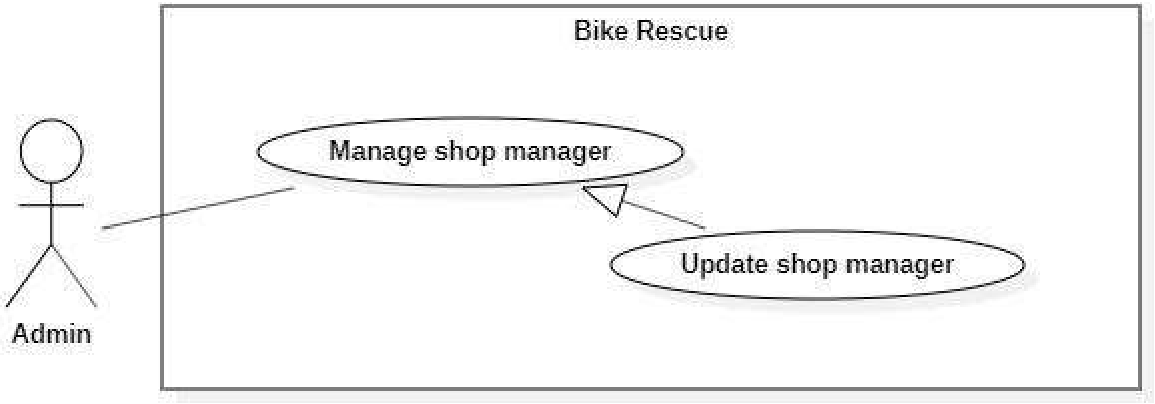
No	Actor Action	System Response
1	- Username is empty. - Password is empty. - Full Name is empty. - Phone is empty.	System shows error message "This field is required.” and highlights the empty fields.
2	Admin clicks on the “Cancel” button.	System shows the “Admin” screen.
3	Network problem.	System shows an error message: “Please check your connection.”
4	Username already existed.	System shows an error message “Username already existed!” and highlights the Username field.

Relationships: N/A

Business Rules:

- In this system, there is no registration for shop managers, only Admin can add a new shop manager.

<Admin> Update Shop Manager



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	Update shop manager		
Author:	**LocNT**		
Date:	**10/6/2020**	**Priority**	Normal

Actor:

- Admin

Summary:

- This use case allows the admin to update information about the shop manager.

Goal:

- Admin updates information of the shop manager successfully.

Triggers:

- Admin clicks on the “Update shop manager” button.

Preconditions:

- Authorized users must login the system with the “Admin” role.

Post conditions:

- Success: Information about the shop manager will be updated.
- Fail: System shows error messages.

Main Success Scenarios:

Step	Actor Action	System Response

1	Admin clicks on the “Update shop manager” button on the “Admin” screen.	“Update shop manager” screen is shown with the following label and field which have current shop manager’s role: - Username: text, required - Password: text, required - Role: select option, list Role - Fullname: text, required. - Address: text, optional. - Phone: integer, 8-11 digits, required.
2	Admin fills out data fields and clicks on the “Confirm” button.	System validates inputted data and shows “Admin” screen [Exception 1, 2, 3, 4]

Alternative Flows: N/A

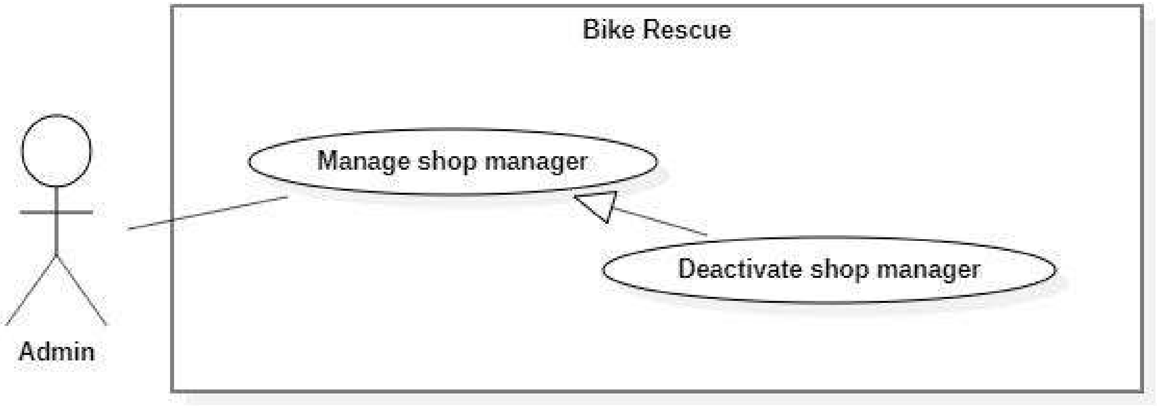
Exceptions:

No	Actor Action	System Response
1	- Username is empty. - Password is empty. - Full Name is empty. - Phone is empty.	System shows error message "This field is required.” and highlights the empty fields.
2	Admin clicks on the “Cancel” button.	System shows the “Admin” screen.
3	Network problem.	System shows an error message: “Please check your connection.”
4	Username already existed.	System shows an error message “Username already existed!” and highlights the Username field.

Relationships: N/A
Business Rules:

- In this system, only the Admin can edit the shop manager’s role.

<Admin> Deactivate Shop Manager



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	Deactivate shop manager		
Author:	**LocNT**		
Date:	**10/6/2020**	**Priority**	Normal
Actor: - Admin **Summary:** - This use case allows the admin to deactivate the shop manager. **Goal:** - Admin deactivates shop manager successfully. **Triggers:** - Admin clicks on the “Deactivate shop manager” button. **Preconditions:** - Authorized must login the system with Admin role. - Shop Manager’s status is currently “Active”. **Post conditions:** - Success: Shop Manager’s status will be changed to “deActive” and cannot login into the system. - Fail: System shows error messages.			

Main Success Scenarios:

Step	Actor Action	System Response
1	Admin clicks on the "Deactivate shop manager" button on the "Admin" screen.	A list of "Active" shop manager will be shown by the system with the "deActive" button right next to every row of the list.
2	Admin clicks on the "deActive" button next to any row of the list.	Confirmation dialog is shown with the Y/N option: "Deactivated user cannot login to system. Are you sure you want to deactivate?"
3	Admin clicks on the "Yes" button.	System shows the "Admin" screen and shows the message "The Shop Manager has been deactivated". [Exception 1]
4	Admin clicks on the "Cancel" button.	System shows the "Admin" screen.

Alternative Flows: N/A

Exceptions:

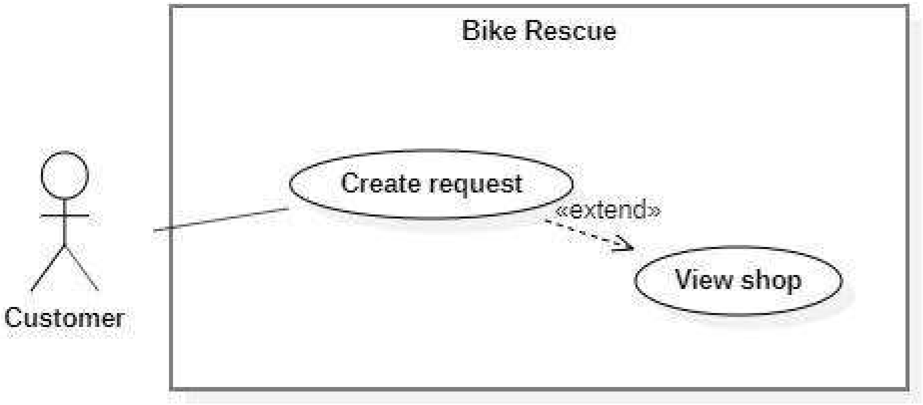
Step	Actor Action	System Response
1	Network problem.	System shows an error message "Please check your connection."
2	Select to deActive a shop manager, which is managing shops.	System shows a warning message "This shop manager is currently managing."

Relationships: N/A

Business Rules:

- In this system, only Admin can deactivate the shop's manager.
- "Inactive" shop managers cannot log in to the system.
- Admin cannot deactivate himself/herself.
- Shops which were deactivated won't be removed from the database.
- Only "Active" shop managers can be deactivated.

<Customer> View List Shop



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	View list shop		
Author:	**LinhCN**		
Date:	**10/6/2020**	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customers to view a list of shops sorted based on distance from the customer's location.

Goal:

- The system shows a list of shops successfully.

Triggers:

- Customer clicks on the "View shop" button.

Preconditions:

- Authorized users must login the system with Customer role.
- Authorized users have to turn on location.

Post conditions:

- Success: A list of surrounding stores will be displayed
- Fail: System shows error messages.

Main Success Scenarios:

Step	**Actor Action**	**System Response**

1	Customer clicks on the “View shop” button on the “Customer” screen.	Confirmation dialog is shown: “Make sure you have location enabled”[Exception 1]
2	Customer clicks on the “OK” button.	The system sends a request to the server containing the location of the customer. Based on the customer's location, the system uses a sort algorithm to respond to the customer's screen. [Exception 2]

Alternative Flows: N/A

Exceptions:

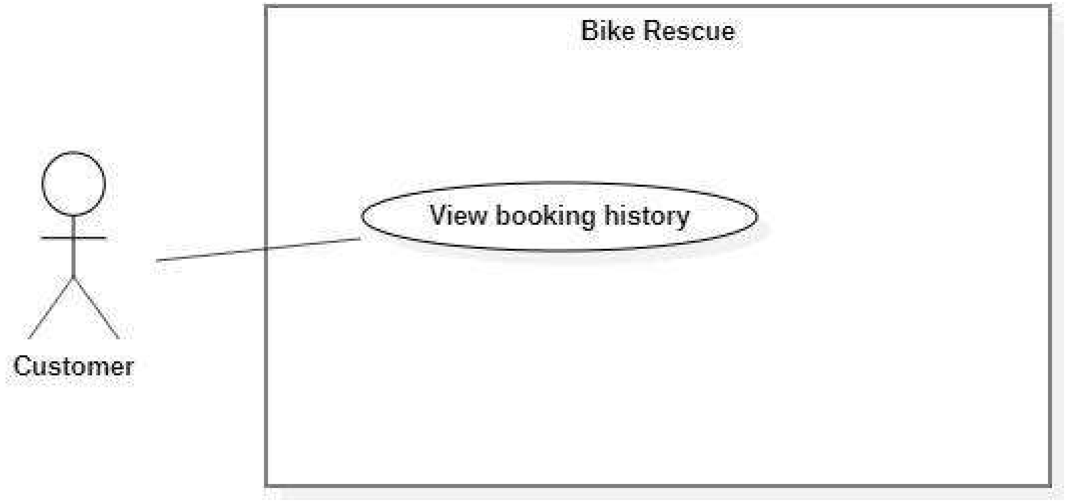
Step	Actor Action	System Response
1	Customer turn off location.	The system requires the user turning on the location in the customer's device.
2	Network problem.	System shows an error message “Please check your connection.”

Relationships: N/A

Business rule:

- In this system, only Customer can view the list shop to book.

<Customer> View Booking History



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	View Booking History		
Author:	**HieuNT**		
Date:	**06/07/2020**	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customers to view a list of successfully booked cases that the current account has booked before.

Goal:

- The system shows a list of booked cases.

Triggers:

- Customer clicks on the “Booking History” button.

Preconditions:

- Authorized users must login the system with Customer role.

Post conditions:

- Success: A list of booked cases will be shown.
- Fail: System shows error messages.

Main Success Scenarios:

Step	Actor Action	System Response
1	Customer clicks on the “Booking History” button on the “Customer” screen.	The system will send the request to the server and query from the database to get all of the successfully booked cases and display a list on the customer’s device screen [Exception 1, 2]

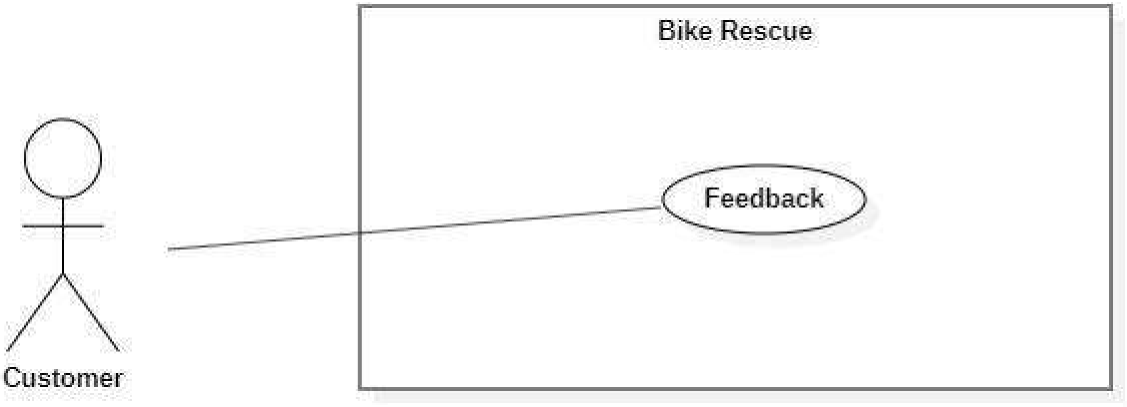
Alternative Flows: N/A

Exceptions:

Step	Actor Action	System Response
1	Customers have not booked any successful case yet	The system shows a dialog to notify customers that they haven't booked any yet.
2	Network problem.	System shows an error message “Please check your connection.”

Relationships: N/A

<Customer> Feedback



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	Feedback		
Author:	**HieuNT**		
Date:	**06/07/2020**	**Priority**	Normal

Actor:

- Customer

Summary:

- This use case allows customers to give feedback on the shops.

Goal:

- The system can receive the feedback successfully with integrity.

Triggers:

- Customer clicks on the “Give Feedback” button after a successful fixed case.

Preconditions:

- Authorized users must login the system with the “Customer” role.
- Authorized users have done with the booking case.

Post conditions:

- Success: The system shows the message that notice customer the feedback is received
- Fail: System shows error messages.

Main Success Scenarios:

Step	Actor Action	System Response

1	Customers click on the “Give Feedback” button on screen after the fixing has been done.	The feedback board includes stars for rating and a textarea to give feedback will be shown.
2	Customer clicks on the “Send” button.	The system will save the feedback, send it to the server to store the feedback to the database, then show the “Thank you” notification on the customer’s screen. [Exception 1, 2]

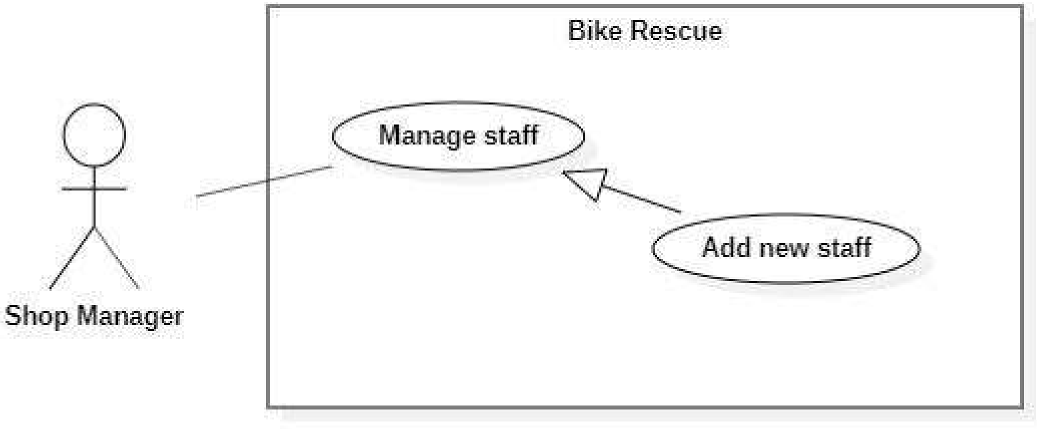
Alternative Flows: N/A

Exceptions:

Step	Actor Action	System Response
1	Customer do not rate or write anything	The system will see this case as no feedback.
2	Network problem.	System shows an error message “Please check your connection.”

Relationships: N/A

<Shop Manager> Add New Staff



USE_CASE – BIKE RESCUE			
Use Case No.		**Use Case Version**	**1.0**
Use Case Name:	Add new staff		
Author:	**LinhCN**		
Date:	**06/07/2020**	**Priority**	Normal

Actor:

- Shop Manager

Summary:

- This use case allows the Shop Manager to add new staff on the shops.

Goal:

- The system can add new staff successfully with integrity.

Triggers:

- Shop manager clicks on the “Add new staff” button.

Preconditions:

- Authorized users must login the system with the “Shop manager” role.

Post conditions:

- Success: The system shows the message that new staff added successfully
- Fail: System shows error messages.

Main Success Scenarios:

Step	Actor Action	System Response

1	Shop manager clicks on “Add new staff” on the screen.	“Add new staff” screen is shown with following label: - Username: text, required - Password: text, required - Role: select option, list Role - Fullname: text, required. - Address: text, optional. - Date of birth: date, required. - Phone: integer, 8-11 digits, required. - Note: text, optional.
2	Shop manager fills out data fields and clicks on the “Confirm” button.	System validates inputted data and shows “Manage staff” screen [Exception 1, 2, 3]

Alternative Flows: N/A

Exceptions:

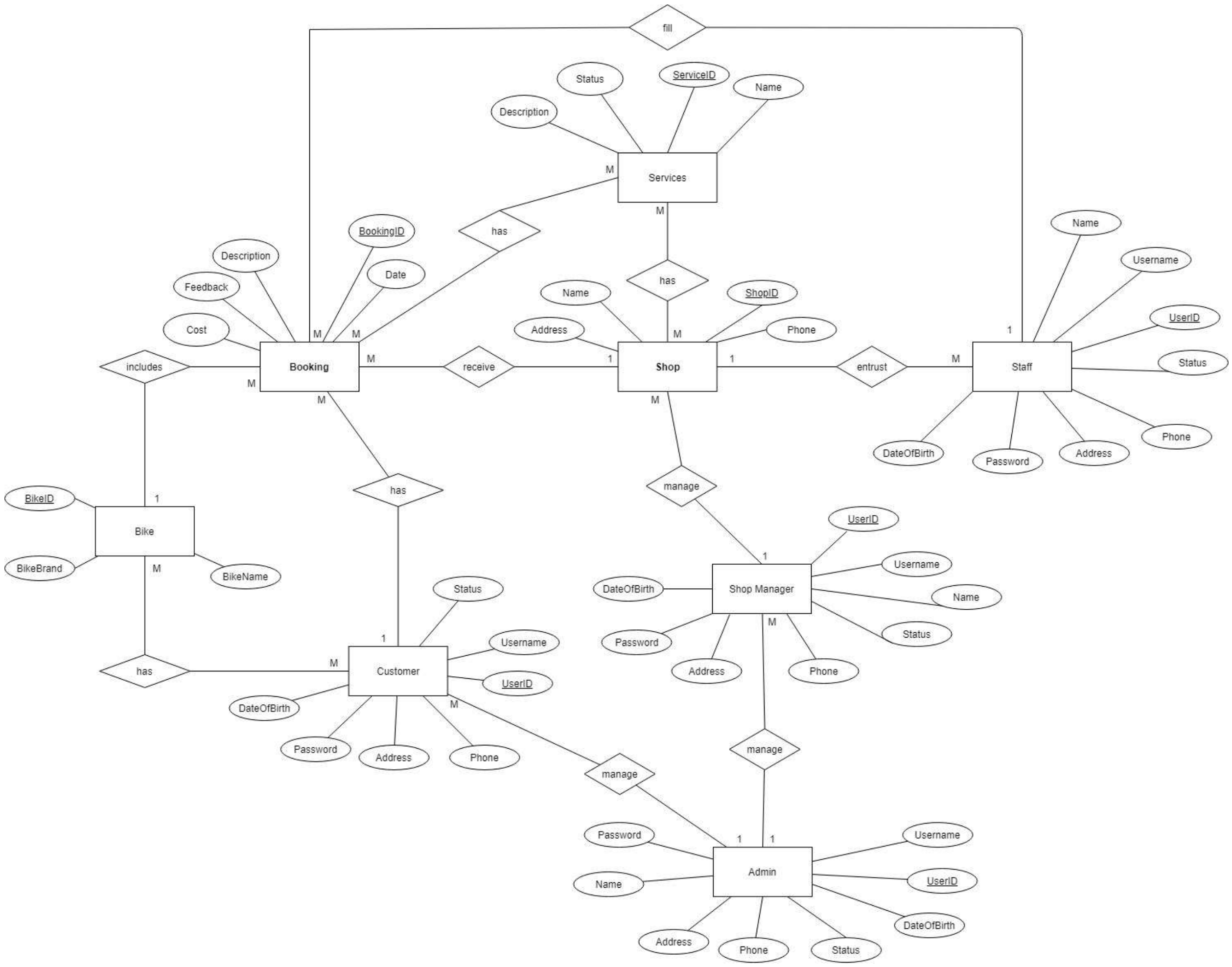
No	Actor Action	System Response
1	- Username is empty. - Password is empty. - Full Name is empty. - Date of birth is not selected. - Phone is empty.	System shows error message "Please input full field.” and highlights the empty fields.
2	Username already exists.	System show error message “Username already exists!” and highlight the error field.
3	Network problem.	System shows an error message: “Please check your connection.”

Relationships: N/A

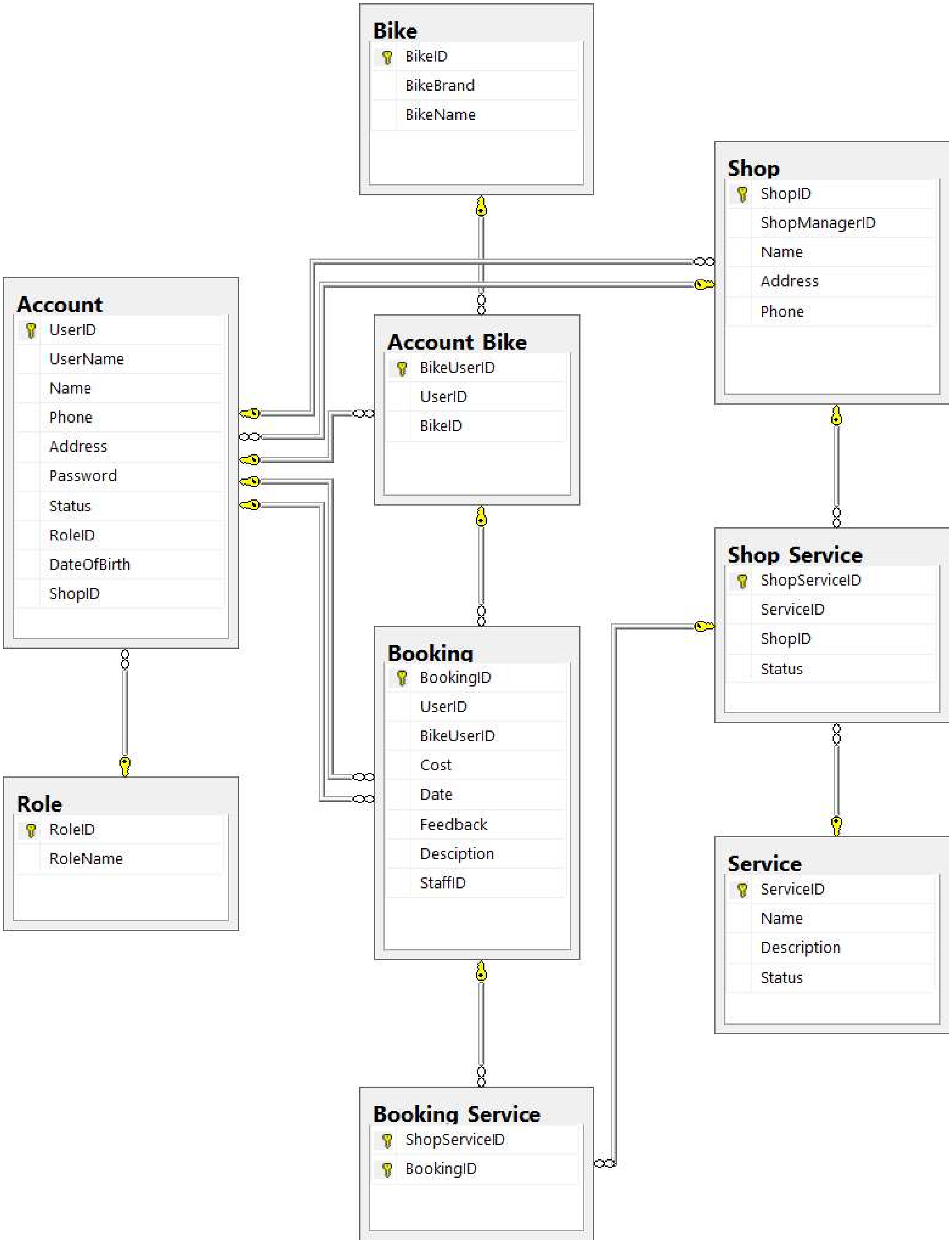
Business Rules:

- In this system, only the Shop manager can add new staff.

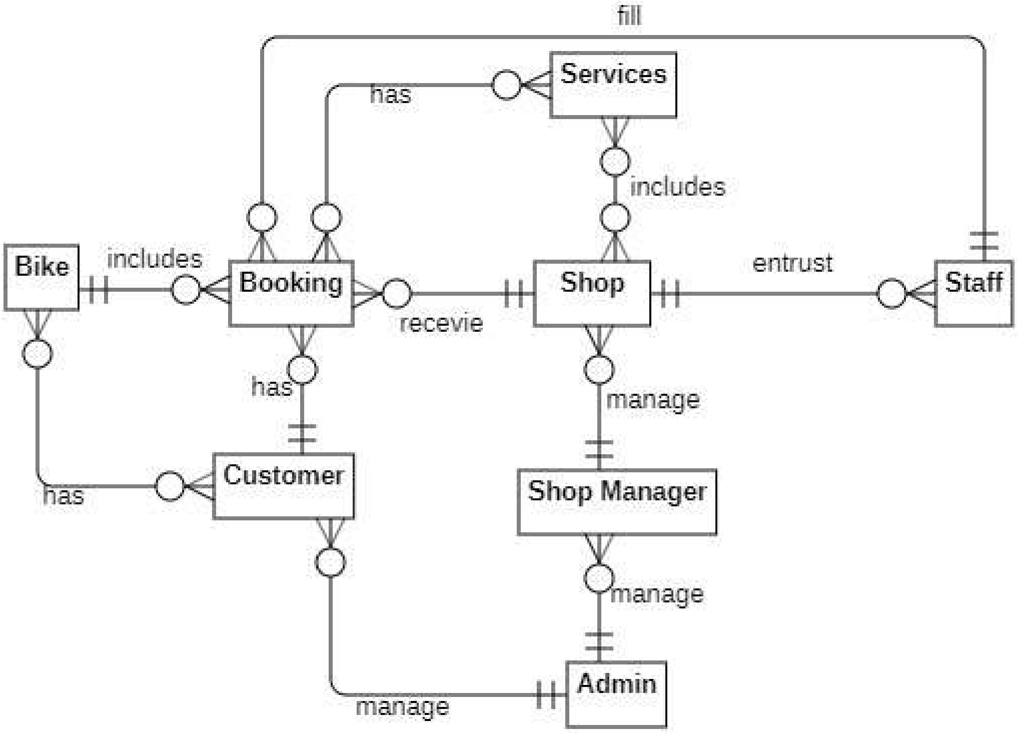
III. ERD


fill
Status
ServiceID
Name
Description
Services
M
M
has
BookingID
Date
Description
Feedback
Cost
M
M
Booking
M
M
M
has
Name
Address
ShopID
Phone
M
receive
1
Shop
1
entrust
M
Staff
1
Name
Username
UserID
Status
Phone
Address
Password
DateOfBirth
includes
has
M
manage
1
BikeID
Bike
BikeBrand
BikeName
M
UserID
Username
DateOfBirth
Shop Manager
Name
Status
Password
Address
Phone
M
Status
1
Username
has
M
Customer
UserID
DateOfBirth
Password
Address
Phone
M
manage
manage
Password
1
1
Username
Name
Admin
UserID
DateOfBirth
Address
Phone
Status

IV. Database Diagram

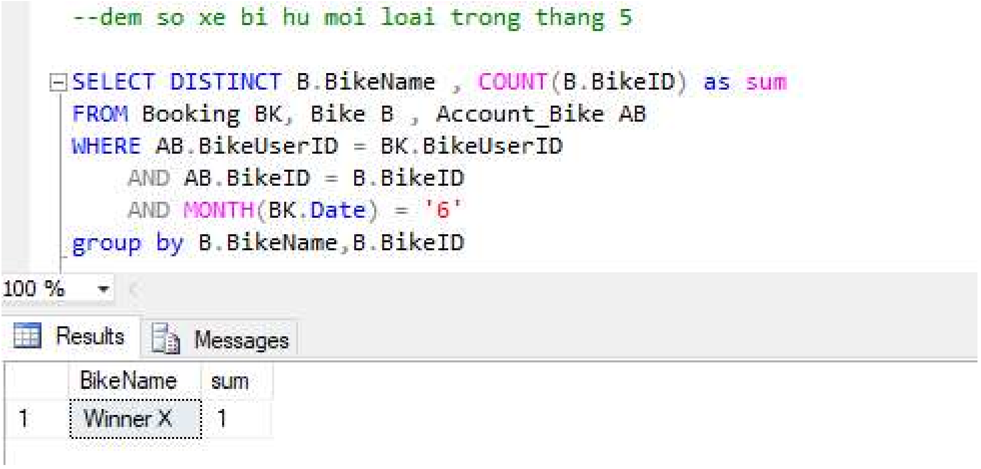


V. Conceptual Diagram



VI. Query Database

- Query 1:

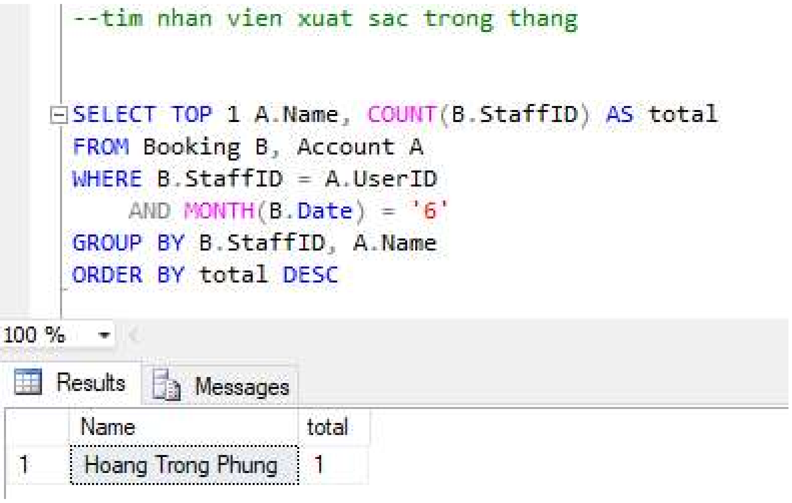


```
--dem so xe bi hu moi loai trong thang 5

SELECT DISTINCT B.BikeName , COUNT(B.BikeID) as sum
FROM Booking BK, Bike B , Account_Bike AB
WHERE AB.BikeUserID = BK.BikeUserID
    AND AB.BikeID = B.BikeID
    AND MONTH(BK.Date) = '6'
group by B.BikeName,B.BikeID
```

	BikeName	sum
1	Winner X	1

- Query 2:

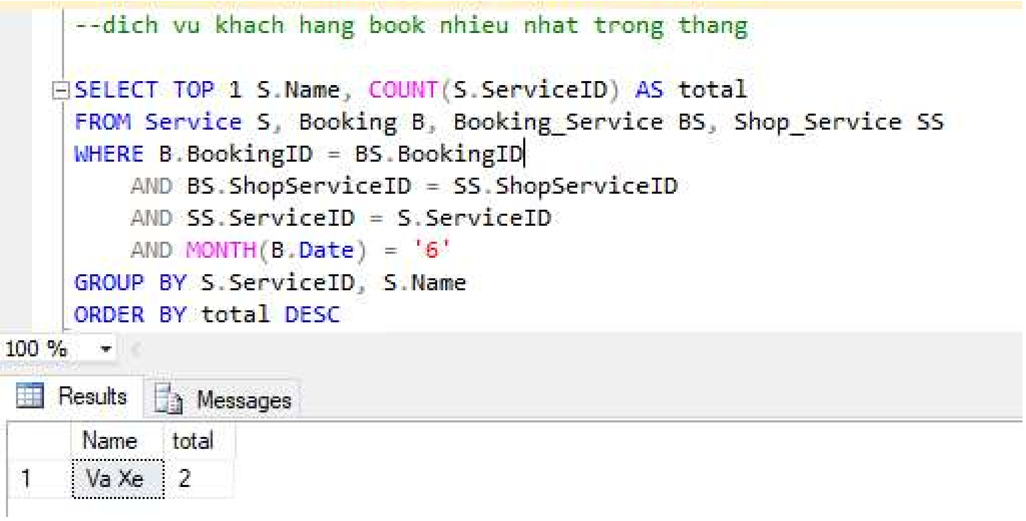


```
--tim nhan vien xuat sac trong thang


SELECT TOP 1 A.Name, COUNT(B.StaffID) AS total
FROM Booking B, Account A
WHERE B.StaffID = A.UserID
    AND MONTH(B.Date) = '6'
GROUP BY B.StaffID, A.Name
ORDER BY total DESC
```

	Name	total
1	Hoang Trong Phung	1

- Query 3:

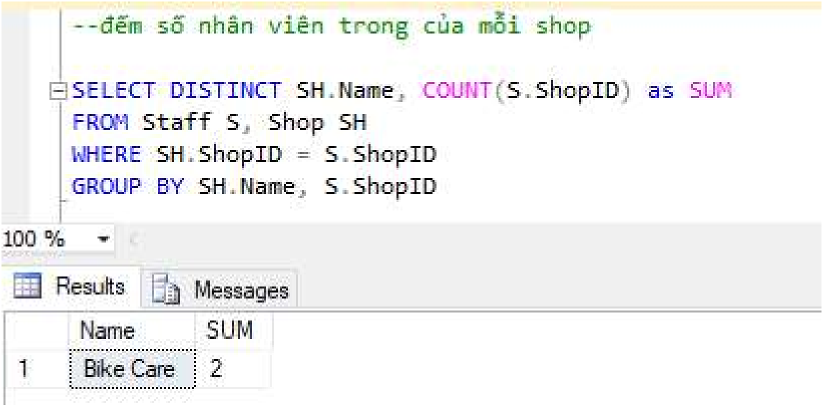


```
--dich vu khach hang book nhieu nhat trong thang

SELECT TOP 1 S.Name, COUNT(S.ServiceID) AS total
FROM Service S, Booking B, Booking_Service BS, Shop_Service SS
WHERE B.BookingID = BS.BookingID
    AND BS.ShopServiceID = SS.ShopServiceID
    AND SS.ServiceID = S.ServiceID
    AND MONTH(B.Date) = '6'
GROUP BY S.ServiceID, S.Name
ORDER BY total DESC
```

	Name	total
1	Va Xe	2

- Query 4:

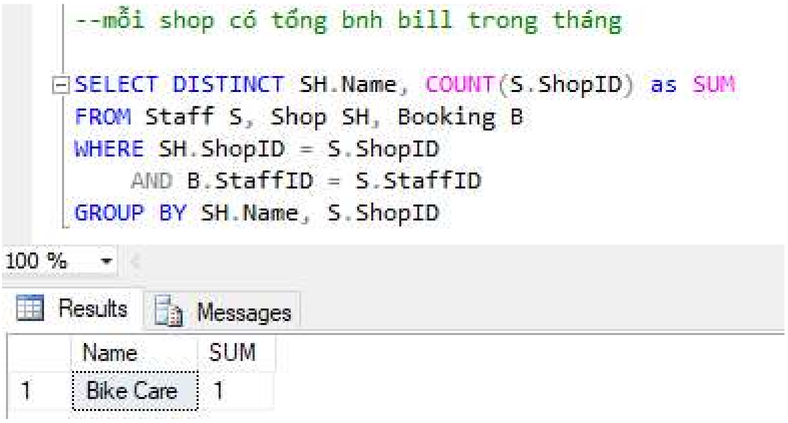


```
--đếm số nhân viên trong của mỗi shop

SELECT DISTINCT SH.Name, COUNT(S.ShopID) as SUM
FROM Staff S, Shop SH
WHERE SH.ShopID = S.ShopID
GROUP BY SH.Name, S.ShopID
```

	Name	SUM
1	Bike Care	2

- Query 5:



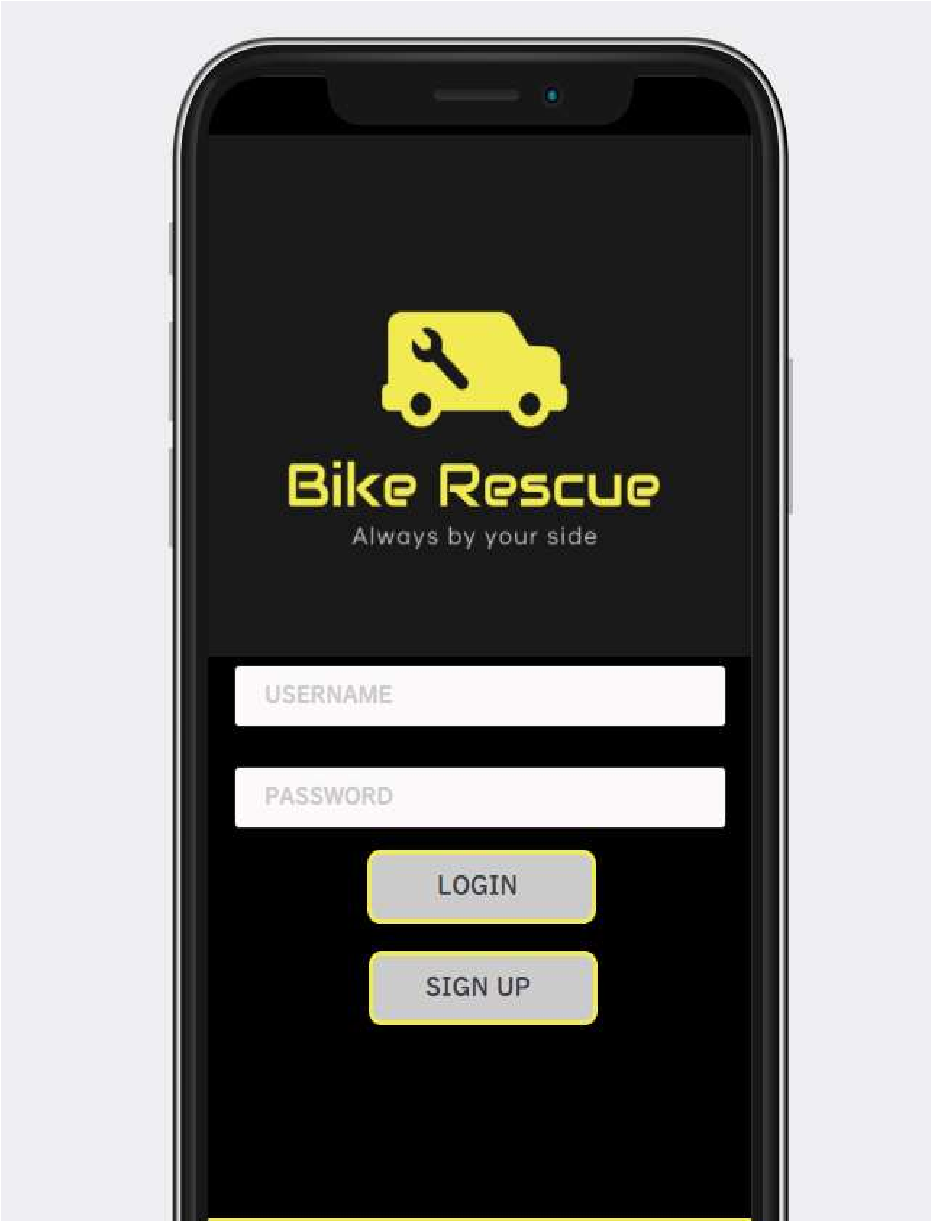
```
--mỗi shop có tổng bnh bill trong tháng

SELECT DISTINCT SH.Name, COUNT(S.ShopID) as SUM
FROM Staff S, Shop SH, Booking B
WHERE SH.ShopID = S.ShopID
    AND B.StaffID = S.StaffID
GROUP BY SH.Name, S.ShopID
```

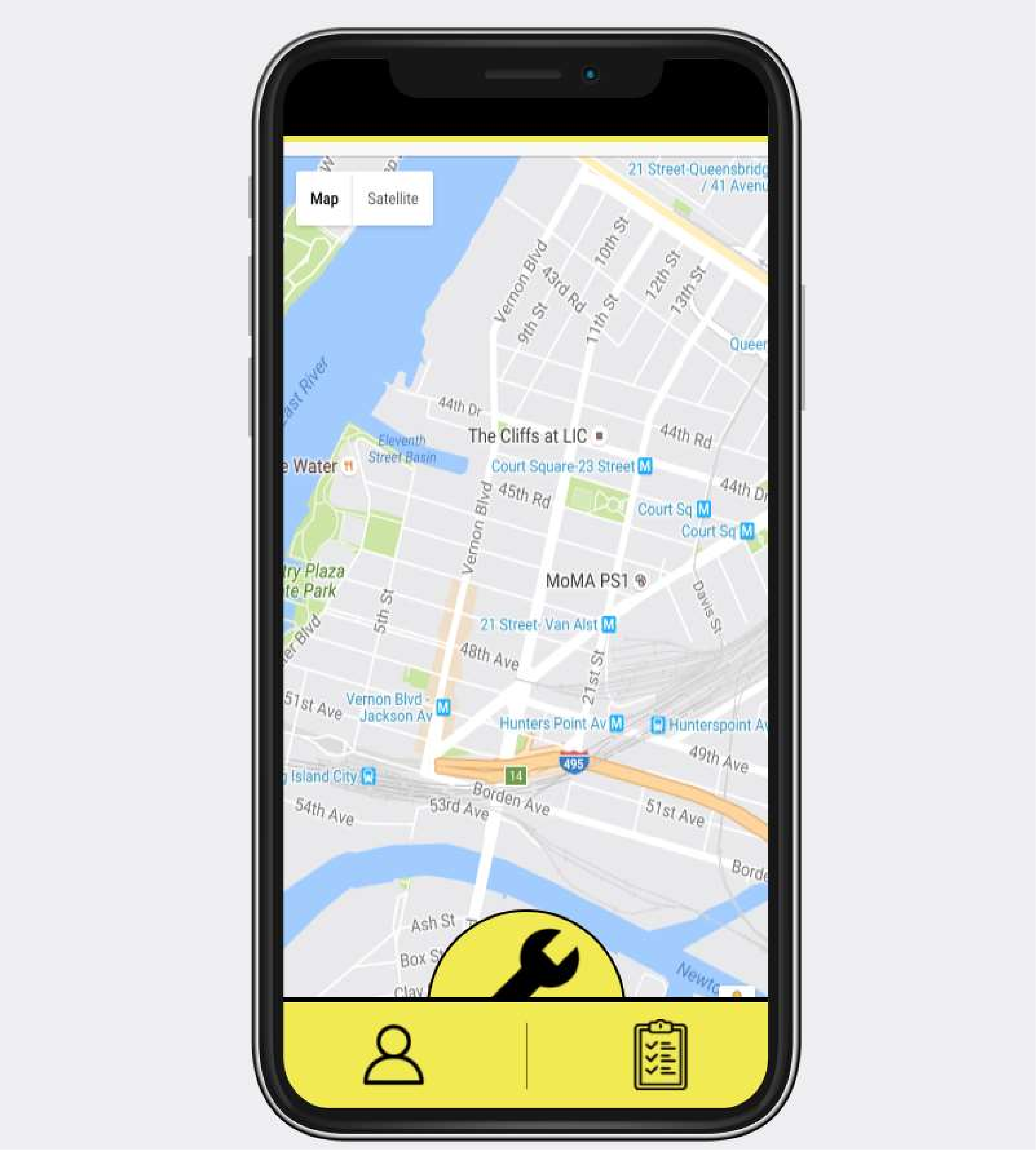
```
   Name       SUM
1  Bike Care  1
```

VII. User Interface (demo)

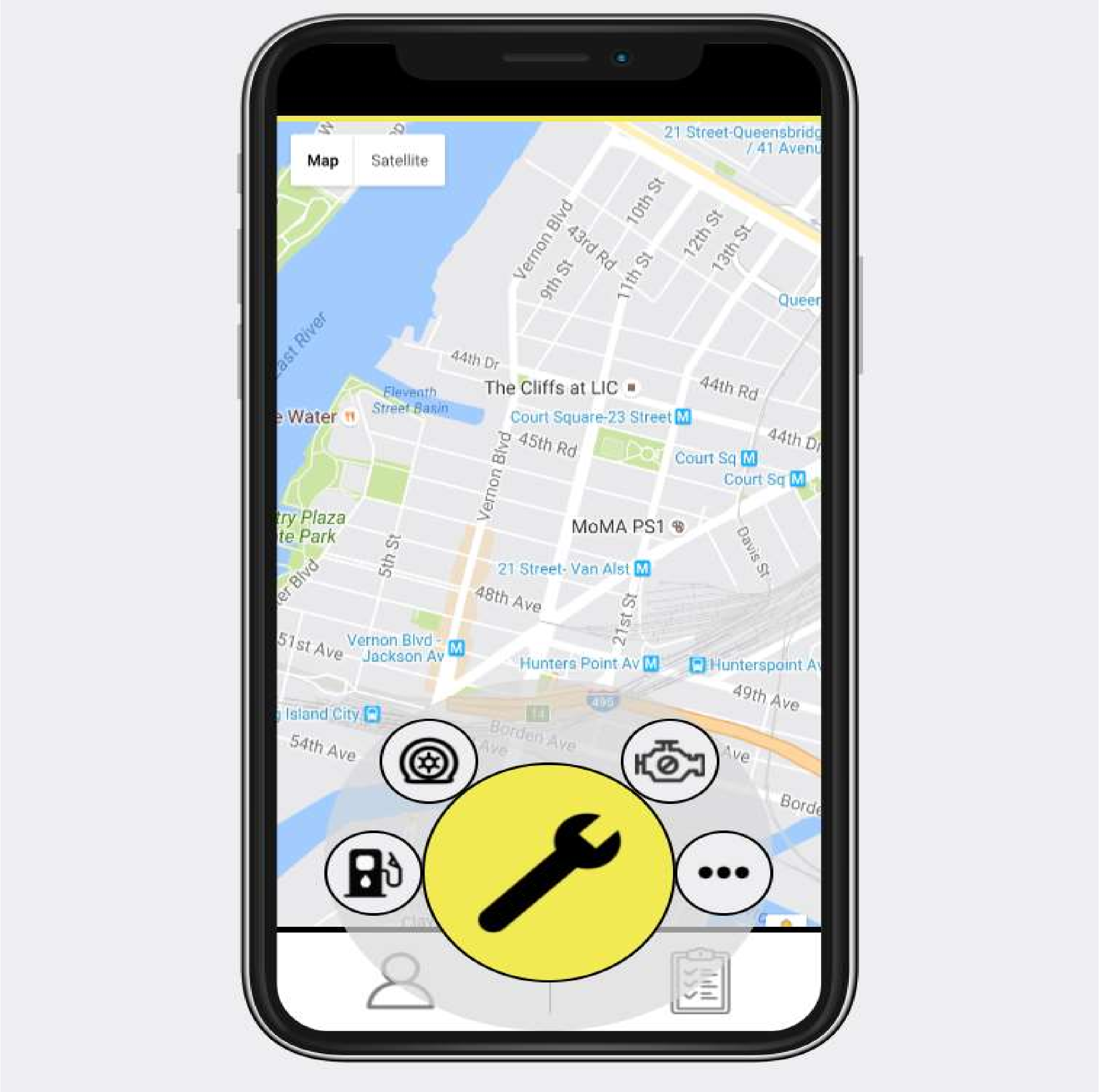
- UI Login



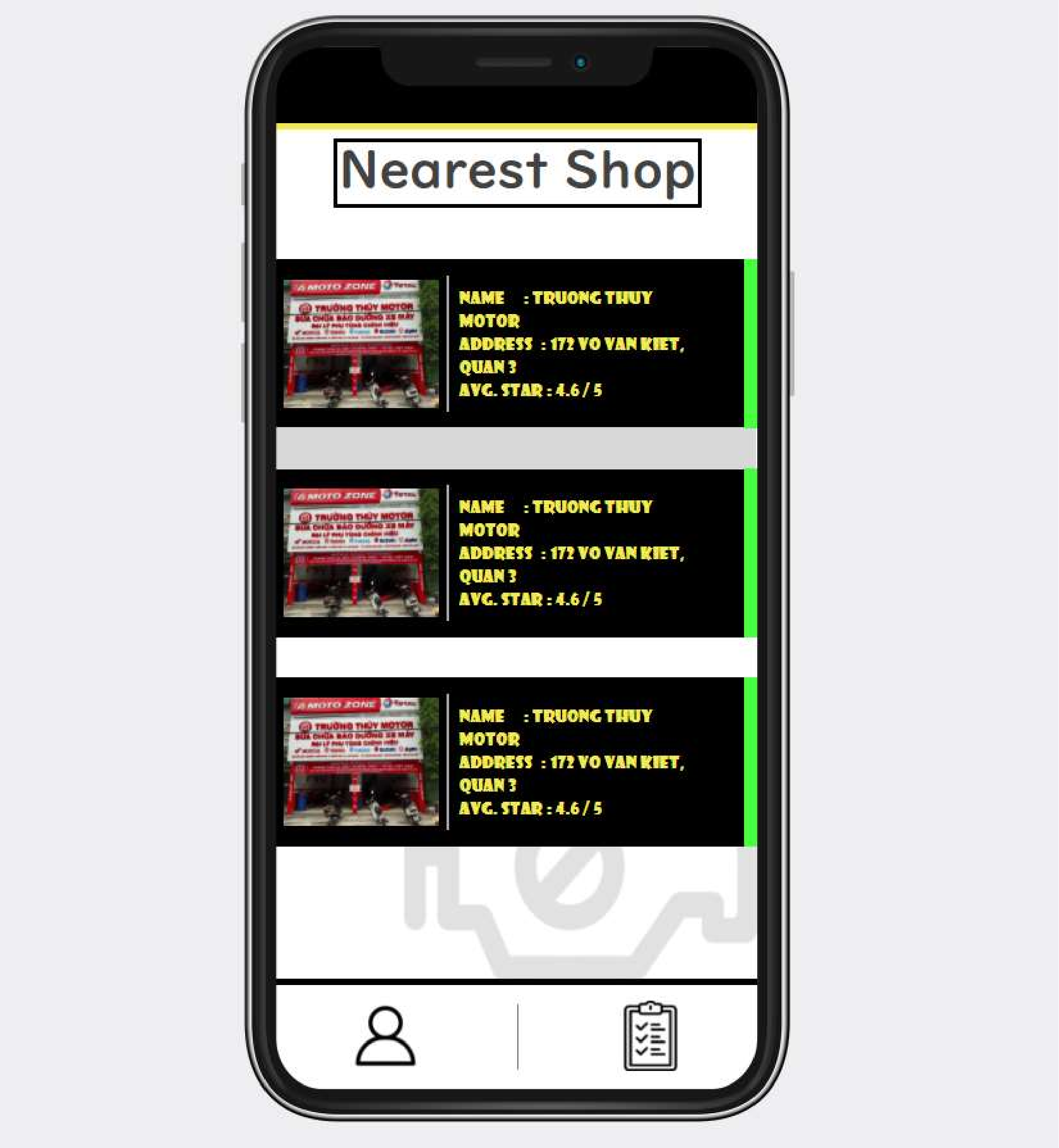
- UI screen when the user logged in successfully.



- Fixing button demonstration



- UI for selecting shops



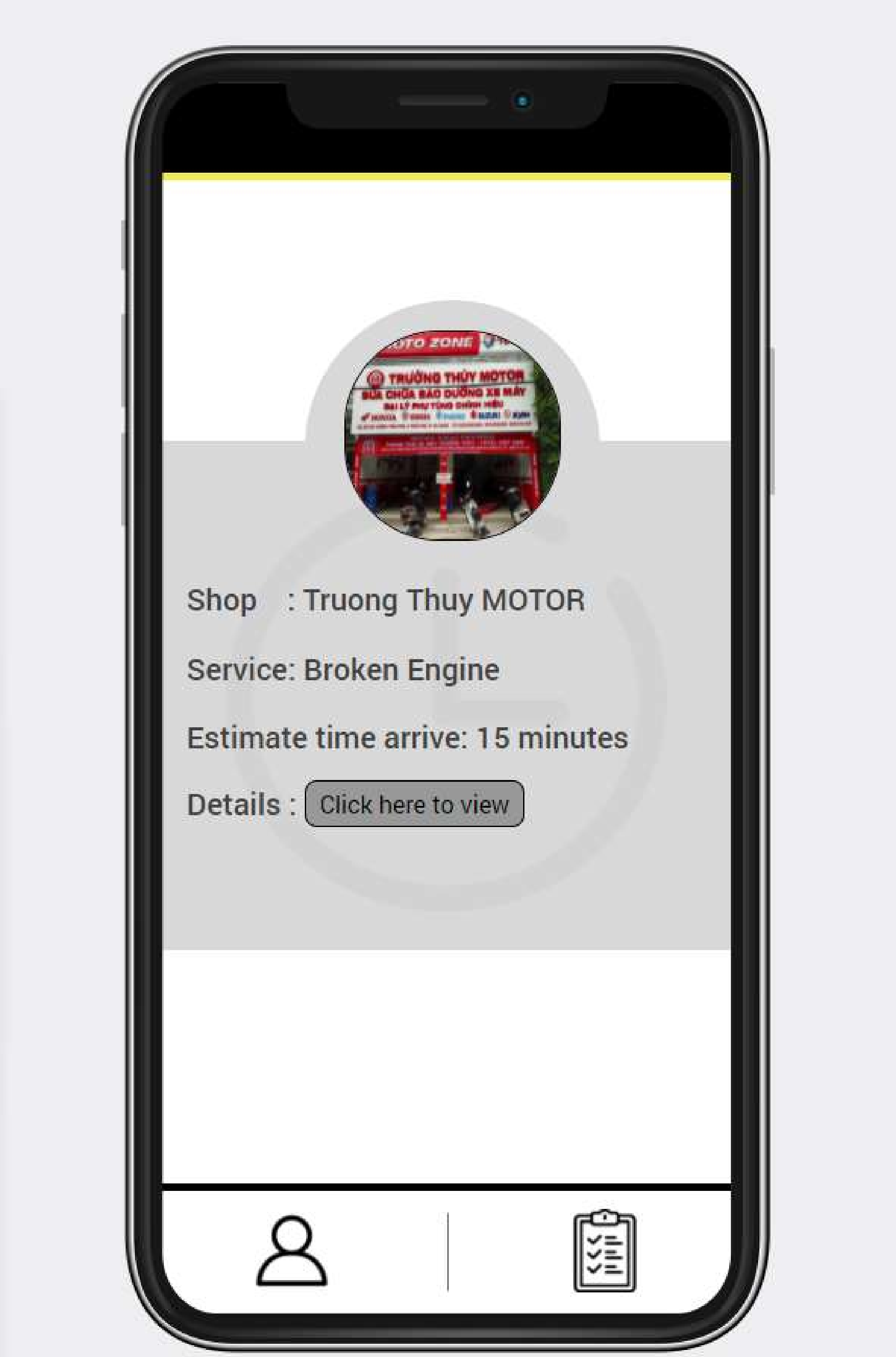
Nearest Shop

NAME : TRUONG THUY MOTOR
ADDRESS : 172 VO VAN KIET, QUAN 3
AVG. STAR : 4.6 / 5

NAME : TRUONG THUY MOTOR
ADDRESS : 172 VO VAN KIET, QUAN 3
AVG. STAR : 4.6 / 5

NAME : TRUONG THUY MOTOR
ADDRESS : 172 VO VAN KIET, QUAN 3
AVG. STAR : 4.6 / 5

-UI after selected shop



Shop : Truong Thuy MOTOR

Service: Broken Engine

Estimate time arrive: 15 minutes

Details :