ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

**CNG 495 – Cloud Computing**

**Term Project Proposal**

**Hasan Eren Yarar 2457802**

**Ata Kaleli 2385474**

**Serhan Turgul 2453603**

**STAGE 1 REPORT**

**Repair-Shop Car Management System:**

**Overview:**

This system is an all-in-one solution designed to simplify the management of vehicle maintenance and repair tasks for both vehicle owners and repair shop owners. It acts as a strong platform that links vehicle owners, service providers, and administrative staff. The system is specially made to improve the process of organizing, managing, and monitoring vehicle repair and maintenance work.

**Key Features and Functionalities*:***

User (Vehicle Owners):

**Registration & Login:** Users are securely register and log in to the system, ensuring authenticated access to the  their vehicle data.

**Maintenance Record View:** Users have access to their past maintenance information.

**User Vehicle Management:** Users can add new vehicles to their profile, including details like model, miles, type, and year.

**Appointment Scheduling:** Users can easily schedule appointments with repair shops, making it more convenient to manage repair and maintenance activities for their vehicles.

Repair Shop Management:

**Access to User Information**: Repair shops can view information about their customers, including contact details and vehicle data.

**Take Maintenance Record:** Shops can add new maintenance records for the services they provide, with detail tracking of all maintenance activities.

**Appointment Management**: Viewing and managing upcoming appointments.

Administrative Functions:

**Repair Shop Statistics**: Administrators can view  total income for each Repair Shop.

**Repair Shop Registration**: New repair shops can be added to the system.

**Technologies:**

- Frontend: Python Tkinter
- Backend Python 3.9
- Database: SQLITE3
- Cloud Services: Google Cloud Platform

**Google Cloud Platform as an Infastructure as a Service (IaaS):**

We are utilising Google Cloud Platform as an Infastructure as a Service (IaaS), by using their Compute Engine to run a Virtual Machine(VM). We run Ubuntu 20.04 on the VM, to run our server script and database.

What we are doing is a Software as a Service (SaaS). We have a server script that runs on Google Cloud Platform which listens to incoming details from each client, executes queires depending on what is provided by the client, and then sends the results of the query back to the client. We consider this as SaaS because the client is a piece of software that communicates with the cloud. Any number of repair shops can utilize this software to make their business with their clients easier.

**Data Types**

**Text Data:**

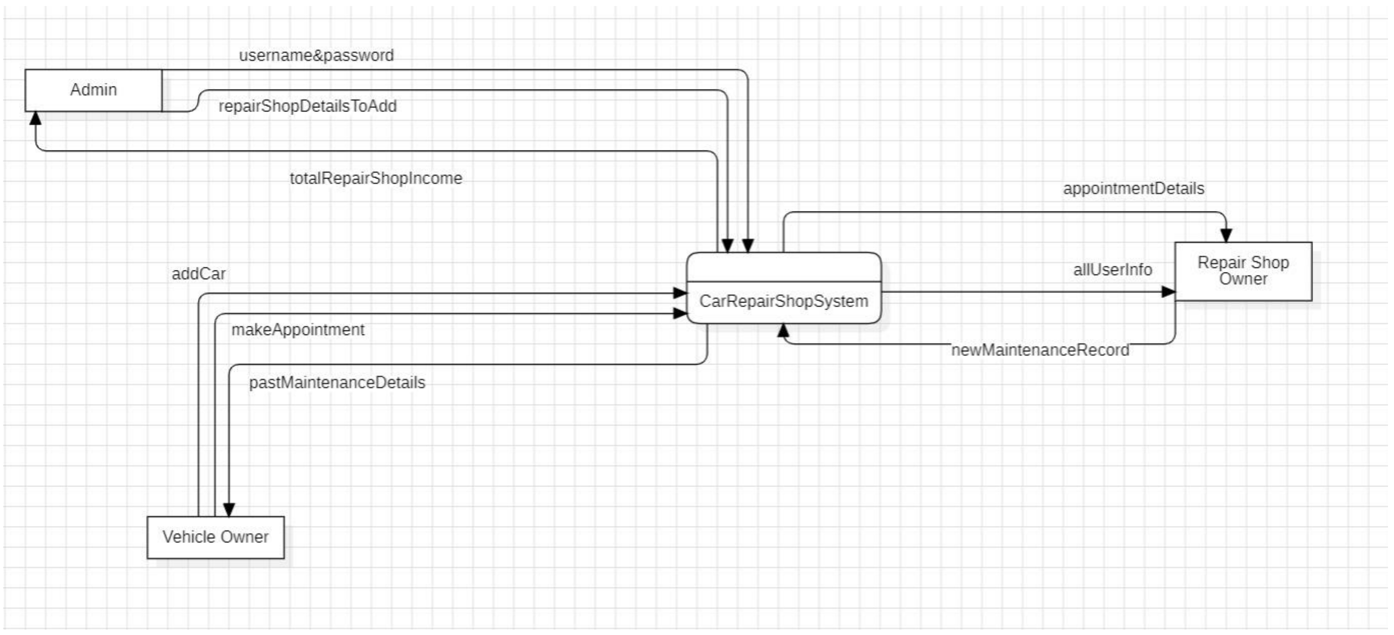
- Maintanance descriptions and details.
- User information such as email name address
- Admin inormation such as email pasword name

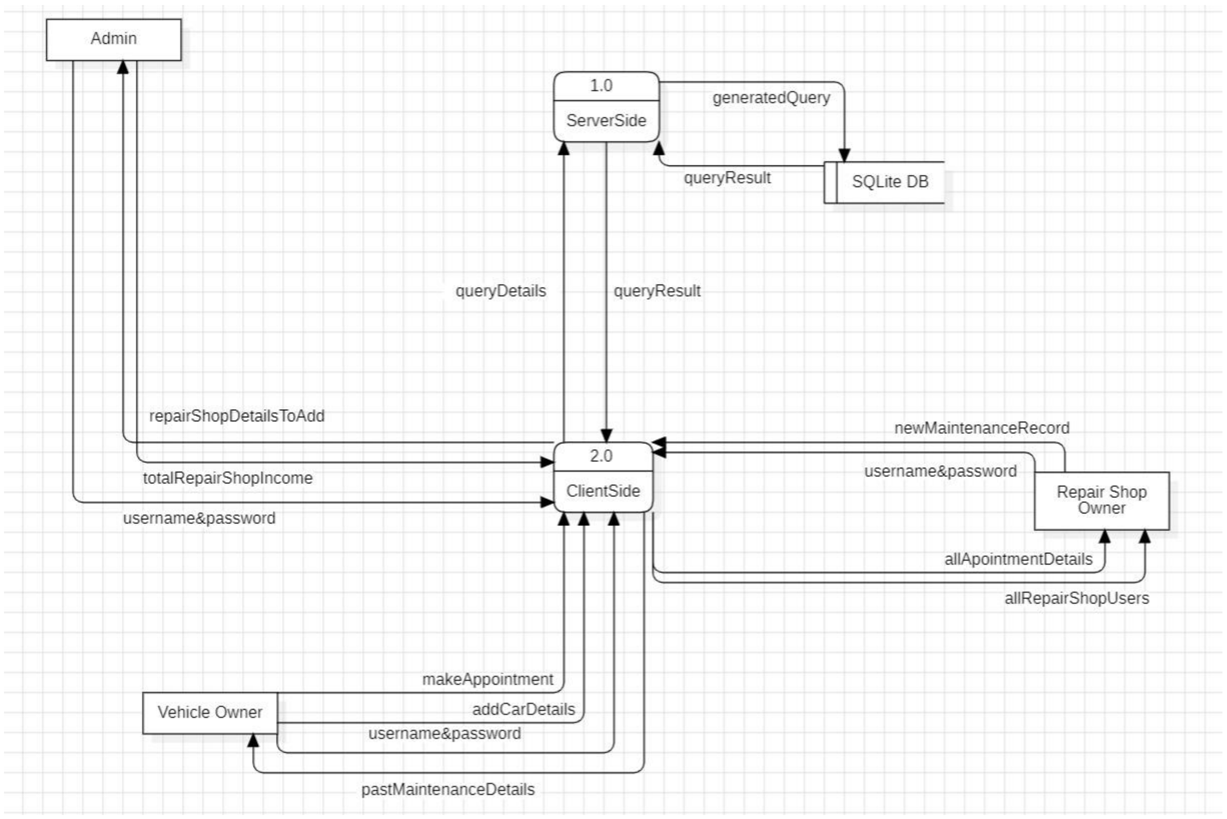
- Repair Shop information such as email password

**Numerical Data:**

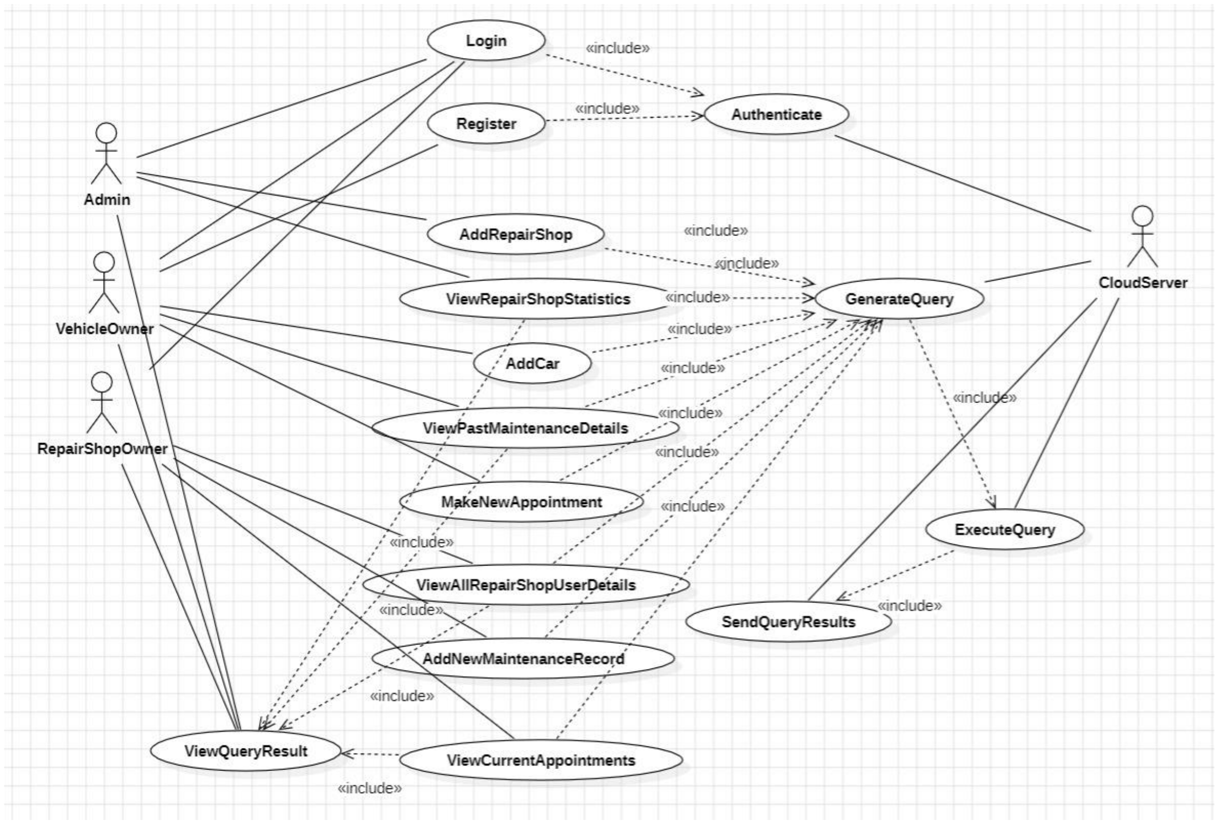
- Miles of car
- Price of the maintanance

**Data Flow Diagram of The System:**





## Client & Service Interactions Diagram:



• ***State the expected contribution for each project group member:***

All team members contributed to both backend and frontend coding. Specifically, Hasan Eren Yarar focused on designing and coding the database, Ata Kaleli handled the graphical user interface and server connection, and Serhan Turgul specialized in integrating the program with Google Cloud Services.

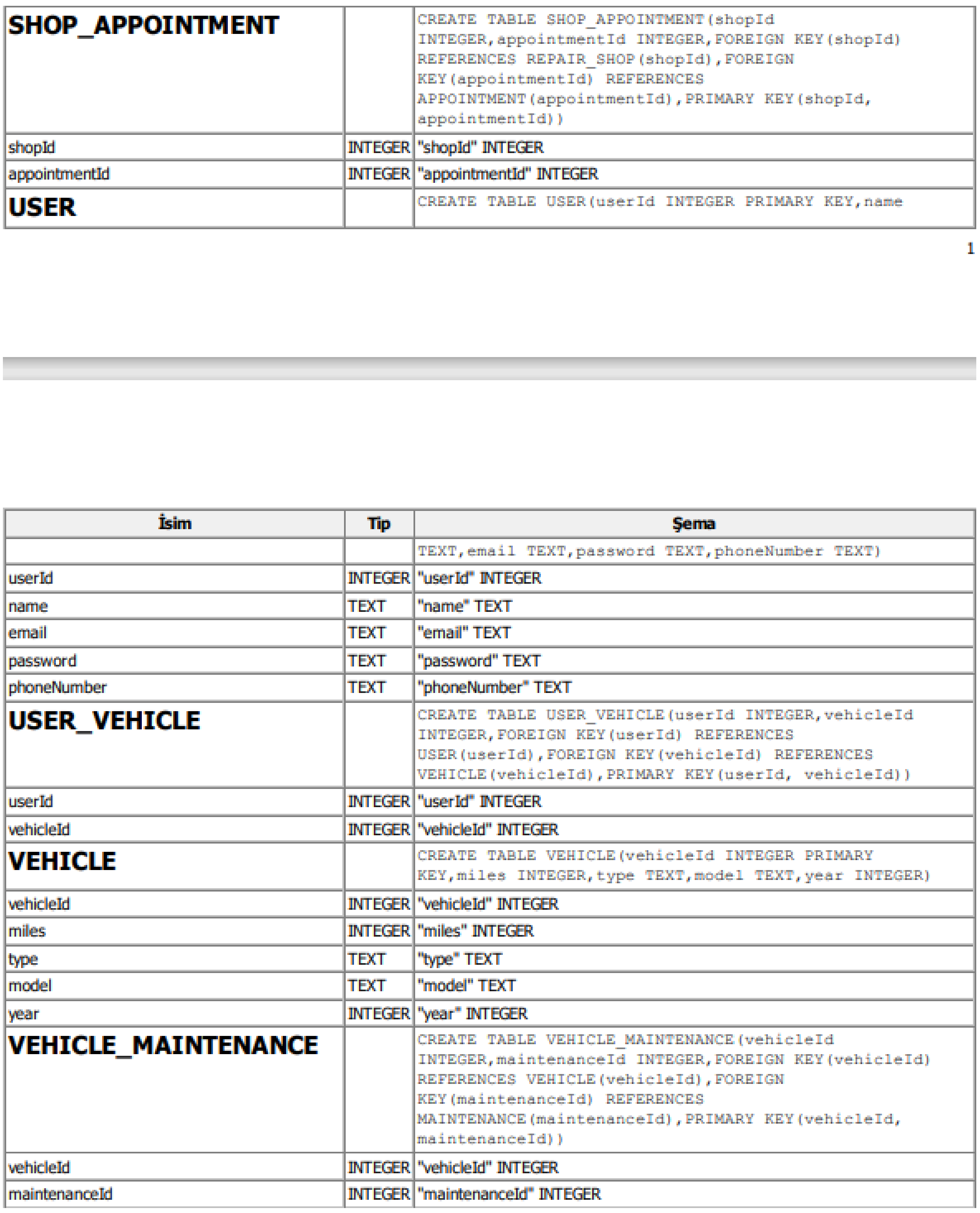
## STAGE 2 REPORT

### PART 1 & 2

**First of all, Eren dealed with database of our system with sqlite3 in pyhton. Below are the screenshot of our system's database.**

## Tablolar (11)

| İsim | Tip | Şema |
|---|---|---|
| **ADMIN** | | `CREATE TABLE ADMIN(adminId INTEGER PRIMARY KEY,name TEXT,email TEXT,password TEXT)` |
| adminId | INTEGER | "adminId" INTEGER |
| name | TEXT | "name" TEXT |
| email | TEXT | "email" TEXT |
| password | TEXT | "password" TEXT |
| **APPOINTMENT** | | `CREATE TABLE APPOINTMENT(appointmentId INTEGER PRIMARY KEY,date TEXT,userId INTEGER,vehicleId INTEGER,shopId INTEGER,FOREIGN KEY(userId) REFERENCES USER(userId),FOREIGN KEY(vehicleId) REFERENCES VEHICLE(vehicleId),FOREIGN KEY(shopId) REFERENCES REPAIR_SHOP(shopId))` |
| appointmentId | INTEGER | "appointmentId" INTEGER |
| date | TEXT | "date" TEXT |
| userId | INTEGER | "userId" INTEGER |
| vehicleId | INTEGER | "vehicleId" INTEGER |
| shopId | INTEGER | "shopId" INTEGER |
| **APPOINTMENT_SCHEDULE** | | `CREATE TABLE APPOINTMENT_SCHEDULE(appointmentId INTEGER,scheduleId INTEGER,FOREIGN KEY(appointmentId) REFERENCES APPOINTMENT(appointmentId),FOREIGN KEY(scheduleId) REFERENCES SCHEDULE(scheduleId),PRIMARY KEY(appointmentId, scheduleId))` |
| appointmentId | INTEGER | "appointmentId" INTEGER |
| scheduleId | INTEGER | "scheduleId" INTEGER |
| **MAINTENANCE** | | `CREATE TABLE MAINTENANCE(maintenanceId INTEGER PRIMARY KEY,cost INTEGER,name TEXT,date TEXT,description TEXT,shopId INTEGER,FOREIGN KEY(shopId) REFERENCES REPAIR_SHOP(shopId))` |
| maintenanceId | INTEGER | "maintenanceId" INTEGER |
| cost | INTEGER | "cost" INTEGER |
| name | TEXT | "name" TEXT |
| date | TEXT | "date" TEXT |
| description | TEXT | "description" TEXT |
| shopId | INTEGER | "shopId" INTEGER |
| **REPAIR_SHOP** | | `CREATE TABLE REPAIR_SHOP(shopId INTEGER PRIMARY KEY,email TEXT,phoneNumber TEXT,address TEXT,status TEXT,password TEXT)` |
| shopId | INTEGER | "shopId" INTEGER |
| email | TEXT | "email" TEXT |
| phoneNumber | TEXT | "phoneNumber" TEXT |
| address | TEXT | "address" TEXT |
| status | TEXT | "status" TEXT |
| password | TEXT | "password" TEXT |



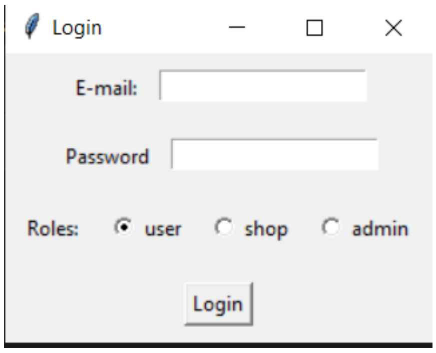
| | | |
|---|---|---|
| **SHOP_APPOINTMENT** | | CREATE TABLE SHOP_APPOINTMENT(shopId INTEGER,appointmentId INTEGER,FOREIGN KEY(shopId) REFERENCES REPAIR_SHOP(shopId),FOREIGN KEY(appointmentId) REFERENCES APPOINTMENT(appointmentId),PRIMARY KEY(shopId, appointmentId)) |
| shopId | INTEGER | "shopId" INTEGER |
| appointmentId | INTEGER | "appointmentId" INTEGER |
| **USER** | | CREATE TABLE USER(userId INTEGER PRIMARY KEY,name |

1

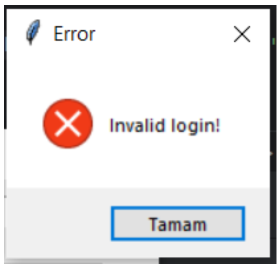
| İsim | Tip | Şema |
|---|---|---|
| | | TEXT,email TEXT,password TEXT,phoneNumber TEXT) |
| userId | INTEGER | "userId" INTEGER |
| name | TEXT | "name" TEXT |
| email | TEXT | "email" TEXT |
| password | TEXT | "password" TEXT |
| phoneNumber | TEXT | "phoneNumber" TEXT |
| **USER_VEHICLE** | | CREATE TABLE USER_VEHICLE(userId INTEGER,vehicleId INTEGER,FOREIGN KEY(userId) REFERENCES USER(userId),FOREIGN KEY(vehicleId) REFERENCES VEHICLE(vehicleId),PRIMARY KEY(userId, vehicleId)) |
| userId | INTEGER | "userId" INTEGER |
| vehicleId | INTEGER | "vehicleId" INTEGER |
| **VEHICLE** | | CREATE TABLE VEHICLE(vehicleId INTEGER PRIMARY KEY,miles INTEGER,type TEXT,model TEXT,year INTEGER) |
| vehicleId | INTEGER | "vehicleId" INTEGER |
| miles | INTEGER | "miles" INTEGER |
| type | TEXT | "type" TEXT |
| model | TEXT | "model" TEXT |
| year | INTEGER | "year" INTEGER |
| **VEHICLE_MAINTENANCE** | | CREATE TABLE VEHICLE_MAINTENANCE(vehicleId INTEGER,maintenanceId INTEGER,FOREIGN KEY(vehicleId) REFERENCES VEHICLE(vehicleId),FOREIGN KEY(maintenanceId) REFERENCES MAINTENANCE(maintenanceId),PRIMARY KEY(vehicleId, maintenanceId)) |
| vehicleId | INTEGER | "vehicleId" INTEGER |
| maintenanceId | INTEGER | "maintenanceId" INTEGER |

**Then he code the necessary queries for our system.**

**Ata dealed with client side of our system in Python. Client side has a connection with server side. In client side, we have several GUIs. They are as follows:**

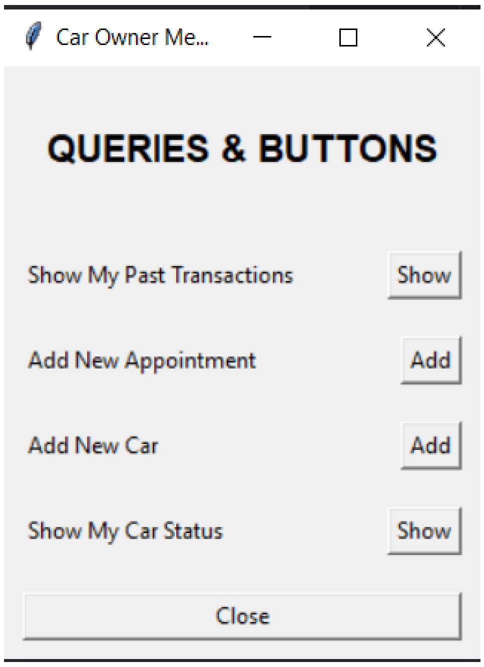


*Şekil 1:Login GUI*

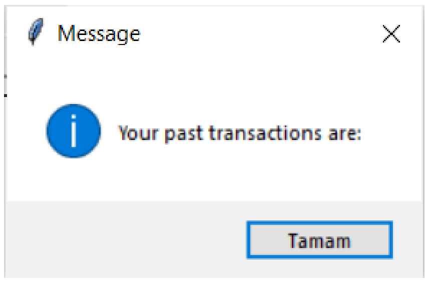


Invalid login!

*Şekil 2: Invalid login*

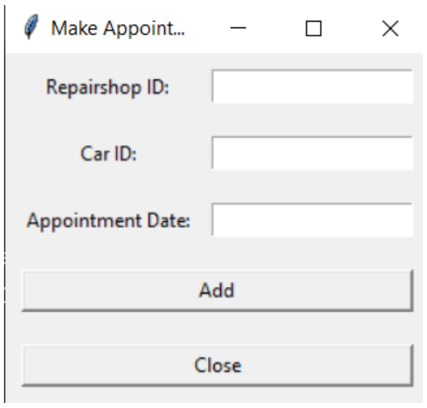


*Şekil 3: User(Car Owner) Menu*

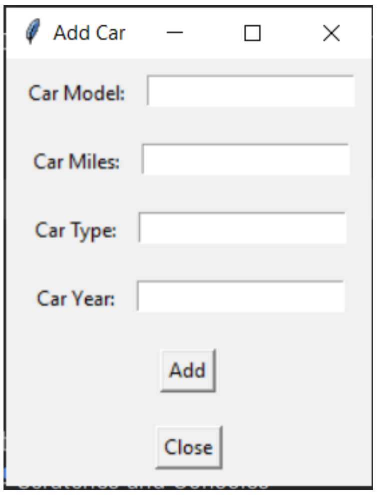


Your past transactions are:

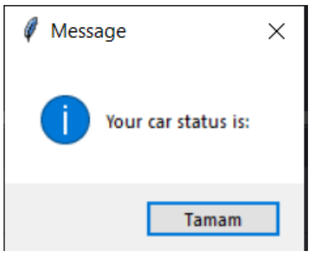
*Şekil 4: User Menu - Show my Past Transactions (To be implemented)*



*Şekil 5: User Menu- Make Appointment(To be implemented)*

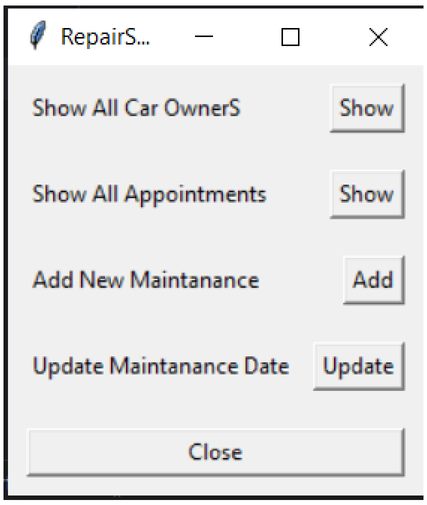


*Şekil 6: User Menu- Add New Car (to be implemented)*

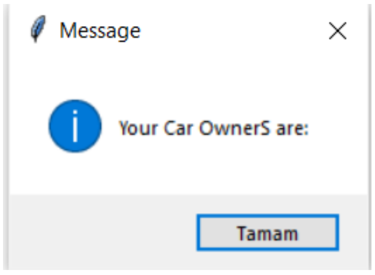


Your car status is:

*Şekil 7: User Menu- Show Car Status (to be implemented)*

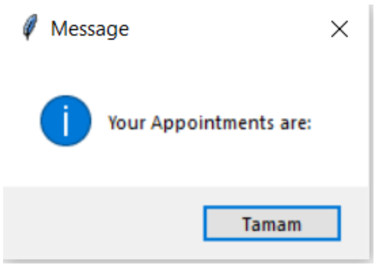


*Şekil 8: Repairshop Menu*


Message

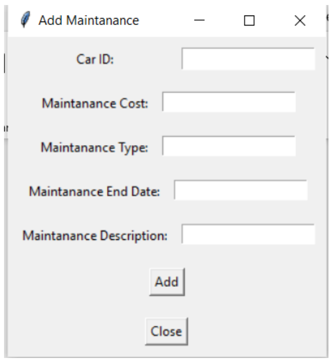
Your Car OwnerS are:

*Şekil 9: Repairshop Menu- Show all Car Owners (to be completed)*


Message

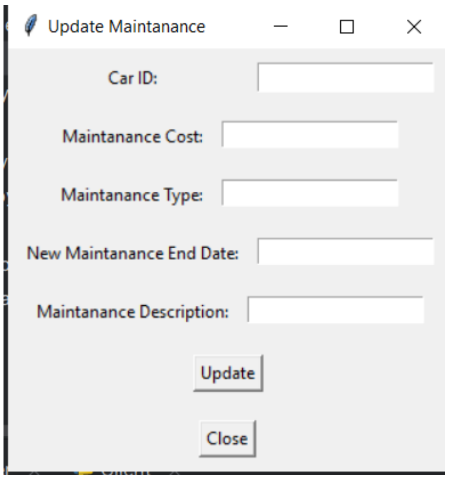
Your Appointments are:

*Şekil 10: Repairshop Menu - Show All Appointment (to be implemented)*

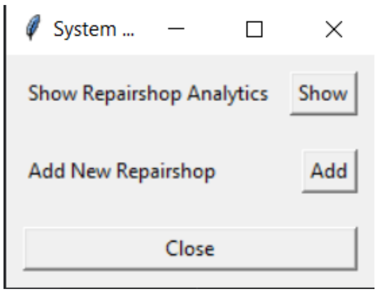

Add Maintanance

|  |  |
|---|---|
| Car ID: |  |
| Maintanance Cost: |  |
| Maintanance Type: |  |
| Maintanance End Date: |  |
| Maintanance Description: |  |

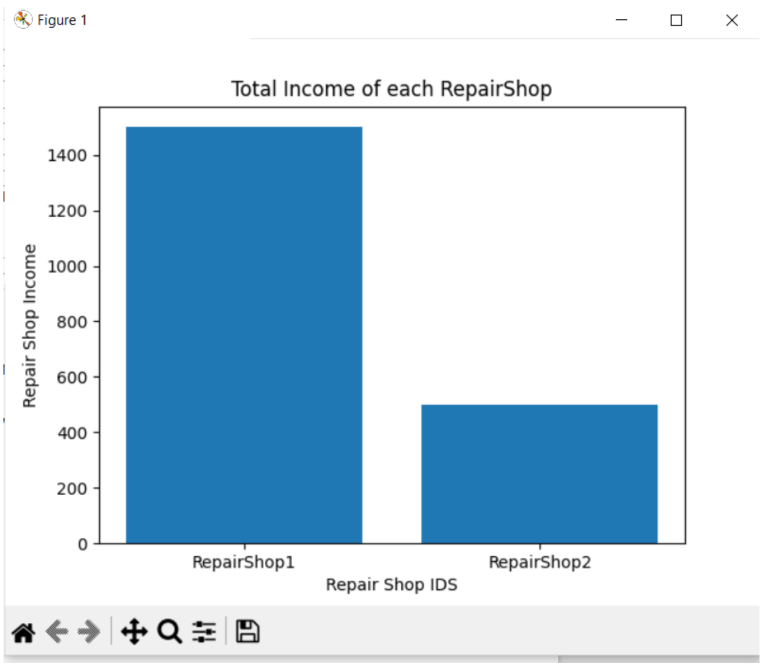
*Şekil 11: Add new maintanance (to be implemented)*



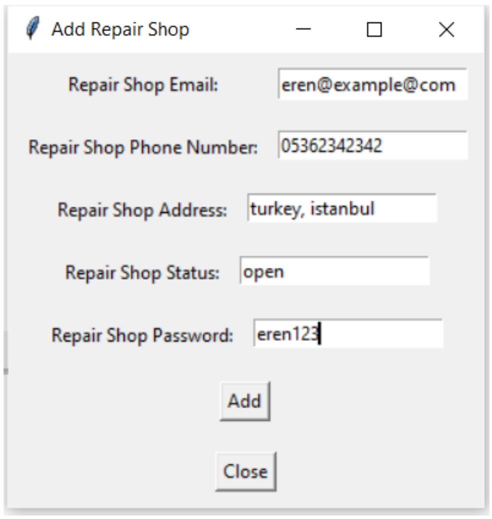
*Şekil 12: Update Maintanance(to be implemented)*



*Şekil 13: System Admin Menu*



*Şekil 14: Show Repairshow Anaysis*



Repair Shop Email: eren@example@com

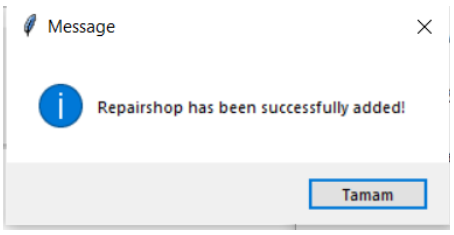
Repair Shop Phone Number: 05362342342

Repair Shop Address: turkey, istanbul

Repair Shop Status: open

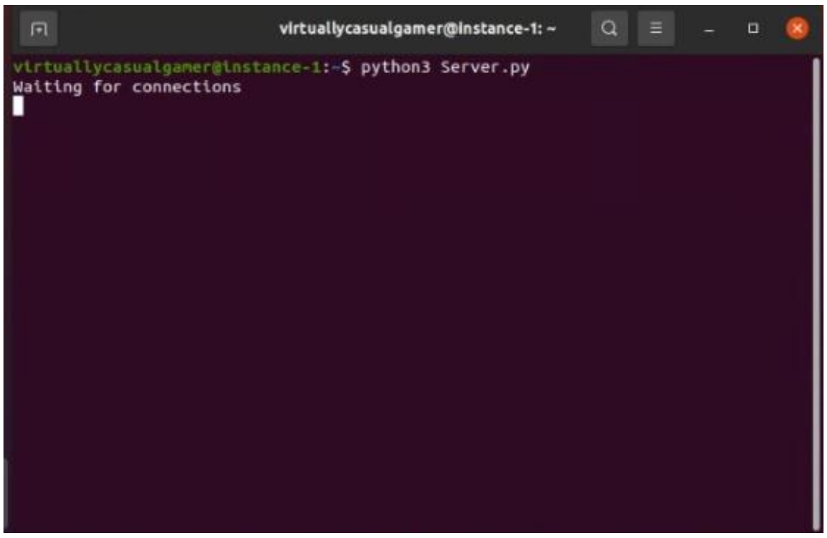
Repair Shop Password: eren123

*Şekil 15: Add new repairshop*



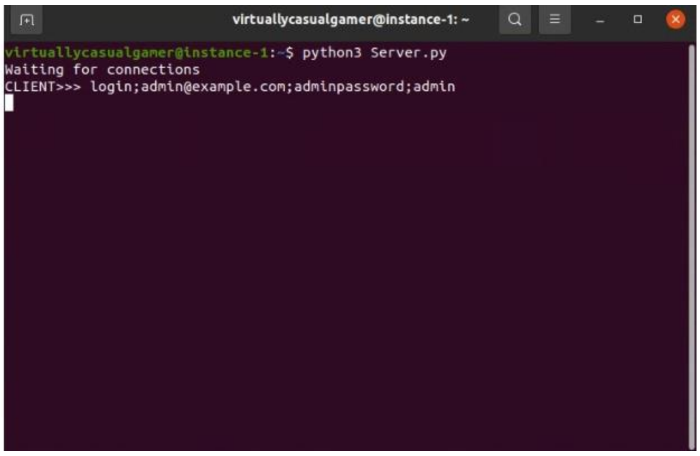
Repairshop has been successfully added!

*Şekil 16: It is added successfully*



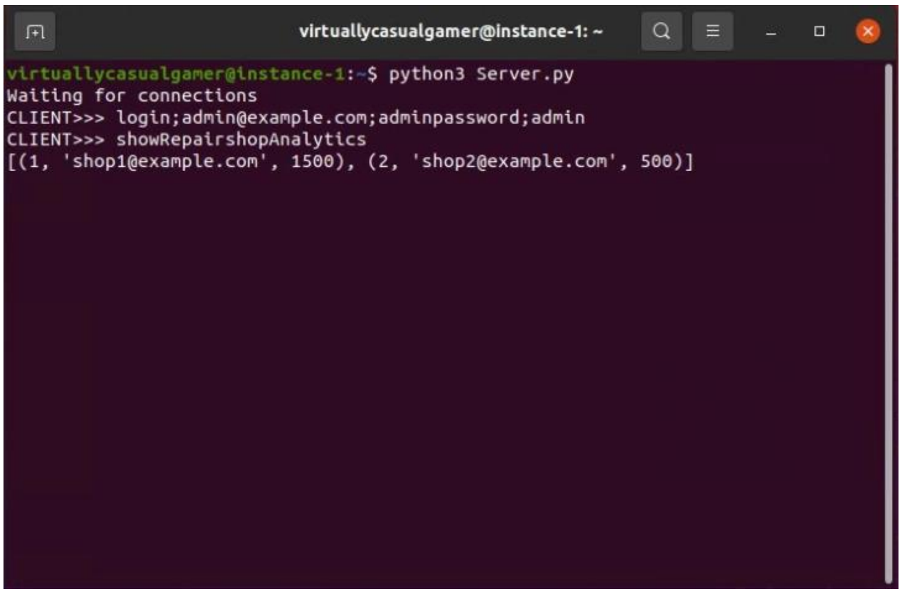
```
virtuallycasualgamer@instance-1:~$ python3 Server.py
Waiting for connections
```

*Şekil 17: Server waits for connections*



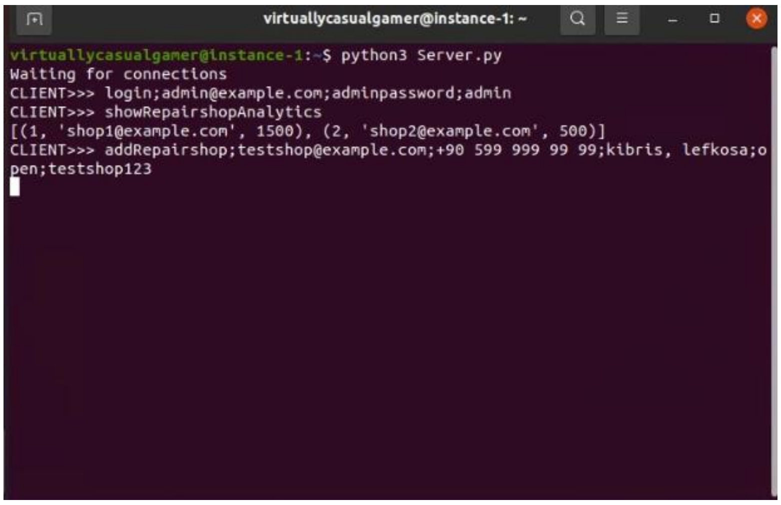
```
virtuallycasualgamer@instance-1:~$ python3 Server.py
Waiting for connections
CLIENT>>> login;admin@example.com;adminpassword;admin
```

*Şekil 18: login query coming from client*



```
virtuallycasualgamer@instance-1:~$ python3 Server.py
Waiting for connections
CLIENT>>> login;admin@example.com;adminpassword;admin
CLIENT>>> showRepairshopAnalytics
[(1, 'shop1@example.com', 1500), (2, 'shop2@example.com', 500)]
```

*Şekil 19: show repairshop analysis query coming from client*



```
virtuallycasualgamer@instance-1:~$ python3 Server.py
Waiting for connections
CLIENT>>> login;admin@example.com;adminpassword;admin
CLIENT>>> showRepairshopAnalytics
[(1, 'shop1@example.com', 1500), (2, 'shop2@example.com', 500)]
CLIENT>>> addRepairshop;testshop@example.com;+90 599 999 99 99;kibris, lefkosa;o
pen;testshop123
```

*Şekil 20: add repairshop query coming from client*

## PART3

### Google Cloud Platform

Google Cloud Platform (GCP) is a suite of cloud computing services provided by Google. It offers a variety of services, including computing power, storage, databases, machine learning, and networking, among others. GCP allows businesses and developers to build, deploy, and scale applications and services on Google's infrastructure.

**For this Cloud technology, we watched a video on youtube, which is the following link:**

https://www.youtube.com/watch?v=RFPlXmgKCtk&ab_channel=AdamTMedia

### Virtual Machine

A virtual machine (VM) is a software-based emulation of a physical computer. It runs an operating system and applications just like a physical machine but is hosted on a physical computer known as a host. Virtual machines are commonly used for server consolidation, testing environments, and running multiple operating systems on a single physical machine. They provide flexibility, resource isolation, and ease of management in computing environments.

**We created a virtual machine in Google Cloud Platform, and we run our server.py in this machine on Ubuntu. Moreover, to connect virtual machine outside, we got the external IP from Google Cloud platform.**

## PART 4:

**Explations and difficulties we achieved:**

For database, in server side, we got an error regarding the table creations and inserting the data. To solve this problem, Eren found a solution which implements operating system module.

For client side, we got an error to connect with server. Later, Ata changed the socket implementation, so the problem is solved.

For server, we really stuck several hours to connect the server with Cloud. Then Serhan watches a youtube tutorial regarding the connection of our server code with Cloud platform we used, then we successfully managed the overall connection.

**Part 5**

1. Complete the not completed parts of GUI, such as RepairShop functionalities (ATA)
2. Add new GUIs to our system (ATA)
3. Better optimization for our database, especially “Scheduling” (EREN)
4. Create new queries and databases for GUIs that will be added (EREN)
5. Adding new functionalities for queries coming from client on server side. (Serhan)
6. Implementing  a notification system that is connected with Cloud. For example, Cloud system will send a notification message to Car owner whose car is almost repaired. (SERHAN)

WEEK1:

Ata planning to complete first part of his task (option1)

WEEK2

EREN planning to complete first part of his task (option3)

SERHAN planning to complete first part of his task (option5)

WEEK3

EREN planning to complete second part of his task (option4)

SERHAN planning to complete second part of his task (option6)

Final WEEK

ATA planning to complete second  part of his task (option2)

Finally, we will delivery the code for client and server. Also we will provide a backup of our database.

References:

https://www.youtube.com/watch?v=RFPlXmgKCtk&ab_channel=AdamTMedia

https://www.vmware.com/topics/glossary/content/virtual-machine.html

https://cloud.google.com/gcp/?utm_source=google&utm_medium=cpc&utm_campaign=emea-emea-all-en-dr-bkws-all-all-trial-e-gcp-1011340&utm_content=text-ad-none-any-DEV_c-CRE_500228034474-ADGP_Hybrid+%7C+BKWS+-+EXA+%7C+Txt+~+GCP+~+General%23v22-KWID_43700071659798940-kwd-281541840951-userloc_9070307&utm_term=KW_cloud.google-NET_g-PLAC_&&gad_source=1&gclid=Cj0KCQiA7OqrBhD9ARIsAK3UXh2-qcaWIfdBAq8v6qBmiB4wKv5yFGp144Mp8SC6cJtahMCchEhGF1AaAoNKEALw_wcB&gclsrc=aw.ds

GITHUB LINK

https://github.com/SerhanTRGL/CNG495.git

ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

**CNG 495 – Cloud Computing**

**Term Project Proposal**

# Repair-Shop Car Management System

**Hasan Eren Yarar 2457802**

**Ata Kaleli 2385474**

**Serhan Turgul 2453603**

## Overview

This system is an all-in-one solution designed to simplify the management of vehicle maintenance and repair tasks for both vehicle owners and repair shop owners. It acts as a strong platform that links vehicle owners, service providers, and administrative staff. The system is specially made to improve the process of organizing, managing, and monitoring vehicle repair and maintenance work.

## Benefits

- Strong platform that links vehicle owners, service providers, and administrative staff
- Improve the process of organizing, managing, and monitoring vehicle repair and maintenance work by holding the data in the system database.
- Ease the process of keaving and bringing car as a car owner, and ease the process of maintaining those records for repairshops

## Novelties and novel idea of our Project

In our world, as a car owner, when you have some issues with your car, such as tire problem or oil change etc. , you need to leave your car to maintanance services. The main problem is that you can not assume how much time does the maintanance will takes . So our system eases the process of this big problem.

## Same Projects similar with ours

**AutoShop Manager:** Comprehensive management software for auto repair shops, handling tasks from appointments to inventory.

**GaragePro**: A professional tool aiding garages in streamlined operations, from customer management to service tracking.

**CarCare** Pro: User-friendly software designed to simplify car maintenance, ensuring efficient service and customer satisfaction.

## Github link for the first system

https://github.com/alexiusacademia/autoshop-management-system

## STRUCTURE OF OUR PROJECT

**Key Features and Functionalities*:***

User (Vehicle Owners):

**Registration & Login:** Users are securely register and log in to the system, ensuring authenticated access to the  their vehicle data.

**Maintenance Record View:** Users have access to their past maintenance information.

**User Vehicle Management:** Users can add new vehicles to their profile, including details like model, miles, type, and year.

**Appointment Scheduling:** Users can easily schedule appointments with repair shops, making it more convenient to manage repair and maintenance activities for their vehicles.

Repair Shop Management:

**Access to User Information**: Repair shops can view information about their customers, including contact details and vehicle data.

**Take Maintenance Record:** Shops can add new maintenance records for the services they provide, with detail tracking of all maintenance activities.

**Appointment Management**: Viewing and managing upcoming appointments.

Administrative Functions:

**Repair Shop Statistics**: Administrators can view  total income for each Repair Shop.

**Repair Shop Registration**: New repair shops can be added to the system.

**Technologies we used**

- Frontend: Python Tkinter
- Backend Python 3.9
- Database: SQLITE3
- Cloud Services: Google Cloud Platform

**Google Cloud Platform as an Infastructure as a Service (IaaS):**

We are utilising Google Cloud Platform as an Infastructure as a Service (IaaS), by using their Compute Engine to run a Virtual Machine(VM). We run Ubuntu 20.04 on the VM, to run our server script and database.

What we are doing is a Software as a Service (SaaS). We have a server script that runs on Google Cloud Platform which listens to incoming details from each client, executes queires depending on what is provided by the client, and then sends the results of the query back to the client. We consider this as SaaS because the client is a piece of software that communicates with the cloud. Any number of repair shops can utilize this software to make their business with their clients easier.

**Data Types**

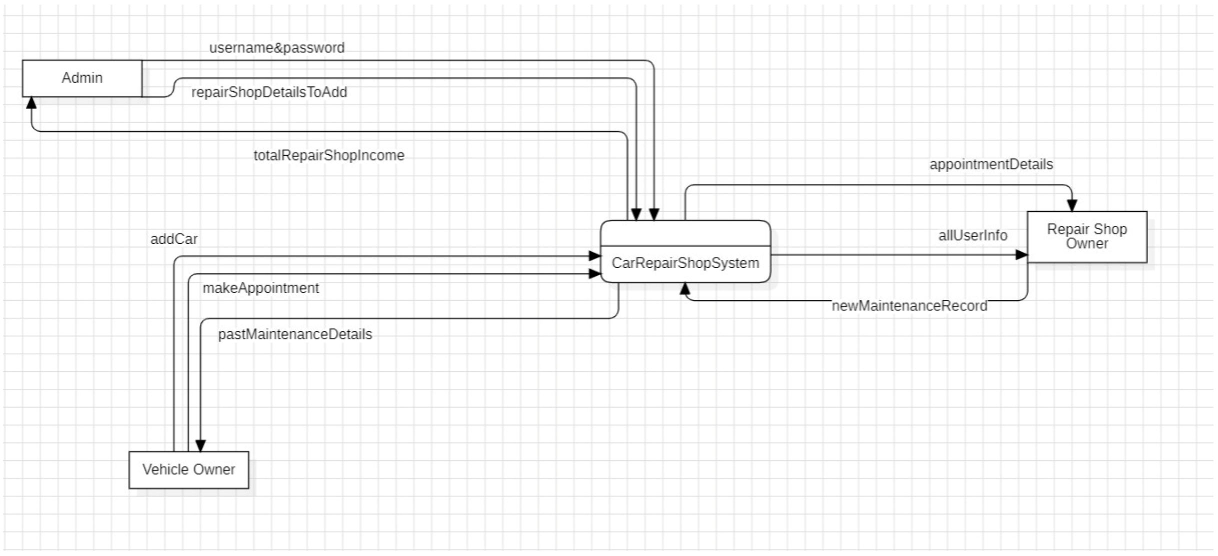
**Text Data:**

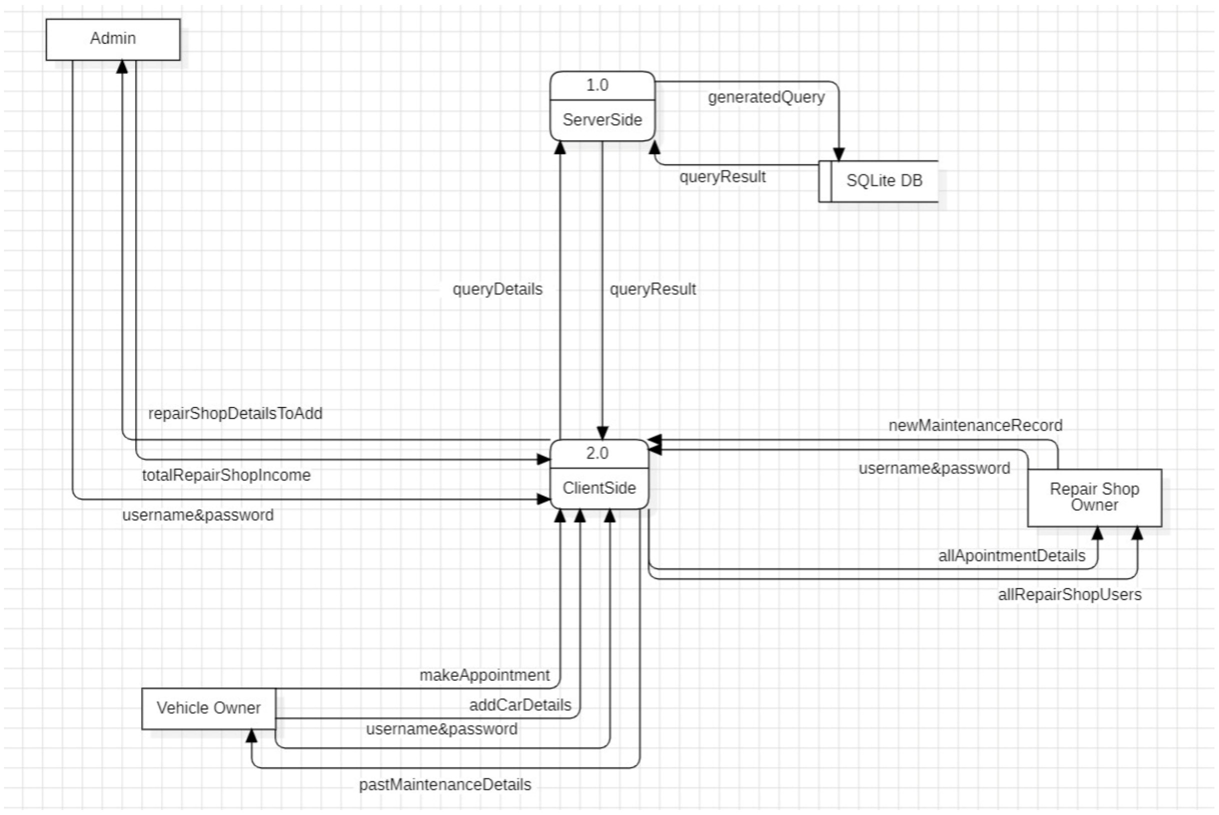
- Maintanance descriptions and details.
- User information such as email name address
- Admin inormation such as email pasword name
- Repair Shop information such as email password

**Numerical Data:**

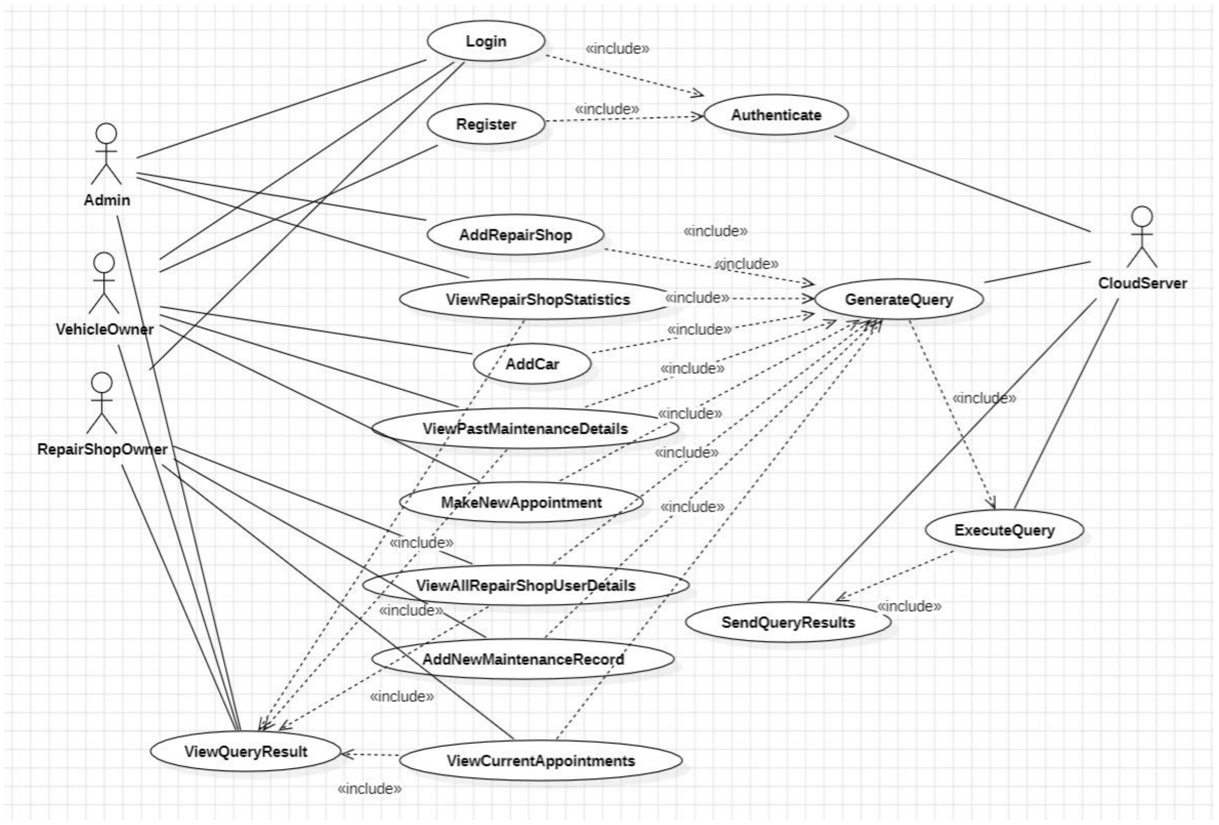
- Miles of car
- Price of the maintanance

## Data Flow Diagram of The System:



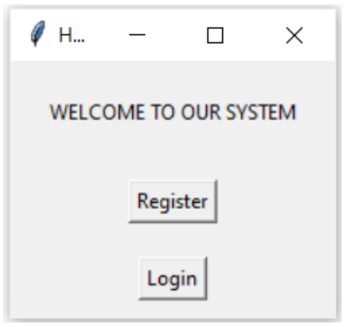


## Client & Service Interactions Diagram:

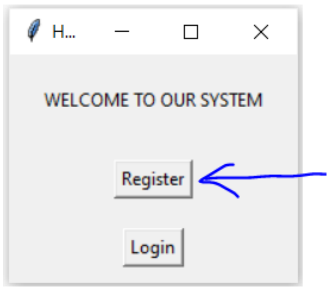


**User Manuel**

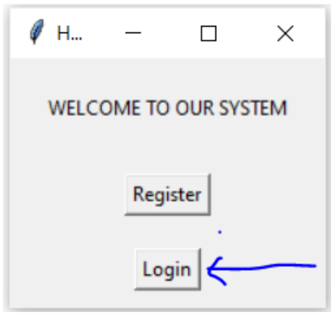
## Home Page



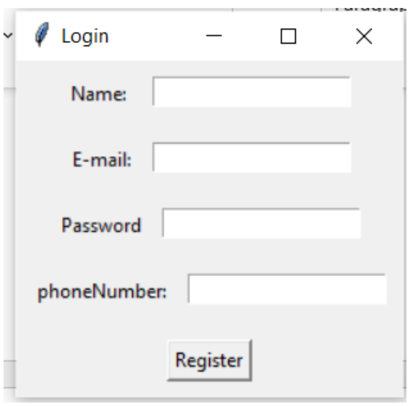
This is our system’s home page. If you dont have an account, you need to create first by clicking the register button.



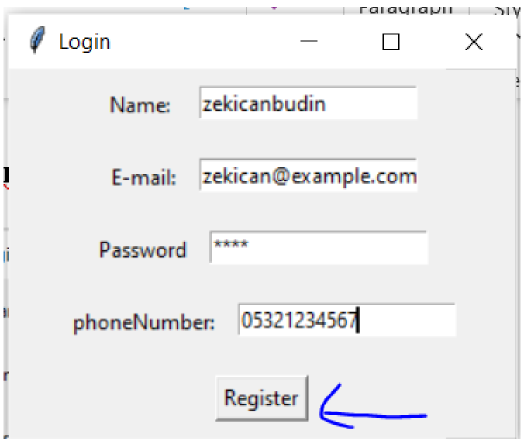
If you have already created your account, you need to login to our system by clicking the login button



## Registration Page

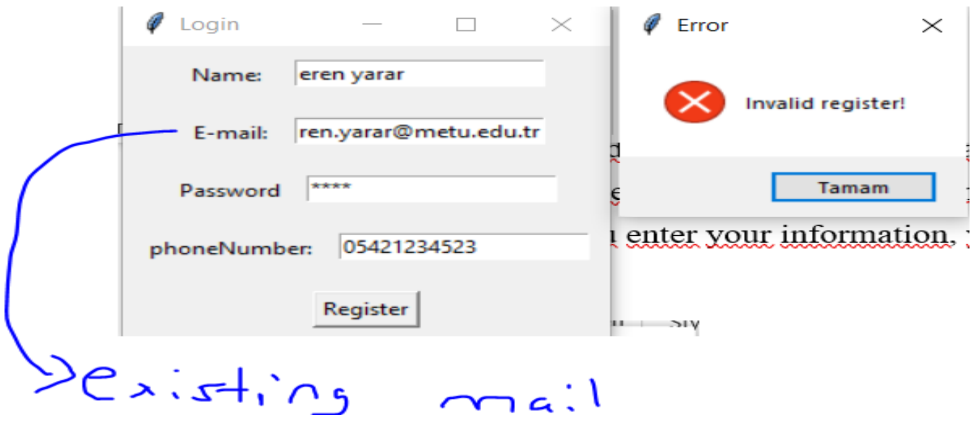


This is registration page. You need to enter your name, email, password and phoneNumber when you are trying to register. Please remember that you need to enter an email which is not existing in our database. After you enter your information, you need to click the register button



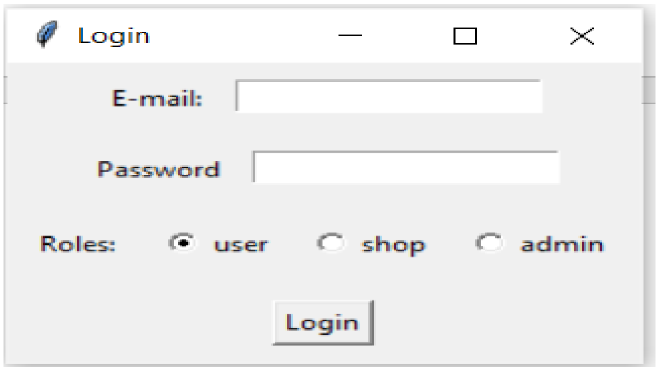
Name: zekicanbudin
E-mail: zekican@example.com
Password ****
phoneNumber: 05321234567

If your registration is success, you will be directed to login page. This means that your registration is succesfully done! Else, you will get an error message that your registration is invalid!



Name: eren yarar
E-mail: ren.yarar@metu.edu.tr
Password ****
phoneNumber: 05421234523

Invalid register!

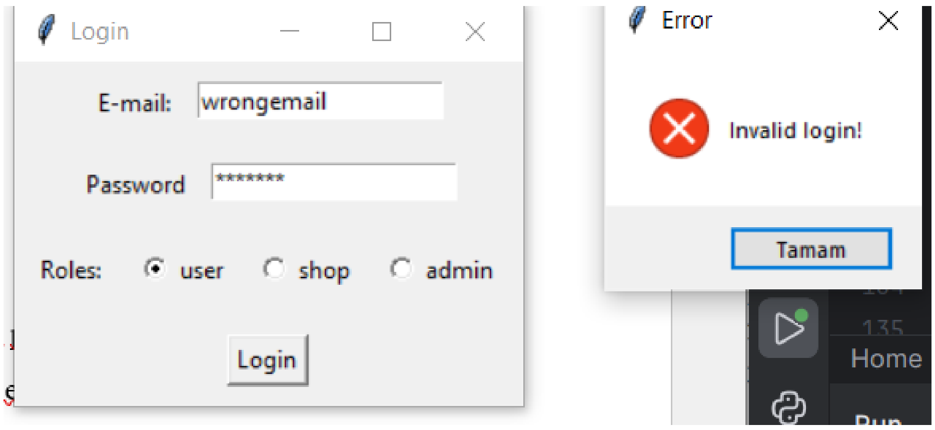

Login

E-mail:

Password

Roles: [✓ user] shop admin

This is our login page. You need to enter your email and password, and select your role. Then you need to click the login. Please remember that you need to enter a valid information, otherwise your login will be unsuccessful!


Login

E-mail: wrongemail

Password *******

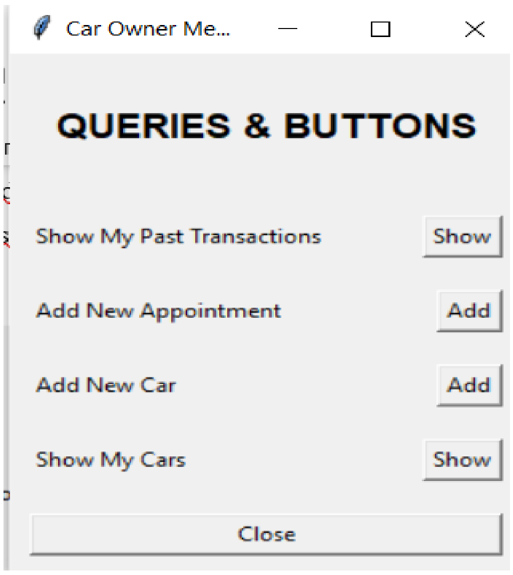
Roles: [✓ user] shop admin

Error

Invalid login!

USER (CAR OWNER) SIDE

This is the screen after you login successfully


Car Owner Me...

QUERIES & BUTTONS

Show My Past Transactions

Add New Appointment

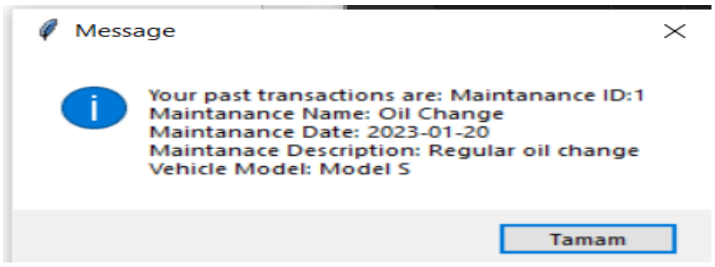
Add New Car

Show My Cars

**In this menu, you can do:**

1. Show your past transactions(car maintanances)
2. Add new appointment
3. Add new car
4. Show my cars

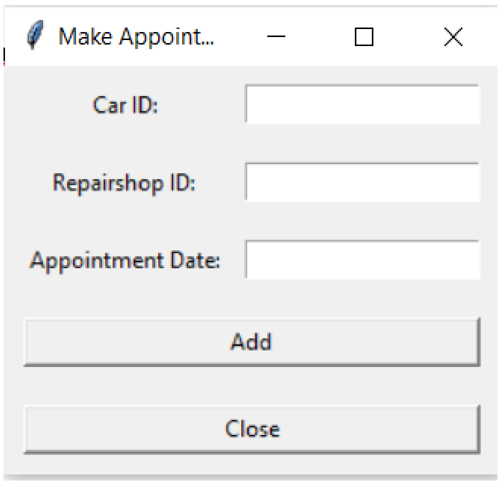
1)



Your past transactions are: Maintanance ID:1
Maintanance Name: Oil Change
Maintanance Date: 2023-01-20
Maintanace Description: Regular oil change
Vehicle Model: Model S

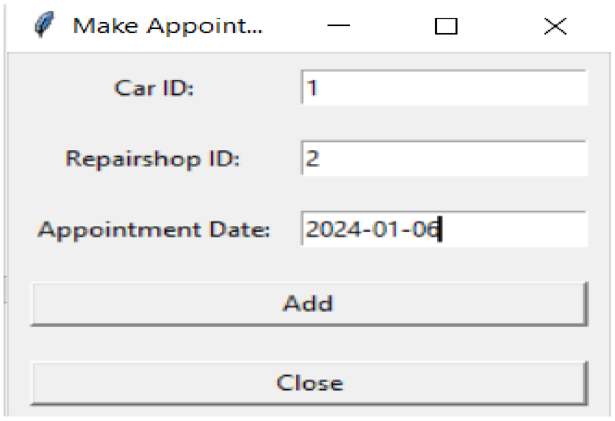
This screen will occur when you click the show button

2)



This menu will occur when you click add button

To add a new appointment, you need to enter your carId, repairshopId and appointmentDate.

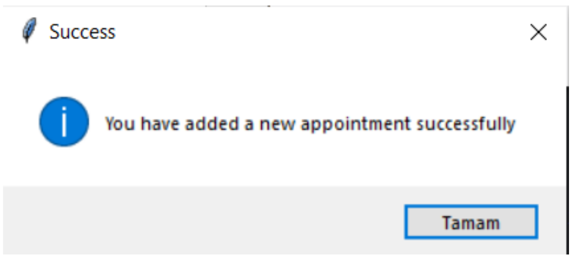


Car ID: 1

Repairshop ID: 2

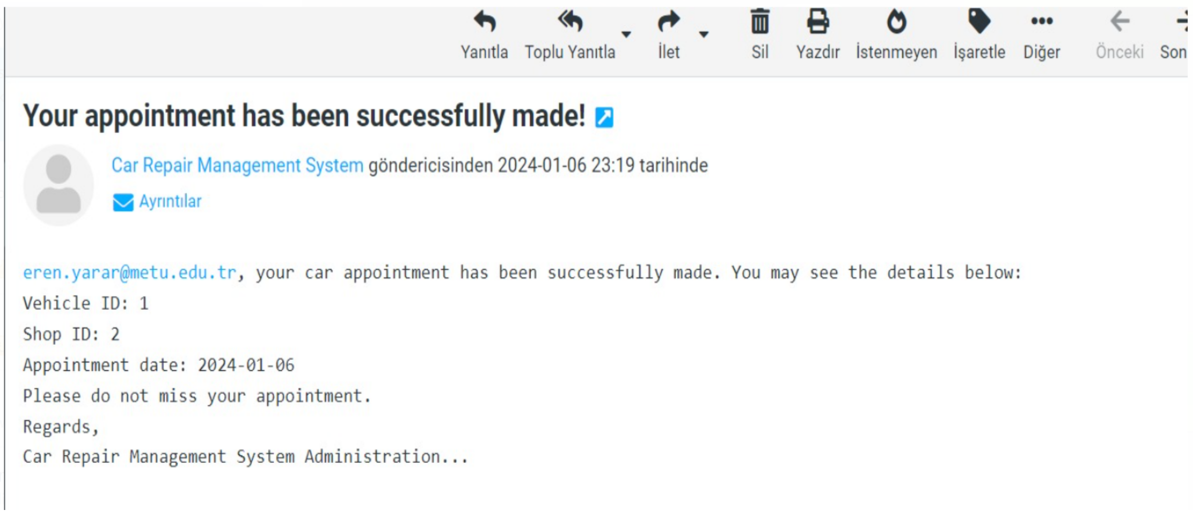
Appointment Date: 2024-01-06

After entering these values, you need to click add button. When your appointment is added, you will be modified by an email thanks to Google mail api.



You have added a new appointment successfully

*Şekil 21:server success message*



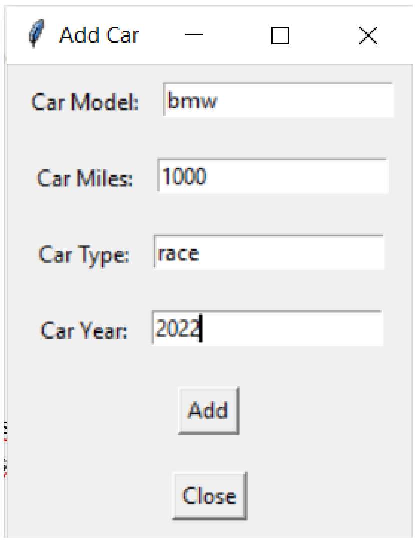
Your appointment has been successfully made!

Car Repair Management System göndericisinden 2024-01-06 23:19 tarihinde

```
eren.yarar@metu.edu.tr, your car appointment has been successfully made. You may see the details below:
Vehicle ID: 1
Shop ID: 2
Appointment date: 2024-01-06
Please do not miss your appointment.
Regards,
Car Repair Management System Administration...
```

*Şekil 22: mail succeess message*

3) To add new car, you need to enter the details of your car’s model,miles,type and year. After you enter these informations, you will be get successful message if your car is added to the system

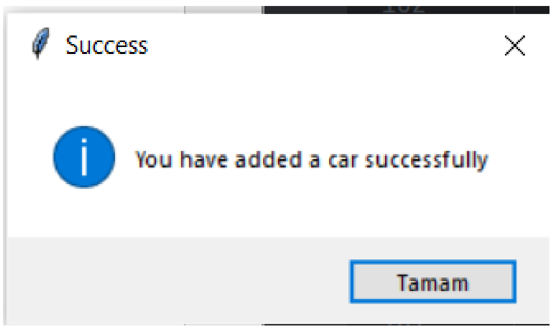

Add Car

Car Model: bmw

Car Miles: 1000

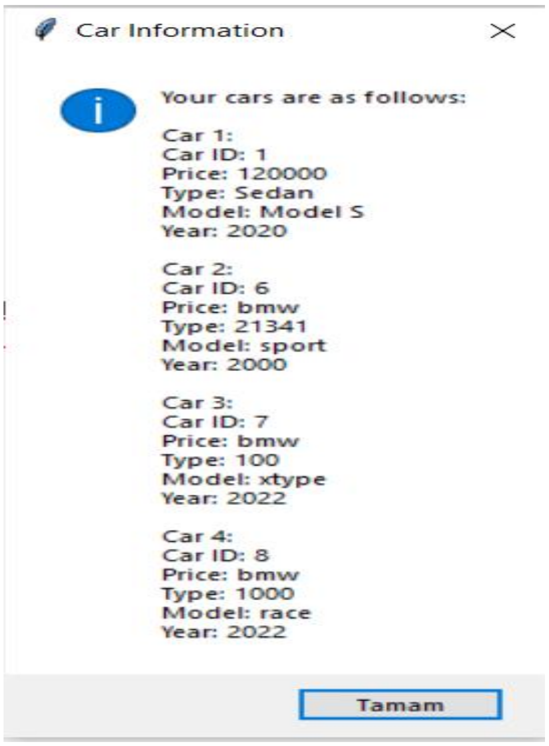
Car Type: race

Car Year: 2022


Success

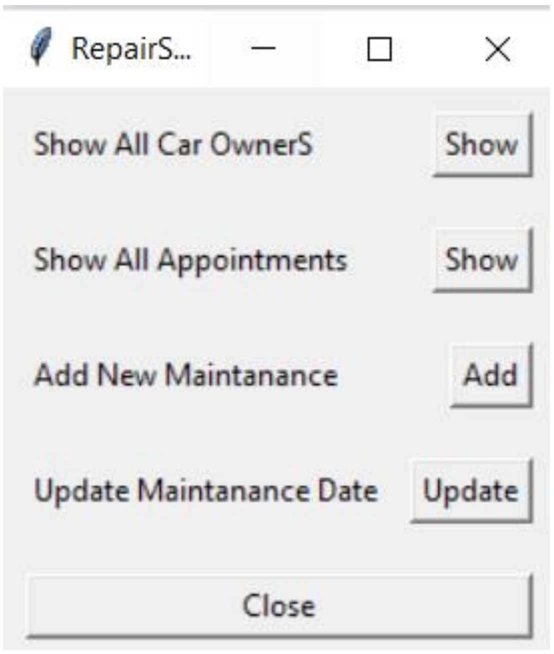
You have added a car successfully

4) When you click show my cars, you can see the cars that are registered to the system. If you dont have any car, you will get a message saying that you dont have any car in the system.



```
Your cars are as follows:

Car 1:
Car ID: 1
Price: 120000
Type: Sedan
Model: Model S
Year: 2020

Car 2:
Car ID: 6
Price: bmw
Type: 21341
Model: sport
Year: 2000

Car 3:
Car ID: 7
Price: bmw
Type: 100
Model: xtype
Year: 2022

Car 4:
Car ID: 8
Price: bmw
Type: 1000
Model: race
Year: 2022
```

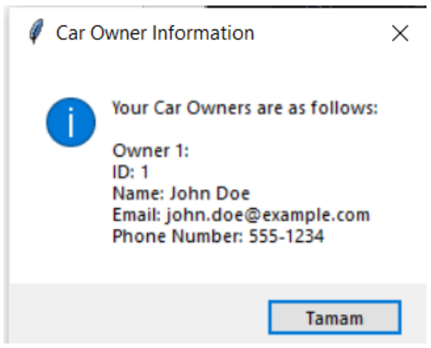
REPAIR SHOP SIDE



This is the menu that will shown to you. You can do the followings from this menu:

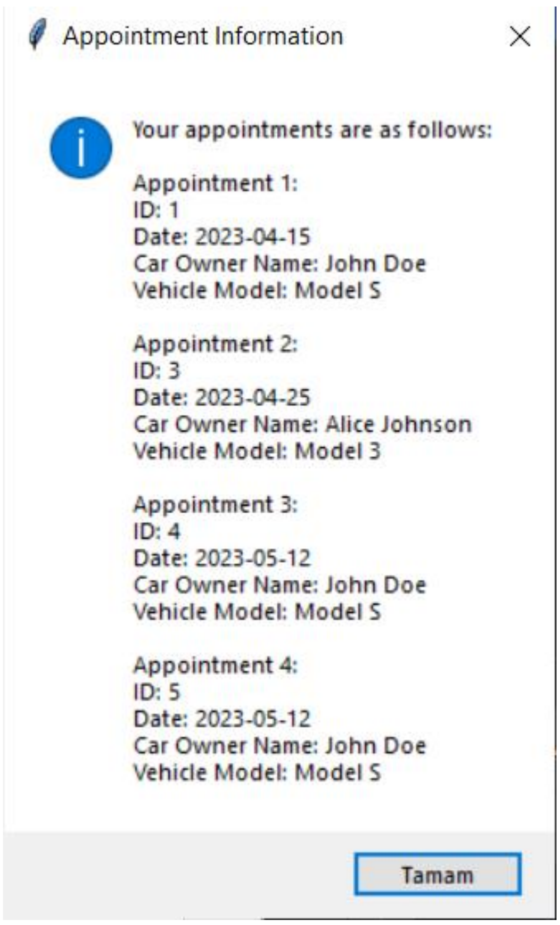
1. Show all car owners
2. Show all appointments
3. Add new maintanance
4. Update maintanance

1)When you click show, you can see the car owners that will register to you



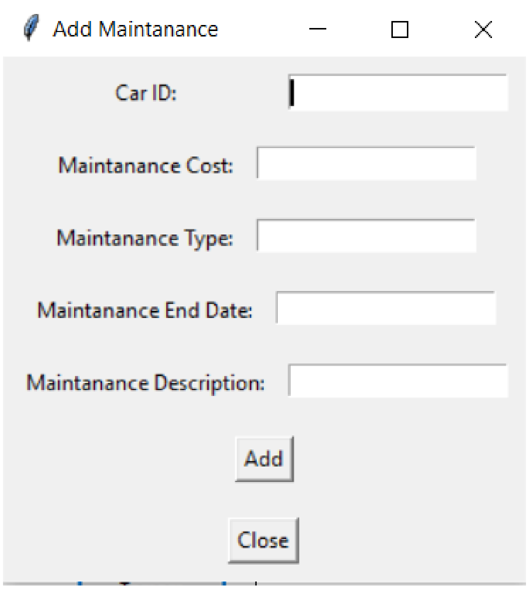
```
Your Car Owners are as follows:

Owner 1:
ID: 1
Name: John Doe
Email: john.doe@example.com
Phone Number: 555-1234
```

2)When you click show, you can see all the appointments done by car owners to you



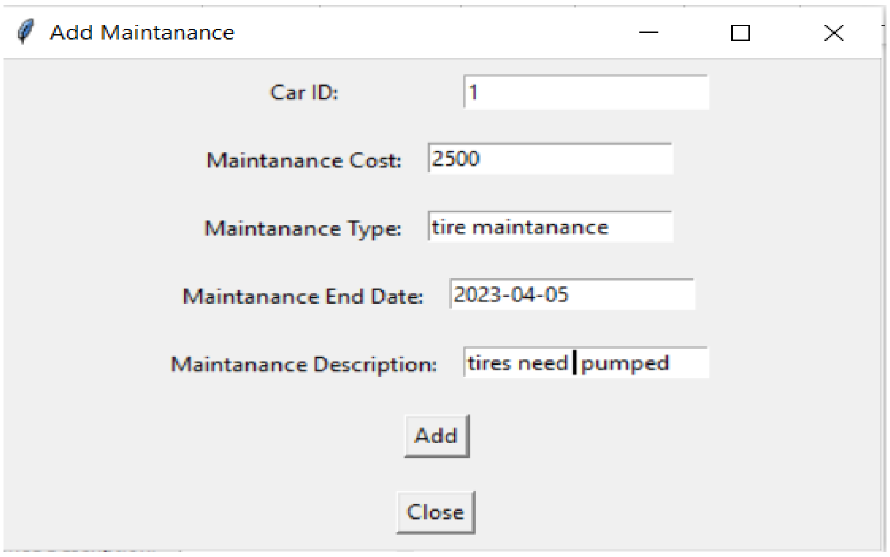
```
Your appointments are as follows:

Appointment 1:
ID: 1
Date: 2023-04-15
Car Owner Name: John Doe
Vehicle Model: Model S

Appointment 2:
ID: 3
Date: 2023-04-25
Car Owner Name: Alice Johnson
Vehicle Model: Model 3

Appointment 3:
ID: 4
Date: 2023-05-12
Car Owner Name: John Doe
Vehicle Model: Model S

Appointment 4:
ID: 5
Date: 2023-05-12
Car Owner Name: John Doe
Vehicle Model: Model S
```

3)when you click add to add a new maintanace, you will see this screen


Add Maintanance

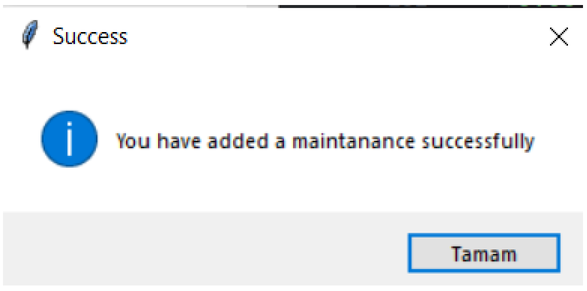
| | |
|---|---|
| Car ID: | |
| Maintanance Cost: | |
| Maintanance Type: | |
| Maintanance End Date: | |
| Maintanance Description: | |

After entering maintanance details, you need to press add button


Add Maintanance

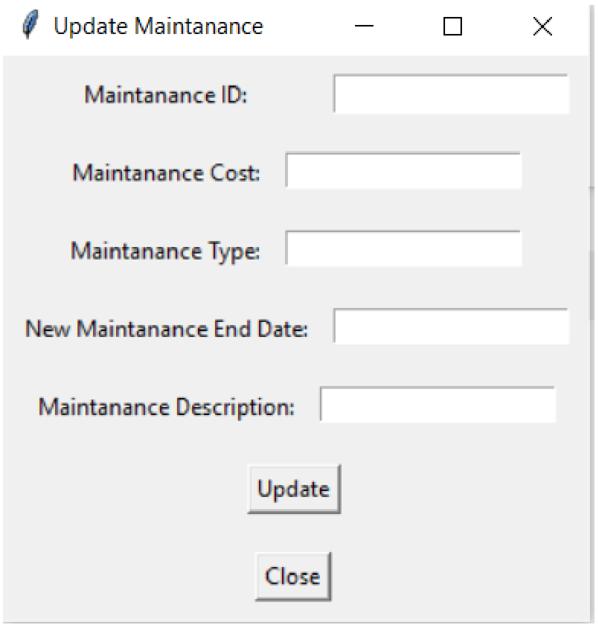
| | |
|---|---|
| Car ID: | 1 |
| Maintanance Cost: | 2500 |
| Maintanance Type: | tire maintanance |
| Maintanance End Date: | 2023-04-05 |
| Maintanance Description: | tires need pumped |

If there will be no problem, you will see this message box that confirms the addition of maintanace


Success

You have added a maintanance successfully

4)You need to click update button

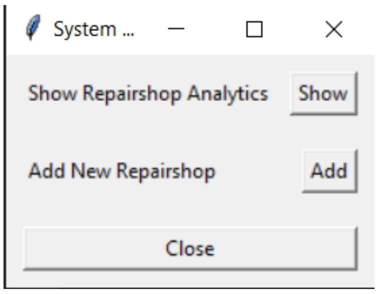

Update Maintanance

| | |
|---|---|
| Maintanance ID: | |
| Maintanance Cost: | |
| Maintanance Type: | |
| New Maintanance End Date: | |
| Maintanance Description: | |

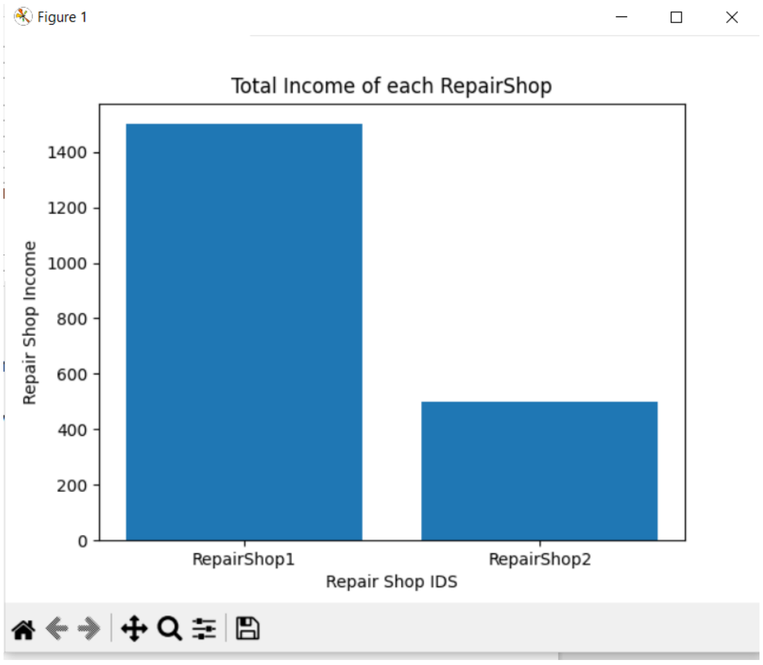
In this page, you need to enter all the details and click update. After that, the data will be updated and If there is no problem, you will achieve a success message.

## SYSTEM ADMIN SIDE

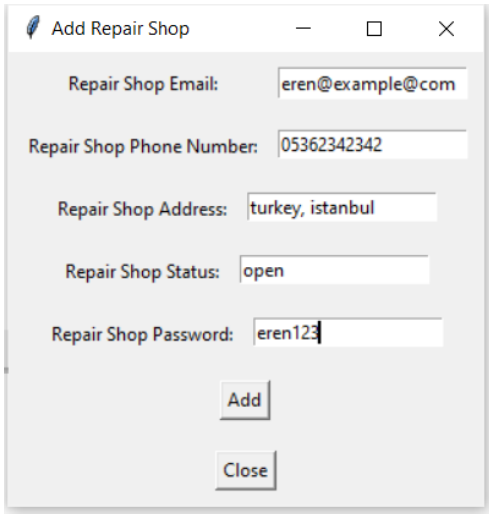
You have two option to do. One of them is show repairshop analytic, and the other one is add new repairshop. The first option shows you the total income of the repairshops compared in a bar chart. The other one adds a new repairshop to system.



*Şekil 23: System Admin Menu*



*Şekil 24: Show Repairshow Anaysis*



Add Repair Shop

Repair Shop Email: eren@example@com

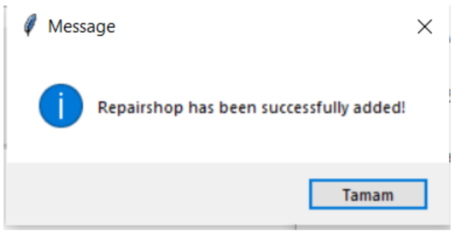
Repair Shop Phone Number: 05362342342

Repair Shop Address: turkey, istanbul

Repair Shop Status: open

Repair Shop Password: eren123

*Şekil 25: Add new repairshop*



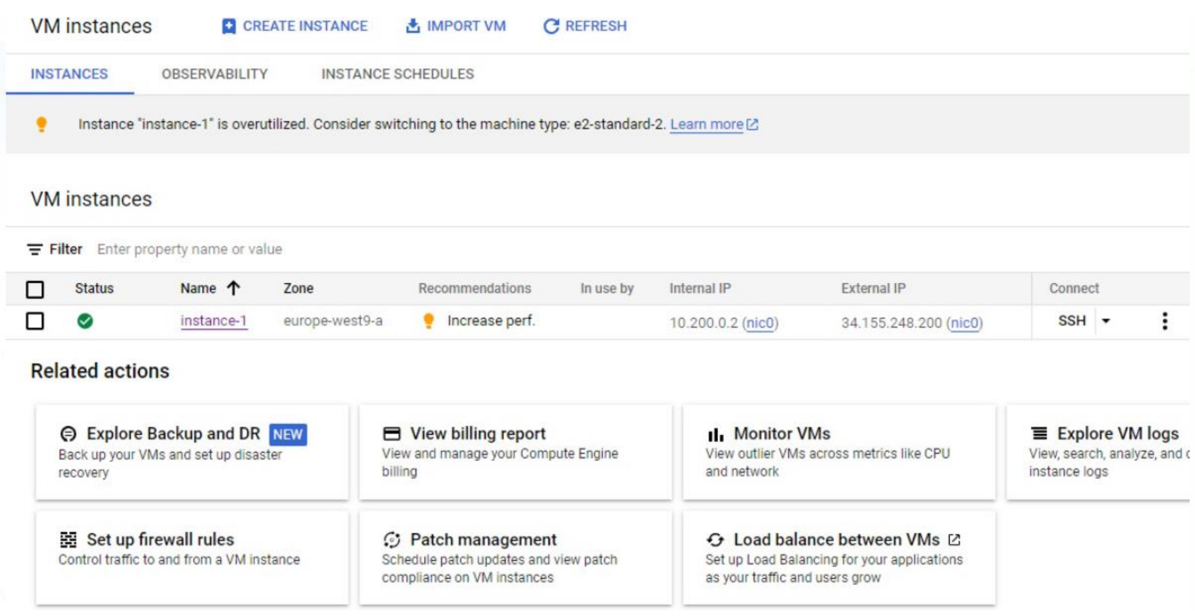
Message

Repairshop has been successfully added!

*Şekil 26: It is added successfully*

## PROJECT STATISTICS

We used sqlite3 from Python for database.


VM instances  CREATE INSTANCE  IMPORT VM  REFRESH

INSTANCES  OBSERVABILITY  INSTANCE SCHEDULES

Instance "instance-1" is overutilized. Consider switching to the machine type: e2-standard-2. Learn more

VM instances

Filter  Enter property name or value

| Status | Name ↑ | Zone | Recommendations | In use by | Internal IP | External IP | Connect |
|---|---|---|---|---|---|---|---|
| ✓ | instance-1 | europe-west9-a | Increase perf. | | 10.200.0.2 (nic0) | 34.155.248.200 (nic0) | SSH |

Related actions

Explore Backup and DR NEW
Back up your VMs and set up disaster recovery

View billing report
View and manage your Compute Engine billing

Monitor VMs
View outlier VMs across metrics like CPU and network

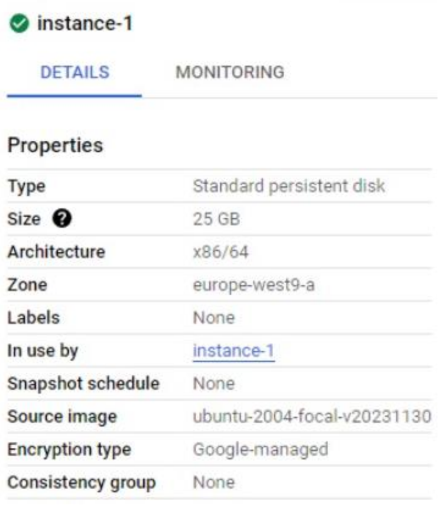
Explore VM logs
View, search, analyze, and d instance logs

Set up firewall rules
Control traffic to and from a VM instance

Patch management
Schedule patch updates and view patch compliance on VM instances

Load balance between VMs
Set up Load Balancing for your applications as your traffic and users grow

This is the size of our virtual machine's SSD. Our db is limited by this much


instance-1

DETAILS  MONITORING

Properties

| | |
|---|---|
| Type | Standard persistent disk |
| Size | 25 GB |
| Architecture | x86/64 |
| Zone | europe-west9-a |
| Labels | None |
| In use by | instance-1 |
| Snapshot schedule | None |
| Source image | ubuntu-2004-focal-v20231130 |
| Encryption type | Google-managed |
| Consistency group | None |

Serhan:

Implementing Cloud infrastructure – time spent 4 days

Implementing Mail in appointment part – time spent 1 day

Ata:

Implementing server-client side repair shop and system admin – time spent 5 days

Eren:

Implementing server-client side with user side and home pageS– time spent 5 days

This is the total amount of time, we start working at 15.12.2023 and we finished working at 06.01.2024

References:

https://www.youtube.com/watch?v=RFPlXmgKCtk&ab_channel=AdamTMedia

https://www.vmware.com/topics/glossary/content/virtual-machine.html

https://cloud.google.com/gcp/?utm_source=google&utm_medium=cpc&utm_campaign=emea-emea-all-en-dr-bkws-all-all-trial-e-gcp-1011340&utm_content=text-ad-none-any-DEV_c-CRE_500228034474-ADGP_Hybrid+%7C+BKWS+-+EXA+%7C+Txt+~+GCP+~+General%23v22-KWID_43700071659798940-kwd-281541840951-userloc_9070307&utm_term=KW_cloud.google-NET_g-PLAC_&&gad_source=1&gclid=Cj0KCQiA7OqrBhD9ARIsAK3UXh2-qcaWIfdBAq8v6qBmiB4wKv5yFGp144Mp8SC6cJtahMCchEhGF1AaAoNKEALw_wcB&gclsrc=aw.ds

GITHUB LINK

https://github.com/SerhanTRGL/CNG495.git