CAR WASH PROGRAM

Object Oriented Programming

FINAL PROJECT

Group Members:

- ✓ **Natnael Zerai** **M80007968**
- ✓ **Aymen Zekeria** **M80007954**
- ✓ **Salem Maktoom** **201709026**

Course: **SEC 321 - 906**

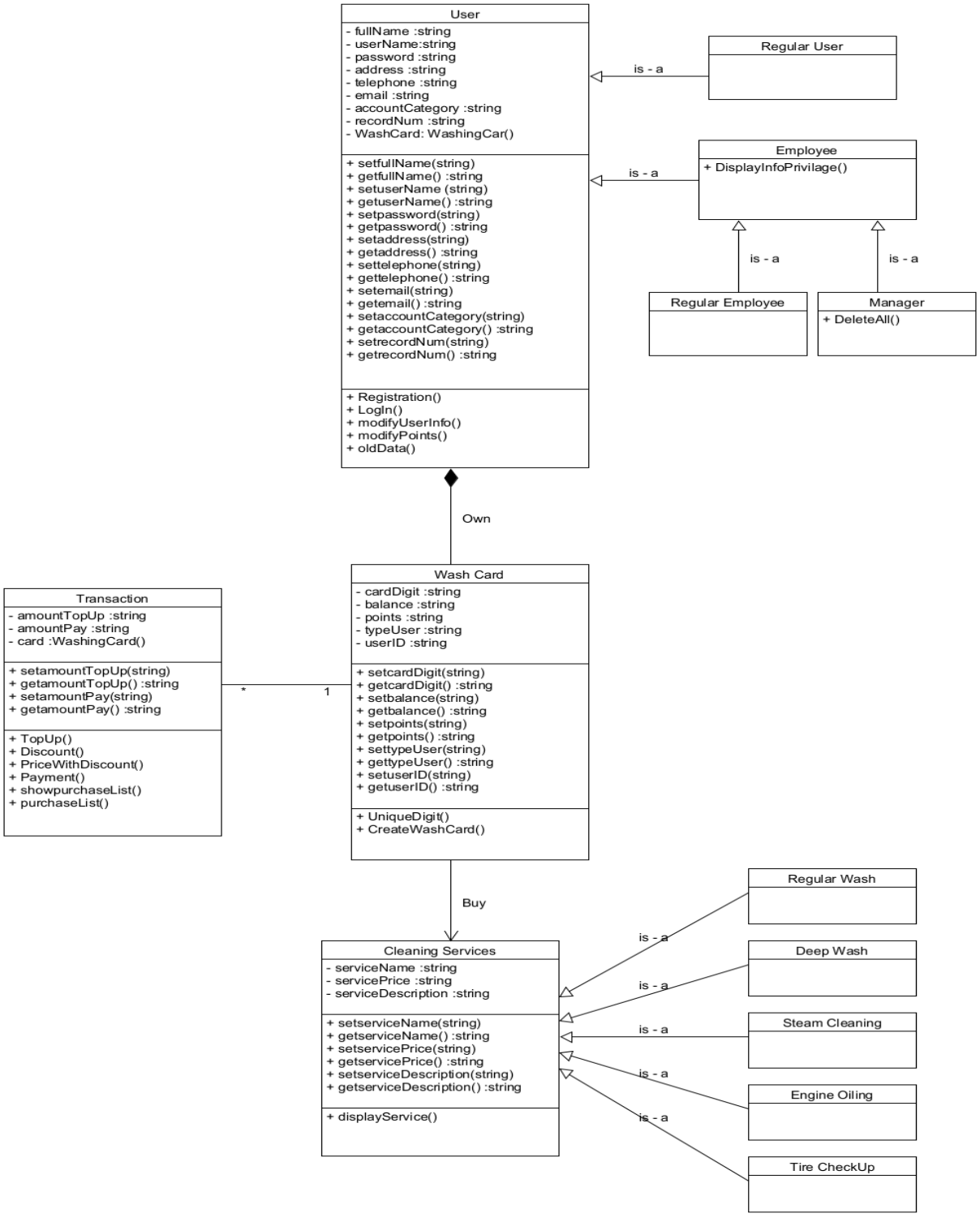
Resources: https://snort.org/

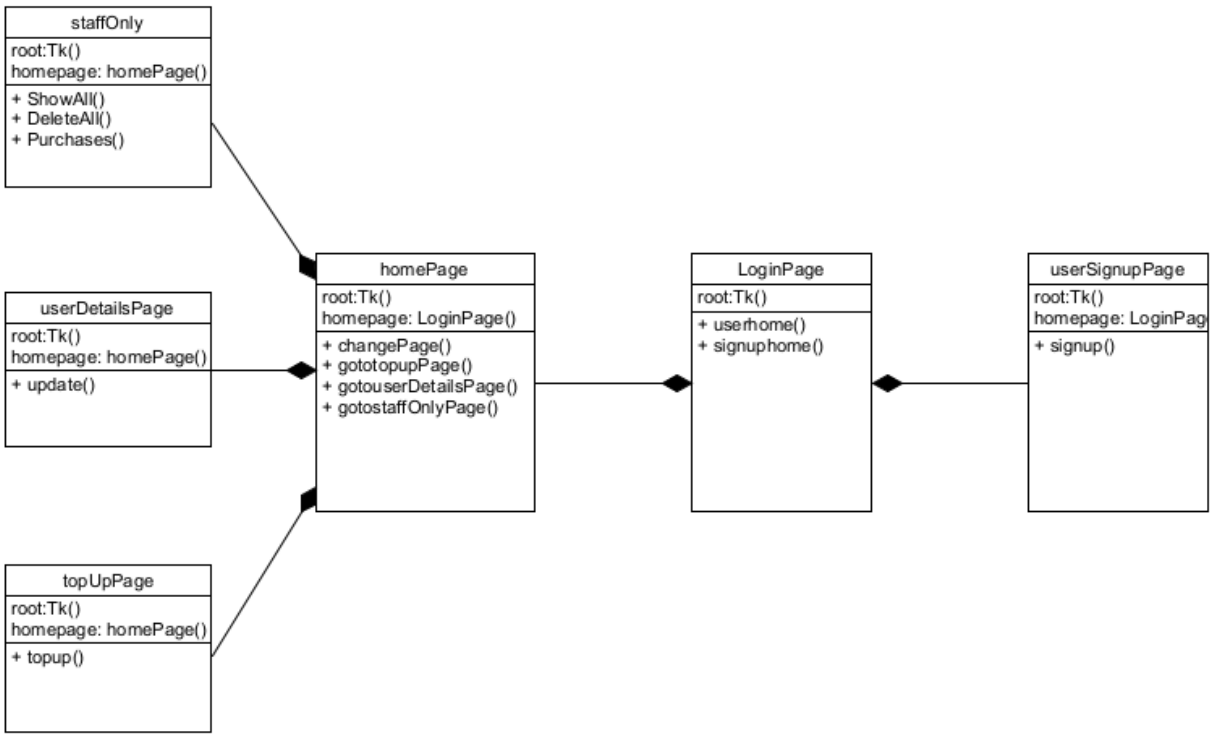
Section 1- Problem Analysis

This project is aimed to provide a digital administration system for a car wash company. There will be a sophisticated program that runs in the background and there will be a user friendly GUI interface that will help the users enjoy the service conveniently. Every person who will access the application will need to be registered in the filesystem (which will be a text file to store the details), if the user isn't registered yet he will be asked to sign up to the system before he/she starts using the application. The sign up page will ask the user to put his details, and as a result will be given a unique identification number which will be used to identify him uniquely and will also be assigned to the wash card that will be given to the user. All transactions inside the store will be through this wash card and it will be used to record the activities of the user in the filesystem, this card can be collected from the shop from an employee. In the sign up page the user will be asked to provide what type of a user he/she is (there are 5 types of users and they are Regular user, Silver user, Gold user, Regular employee and Manager). After the user has successfully signing up a user can use the credentials that he has used to sign up to login to the system. These details will be stored in two separate files, the username and password will be stored in one file (webuser.txt) which then will be used to verify the user, and all the records including the username and password will be stored in another file (records.txt) When the user enters to the system using the credentials that he has used to signup to the system earlier, the filesystem will check whether the user is an existing user, additionally it will check if the username and password match. If the user isn't an existing user, or if he has typed either his username or password incorrectly, then this user will not be able to gain access to the system and he will be asked to double check his entry or to sign up if he hasn't done yet. When the user finally meets the requirements above (signing up and then putting the same credentials to the login button), he will be directed to the home page of the app, where he will have the choice to buy a product, see and update if needed the information that was previously used to sign in, top up money to the wash card using a credit card. Additionally, there is a button where employees and managers can use to go a secure area. When a user clicks this button the system will automatically check if he has the authority to access this area and block access if he isn't authorized. If the user is authorized to access it then it will take the user to the privileged area where customer and purchase information can be accessed. When a regular employee enters this secure zone, he will be allowed to see the details of the customers but can't have additional access. However, when a manager accesses this area he can see the details of the customers, he can see details of the transactions that were made (which was stored in a separate file called products&prices.txt) and he can delete the users data if he want. An additional service that was required was to make the system recognize the type of user and give him/her discount accordingly (here managers will have the most discount rate which will be 50%, Employees will have 30% discount, and Gold, Silver, and Regular users will have 35%,25% and 15% discount respectively. Finally, the user can click the exit button at the top of the screen and the app will close immediately, however his details will be stored in a secure file which will be used when the user comes back at a later time.

Resources: https://snort.org/

Section 2 - Design




staffOnly
root:Tk()
homepage: homePage()
+ ShowAll()
+ DeleteAll()
+ Purchases()
userDetailsPage
root:Tk()
homepage: homePage()
+ update()
topUpPage
root:Tk()
homepage: homePage()
+ topup()
homePage
root:Tk()
homepage: LoginPage()
+ changePage()
+ gototopupPage()
+ gotouserDetailsPage()
+ gotostaffOnlyPage()
LoginPage
root:Tk()
+ userhome()
+ signuphome()
userSignupPage
root:Tk()
homepage: LoginPag
+ signup()

Section 3 – Plan

Step1= Run the program by creating an instance of LoginPage in the Test module and giving it root as a parameter.

Step 2: User selects Login or Sign up.

Step 3: Assuming that the user is coming for the first time he will go to SignupPage by clicking Sign up button.

Step 4: The user will fill all his details in the SignupPage and will click the Signup button which will put all his details in a separate file named records.txt and his username and password in another file named webuser.txt.

Step 5: The program exits and then the user is taken to the LoginPage when he starts from the beginning.

Step 6: The user is asked to put his username and password, if he puts the correct username and password he will be taken to the home page

Step 7: If he didn't put the correct username and password he will be asked to put the username and password correctly or signup if he hasn't already.

Step 8: In the Hope page the user can select items to buy and buy them by clicking Pay button, if he doesn't have enough balance in his wash card, he will asked to put money in his wash card which can be done buy pressing the top up button or he can top up from the cash machine found in the shop

Step 9: The user can also go to his details page by pressing userinfoPage button

Step 10: The user can top up money using the top up button which will take him to topupPage where he will put his credit card details and the amount of money he wants to put and this activity will be reflected in his record in the records.txt file

Step 11: If the user who has just joined is an employee or a manager they will have a staff only button which can be used to go to securePage only for employees. if the employee goes to the page to the page he can access the details of the customers and if the manager goes to the page he can see the details, see the transactions and delete all the data of the customers

Step 12: All pages will have back button which can be used to go back to the previous page

Step 13: Finally the user can click on the exit button to close the Application.

Section 4 Implementation and Test Cases

The program has three modules

1. CleanCarSystem Module

- It will Contain the system classes with their methods

- CleanCarServices Class
- WashCard Class
- User Class
- Transaction Class
- Cleaning Service Class

2. GUI Module

- It will contains the GUI pages classes with their methods

- LoginPage class
- userSignup class
- homePage Class
- userinfoPage Class
- StaffonlyPage Class
- topUpPage Class

3. Test Module

- A module where all the test instance is created

Section 5: Self-reflection

Generally, this project was a great experience for us. It gave us a great satisfaction to build a program that has many functions in which can be applied in real life. Before, we did this project we used to wonder how the skilled programmers build a software or an application and how do they manage the huge lines of codes. However, this project helped to find answers for our questions. A software or application is a big system made up of different small programs that step by step and little by little produce a huge multifunctional system.

This project helped us to implement all the things that you have learned in our Object-Oriented main course, starting from how to build class, how to set its attributes and behaviors, how to differentiate between the different type of relations, and how to decide the relations between the classes like inheritance, association, aggregation, and composition.

This project was challenging for us and we also face some obstacles. The first difficulty is that we had a limited knowledge in using and implementing the Tkinter Module. In the class we only learned the vey basic knowledge of using the GUI module. As a result, we did our research by ourselves to further extend our knowledge on using the Tkinter GUI module and to implement it wisely in our project. The second difficulty was in handling the file system of the program. Writing the code for the file system to us a longer time than the other parts of the program as most of the functions are related to the file system and it need to be updated and arranged every time the user makes a new decision. However, we managed to to think and be creative in order to find solution for our difficulty which was a great practice for our problem-solving skills.

For every member to understand the project of have the same point of view, first we discussed the problem, what are expected to do, how are we going to build the program. Then, we divide the task among us. The following are the contribution of each member of the group:

1) Contribution of **Natnael Zerai - M8007968**:
   The main role of Natnael was building the Main_car_wash system program. He built the module program that has all the classes and functions that allow the Car Wash project to make different tasks like payment, purchase, top up, updating details, registration, storing the information on files, etc. Besides, Natnael, helped in designing the ULM of the program.

2) Contribution of **Aymen Zekeria - M8007954**:
   The main role of Aymen was building the GUI module of the program. He built different GUI pages like login page, signup page, home page, special privileges page, etc. in which the user can easily interact with. Moreover, Aymen also contributed on designing the UML of the program.
3) Contribution of **Salem Maktoom -**:
   Salem designed the ULM of the program.

Resources: https://snort.org/