ST2414: Programming in Security

Assignment: CA2 External Documentation

Name: Seow Wee Hau Winston

1. Purpose Of Program

The purpose of this program is to function as an Electronic Service Provider (ESP) for registered or new users to purchase services provided by “Winston ESP” as well as for admins to perform administrative functions to maintain the smooth running of the ESP.

2. Features

2.1 Basic Features

Basic Features For This Program Include:

1. Build upon what was done in Assignment 1 – the Electronic Service & Protection (ESP) System.
2. Implement the ESP system using sockets, one to act as the server (file system) and the other to serve as the client (front end).
3. The client socket would be an extension of Assignment 1, where the services offered is displayed to the user and the user may place his subscription orders.
4. Each subscription has datestamps indicating start and end of subscriptions.
5. The server socket will supply input files for the users’ current subscriptions, including end dates.
6. The client will read the files from the server to determine the end date for services currently subscribed.
7. The system can then proceed to take subscription orders and payment as with Assignment 1.
8. Finally, send subscription length back to server and write back to user files.
9. Ensure that there are printout at your server end to indicate communication with the client.

2.2 Advanced Features

Advanced Features For This Program Include:

1. **Administrative system** - For staff to update subscription offered and send back to the server.
2. **Administrative system** - For staff to reset user password when requested by the user.
3. **Administrative system** - For staff to delete user account when requested by the user.
4. **User system** - Saving of carts for future use.
5. **User system** - Implementing discount system for specified users.
6. **User system** - Better output display than what is given in the examples.
7. **User system** – Services that have been added more than one, can be computed together to get more than a year of subscription. Services that expired will also be updated accordingly.

3. Folder For Input/Output Files

All files are located inside the **/TxtFiles** subfolder.

The files include:

1. “discounts.txt”
2. “invoices.txt”
3. “orderInCart.txt”
4. “passwd.txt”
5. “services.txt”
6. “subscriptions.txt”
7. “usersForgotPassword.txt”
8. “usersPendingDeletion.txt”
9. “winstonBusinessInfo.txt”

4. Accounts

4.1 User Accounts

Username	Password
Betty	Betty
Carl	Carl
Donovan	Donovan
Edward	Edward
Faith	123456789
Winston	Winston
Ryan	Ryan

NOTE:

- Account “Winston”: Used to test the reset password function for admin.
- Account “Ryan”: Used to test the delete user function for admin.

User	Current Subscriptions	Discounts Provided
Betty	**Firewall Services:** 1 **Security Ops Centre:** 3 **Hot Site:** 3 **Data Protection:** 4	**Firewall Services:** 0.7 **Security Ops Centre:** 0.3 **Hot Site:** 0.4 **Data Protection:** 0.5
Carl	**Firewall Services:** 1 **Security Ops Centre:** 2	**Firewall Services:** 0.1 **Security Ops Centre:** 0.2 **Hot Site:** 0.3 **Data Protection:** 0.4
Donovan	**Firewall Services:** 2 **Security Ops Centre:** 1 **Hot Site:** 1 **Data Protection:** 1	-
Edward	-	-
Faith	**Firewall Services:** 3 **Security Ops Centre:** 1 **Hot Site:** 1 **Data Protection:** 1	**Firewall Services:** 0.6
Winston	**Firewall Services:** 1	**Firewall Services:** 0.7 **Data Protection:** 0.5
Ryan	**Firewall Services:** 1 **Data Protection:** 1	-

4.2 Admin Account

Username	Password
Admin	Admin

5. Input/Output Files

5.1 Main Program Files

Program files used by either client or server:

1. userClient.py
    - File contains program for registered and new users to run to use the ESP.
2. generateNewUser.py
    - File contains functions to create a new user when requested from the "generalServer.py" file.
3. userCreation.py
    - File contains functions to generate a user's functions when operating the "userClient.py" file.
4. adminClient.py
    - File contains program for admins to run to use the ESP.
5. generalServer.py
    - File contains the server which will receive input from either clients, user or admin ("userClient.py" or "adminClient.py") and execute functions depending on the data received.

5.2 Folder "/Txtfiles" Files

Text files which hold the ESP information required to run the program:

1. "discounts.txt"
    - File contains all discounts for each user. Edited only by administrator.
    - Contents Of File:
        - "#" is for comments or service key to set the discounts dictionary.
        - Service keys are the names of each service, after "#" Discounts For:"
        - Each user has a new discount per service in each line and they contain the service value for the service key.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Discounts For:
# Firewall Services
# Security Ops Centre
# Hot Site
# Data Protection
User 1: Betty
0.2
0.3
0.4
0.5
User 2: Carl
0.1
0.2
0.3
0.4
```

2. "invoices.txt"
    - File contains all invoices that users have ordered. Only items that have been ordered and paid will be here.
    - Contents Of File:
        - "#" is for comments.
        - First, invoice number.
        - Second, username.
        - Third, date when service(s) was ordered.
        - Fourth, all service name and amount ordered.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Order(s) that has been made:
# Legend:
# Invoice Number
# Username
# Date Ordered
# Rest Of Lines: Services and Cost (Each In A Different Line)
Invoice No: I1
Betty
2019-11-27
Firewall Services
1
Security Ops Centre
3
Hot Site
2
Invoice No: I2
Carl
2020-11-02
Firewall Services
1
Security Ops Centre
2
```

3. “orderInCart.txt”
    - File contains all orders that users have placed in the cart. Only items that are in the cart and not ordered before the user leaves will be here.
    - Contents Of File:
        - “#” is for comments.
        - First, order number for reference.
        - Second, username.
        - Third, all service names and their quantity.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Order(s) That Failed To Be Purchased/ Continue Order At Later Date:
# Legend:
# Order Number
# Username
# Rest Of Lines: Services and Cost (Each In A Different Line)
Order No: A1
Betty
firewall services
3
security ops centre
1
hot site
1
data protection
2
```

4. “passwd.txt”
    - File contains all usernames and encrypted passwords for admin and all registered users. Edited only by administrator.
    - Contents Of File:
        - “#” is for comments.
        - Firstly, username.
        - Secondly, encrypted password.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Username && Password:
# Legend:
# Username
# Encrypted Password
User 1: Admin
nimdA
User 2: Betty
ytteB
User 3: Carl
lraC
User 4: Donovan
navonoD
User 5: Edward
drawdE
```

5. “services.txt”
    - File contains all services provided by the ESP. Edited only by administrator.
    - Contents Of File:
        - “#” is for comments.
        - Firstly, service name.
        - Secondly, service cost.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Services:
# Legend:
# Service Name
# Cost Of Service
firewall services
1200
security ops centre
4200
hot site
8500
data protection
10000

```

6. “subscriptions.txt”
    - File contains all subscriptions for each user. Edited only by administrator.
    - Contents Of File:
        - “#” is for comments or service key for the subscriptions dictionary.
        - Services keys are the name of each service after the “# Legend:”
        - Each user has an expiration date for that service as well as subscription amount per service in each line which contain the amount of subscriptions they have ordered for that service.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Current Total Number Of Subscriptions Of Users:
# Legend:
# Firewall Services
# Security Ops Centre
# Hot Site
# Data Protection
User 1: Betty
2023-01-30
3
2022-11-26
3
2023-11-26
5
2023-11-26
4
User 2: Carl
```

7. “usersForgotPassword.txt’
    - File contains all names of users that have requested for a password reset.
    - Contents Of File:
        - “#” is for comments.
        - Contains all usernames of users that have requested for a password reset.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Users That Requested A Reset On Their Account:
# Legend:
# Usernames That Have Requested A Reset
Winston

```

8. “usersPendingDeletion.txt”
    - File contains all names of users that have pending deletion of their account.
    - Contents Of File:
        - “#” is for comments.
        - Contains all usernames of users that are pending for their deletion of account.

```
# ::DO NOT EDIT THIS FILE DIRECTLY::
# Users That Requested Deletion Of Their Account:
# Legend:
# Usernames That Are Pending Deletion
Ryan

```

9. “winstonBusinessInfo.txt”
    - File contains all information regarding Winston (E)Service Provider [ESP].
    - Contents Of File:
        - “#” is for comments.
        - Firstly, business name.
        - Secondly, business location.

```
# ::Take Note Of Lines::
# First Line: Business Name
# Second Line: Business Location
Winston's (E)Service Provider
Singapore 123456
```

6. Limitations

Limitations brought over from CA1:

1. All .txt files except for “winstonBusinessInfo.txt” cannot be written directly, only either modified by admin or not modified at all.
    - Files Modified By Administrator Through Program:
        - services.txt
        - passwd.txt
        - discounts.txt
        - subscriptions.txt
        - orderInCart.txt
        - usersForgotPassword.txt
        - usersPendingDeletion.txt
    - File That Are Not Used In The Administrator Program But Can Be Modified:
        - winstonBusinessInfo.txt
    - File That Are Not Used In The Administrator Program And Cannot Be Modified:
        - invoices.txt
2. New User’s names cannot be only integers or have spaces in it.
    - **EXAMPLE:** “123” is not allowed. “Winston123” is also not allowed.
    - **EXAMPLE:** “Winston Seow” is not allowed. “WinstonSeow” is allowed.
3. Users must remember the cart order number if not, that cart will not be able to be taken out of the “ordersInCart.txt” file.
    - **EXAMPLE:** “A5” is the cart order number, but the user forgot and could not type the cart order number.

Limitations in CA2:

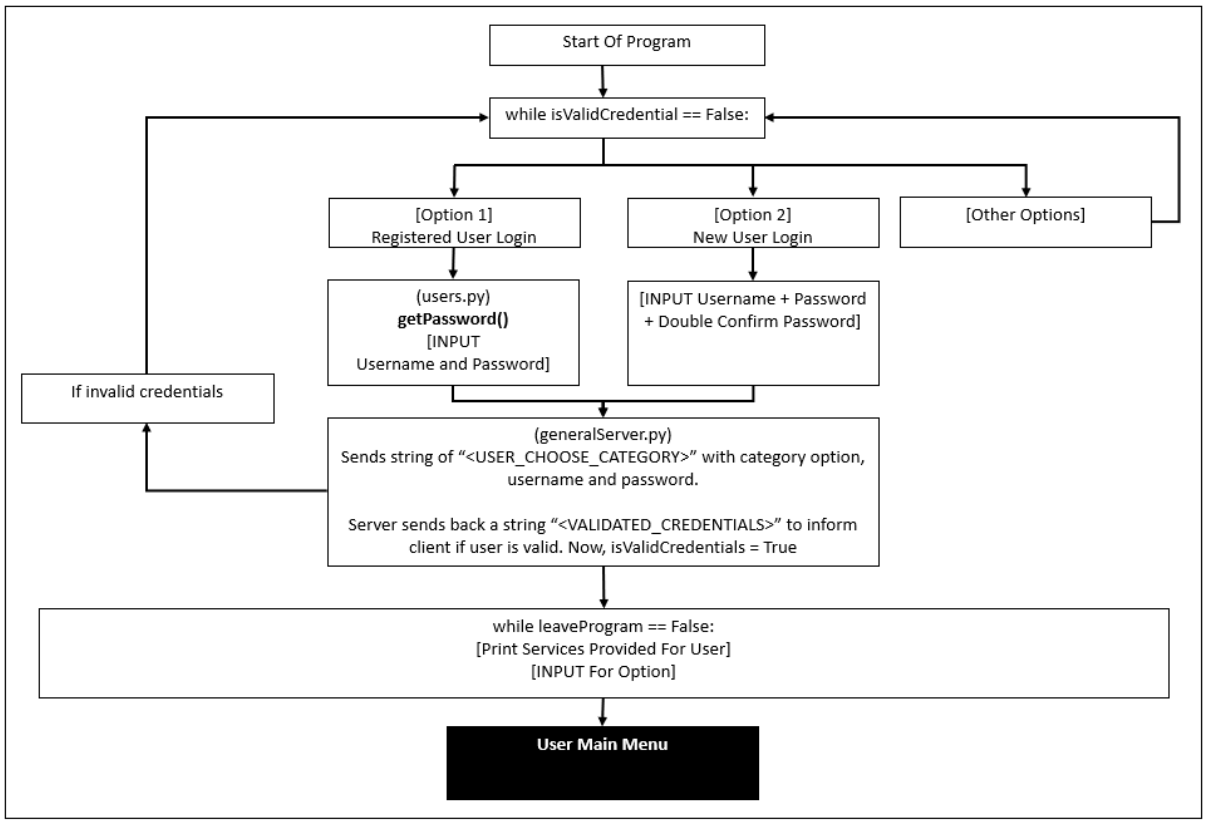
1. The inputs given by the client should not contain a “,” which would cause errors in the program.
2. Service names should not be longer than 20 characters to ensure that it is readable in the prompt.

7. Flowchart

7.1 Client Program

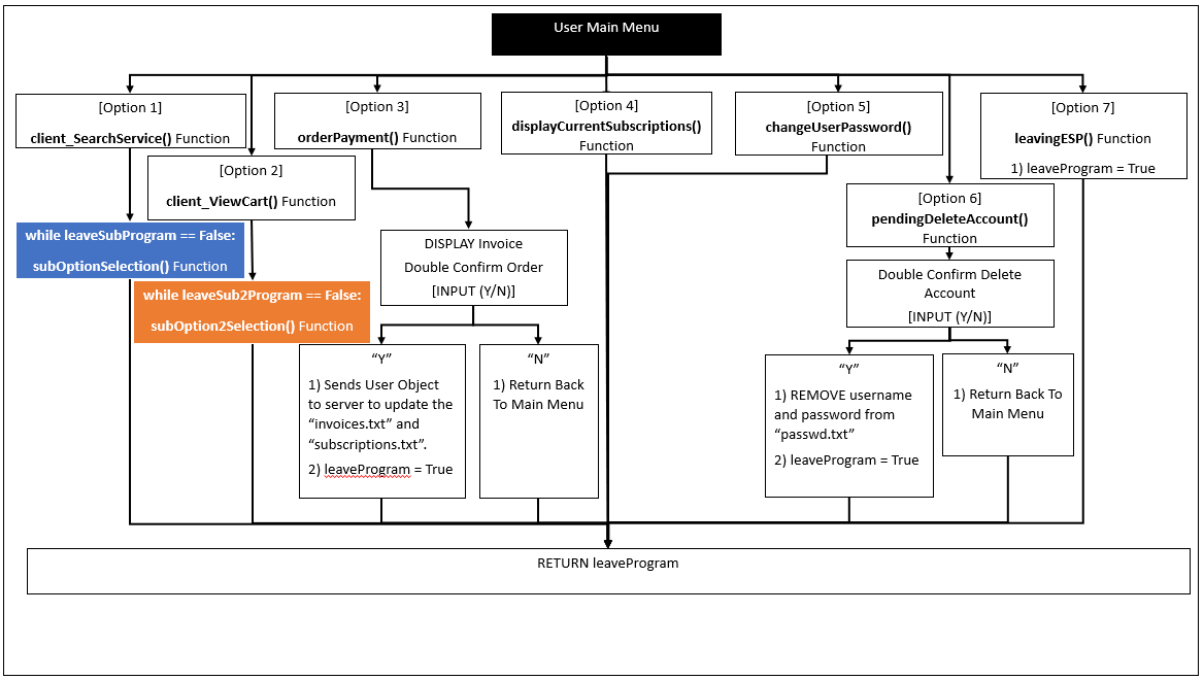
The client program will be executed using the “userClient.py” file.

“userClient.py” File Part 1 – **Validate User**:



- **getPassword()** Function
  - For each character user inputs, it converts it into an “*”. If the input is “ENTER”, to finish the password, it breaks the loop.
- **NOTE:** IF user password from registered user is left empty, the user can request for a reset password, with a simple yes or no confirmation.
- “<USER_CHOOSE_CATEGORY>”
  - This string is sent to the server to inform the server that the data sent with it is to validate the credentials of the user.

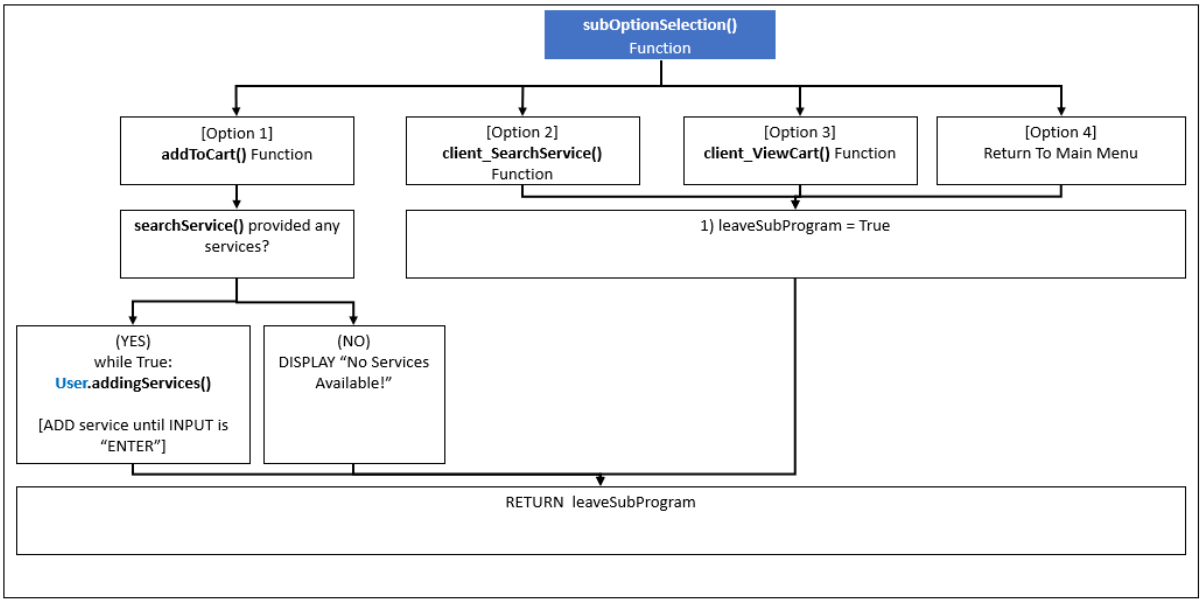
"userClient.py" File Part 2 – **User Main Menu**:



- **client_SearchService()** Function [Main Menu Option 1]
  - INPUT a keyword provided by the user, this keyword is then appended to a string "<DISPLAY_LISTOFSERVICES>".
  - SEND this string to the server to search for available services.
  - RECEIVE a new string of the names of available services, from server, which is made into an array.
  - **subOptionSelection()** Function – Will be going into detail in the following pages.
- **client_ViewCart()** Function [Main Menu Option 2]
  - DISPLAYS all items which the user ordered.
  - SEND name of service to server so that client RECEIVES cost of each service from server, through a string "<USER_GETPRICEOFSERVICE>".
  - **subOption2Selection()** Function – Will be going into detail in the following pages.
- **orderPayment()** Function [Main Menu Option 3]
  - IF current cart does not contain items, DISPLAY "No Services Added" and RETURN back to the main menu.
  - IF current cart contains items, checks if user has viewed cart yet (**client_ViewCart()**).
    - IF have not viewed cart,
      - SEND name of service to server so that client RECEIVES cost of each service from server, through a string "<USER_GETPRICEOFSERVICE>".
    - SEND a string "<USER_GETBUSINESSINFO-->" to server so that client RECEIVES business information and invoice number from server.
    - CALCULATE new expiration date for each added service.
    - DISPLAY invoice.
    - Double confirm whether to pay.
      - IF "Y", SEND the **User** object to server.
      - IF "N", RETURN to main menu.

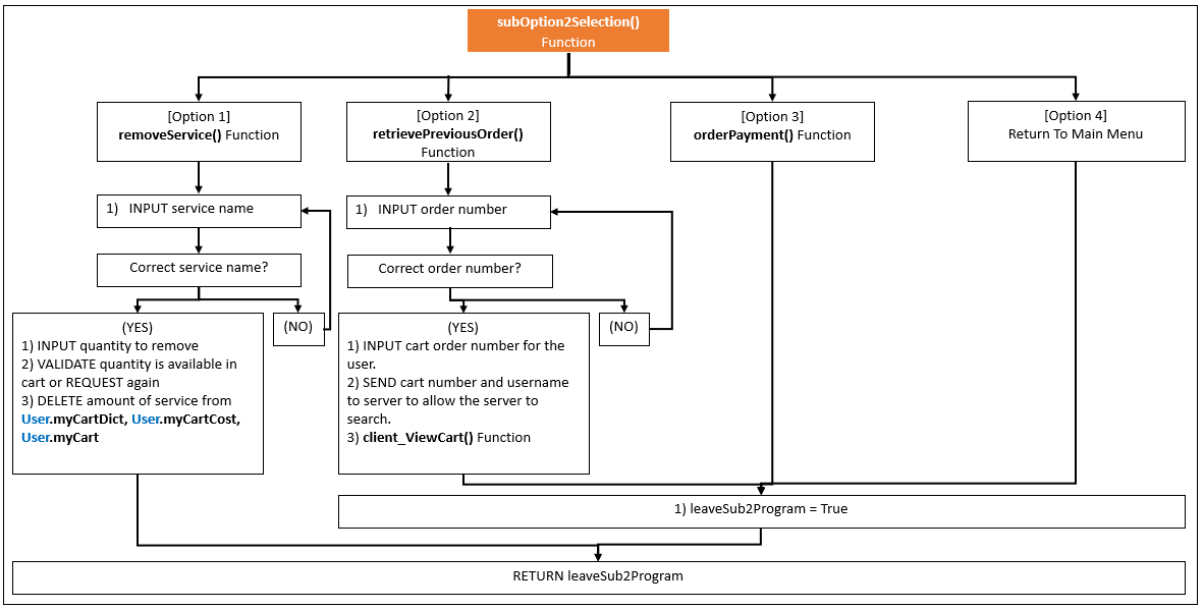
- **displayCurrentSubscriptions()** Function [Main Menu Option 4]
    - DISPLAY all subscriptions start and end dates.
        - Start dates are determined from the end dates as well as the number of subscriptions.
- **changeUserPassword()** Function [Main Menu Option 5]
    - INPUT new password twice.
    - If both passwords are the same:
        - SEND a string "<USER_CHANGEPASSWORD--->" with the new password appended to it.
    - If both passwords are not the same:
        - REINPUT the new password twice.
- **pendingDeletionAccount()** Function [Main Menu Option 6]
    - DISPLAY terms of service to inform users of the consequence of deleting account.
    - INPUT double confirmation of user.
        - IF "Y", SEND a string "<DELETED_ACCOUNT>" with username of the user appended to it. Leave entire ESP program.
        - IF "N", RETURN to main menu.
- **leavingESP()** Function [Main Menu Option 7]
    - IF there are still items in the cart,
        - DISPLAY "There are still item(s) in the cart".
        - INPUT an option whether the user wants to save cart.
            - IF "Y", SEND a string "<USER_SAVECART--------->" with **User** object to server.
            - IF "N", discard cart.
        - Leave entire ESP program.

“userClient.py” File Part 3 – **subOptionSelection:**



- **addToCart()** Function
    - Depending on the number of services,
        - IF number of services displayed is 0, **addToCart()** denies entry.
        - IF number of services displayed is more than 0, start a while loop.
            - ❖ Each service name has a digit beside it which the user can input.
            - ❖ VALID integer is than added to the cart using User**.addingServices()** Function.

"userClient.py" File Part 4 – **subOption2Selection:**

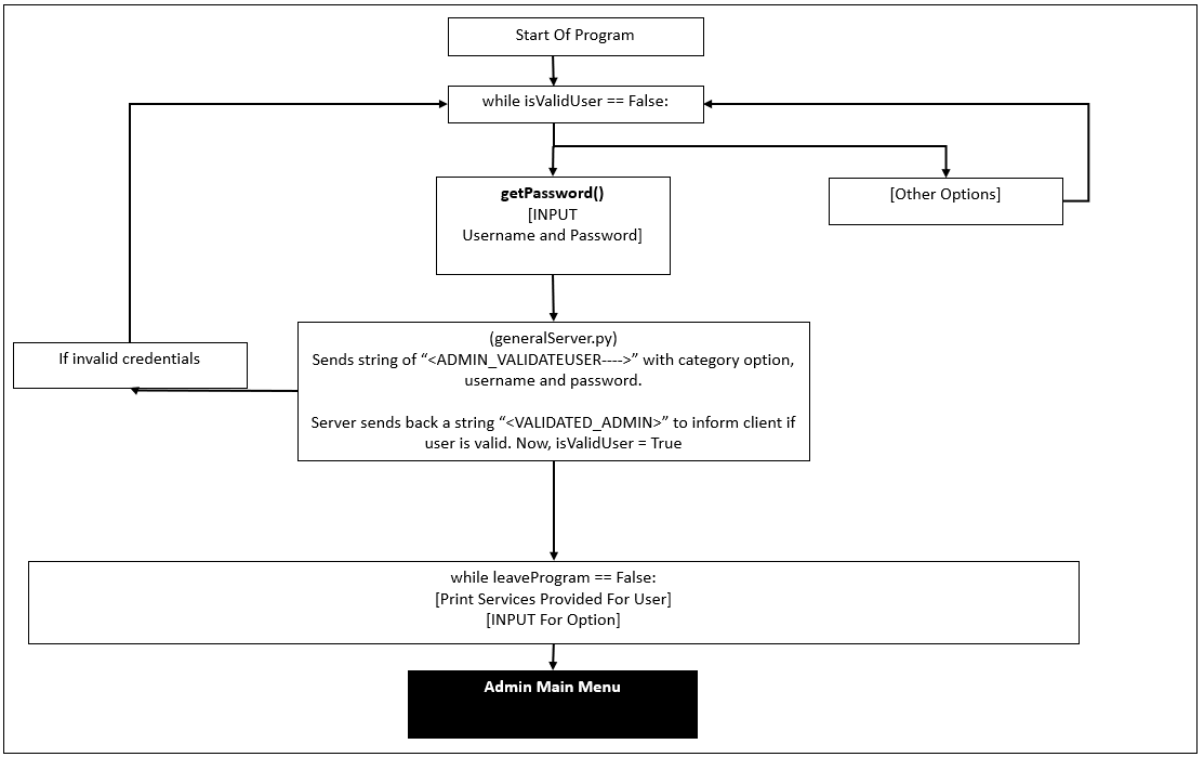


- **removeService()** Function
  - INPUT name of service to remove.
  - IF name of service is valid,
    - INPUT amount of service to remove.
      - IF amount of service to remove is valid, remove amount.
      - IF amount of service to remove is not valid, REINPUT amount of service to remove.
- **retrievePreviousOrder()** Function
  - INPUT cart number that was previously given.
  - SEND a string "<USER_GETPREVIOUSORDER->" and the cart number and username is appended to the end of the string, RECEIVE data from server.
    - IF data contains "<NOORDER>", then DISPLAY "Sorry, this order does not exist".
    - Else, collect the order and add it into the cart.

7.2 Admin Program

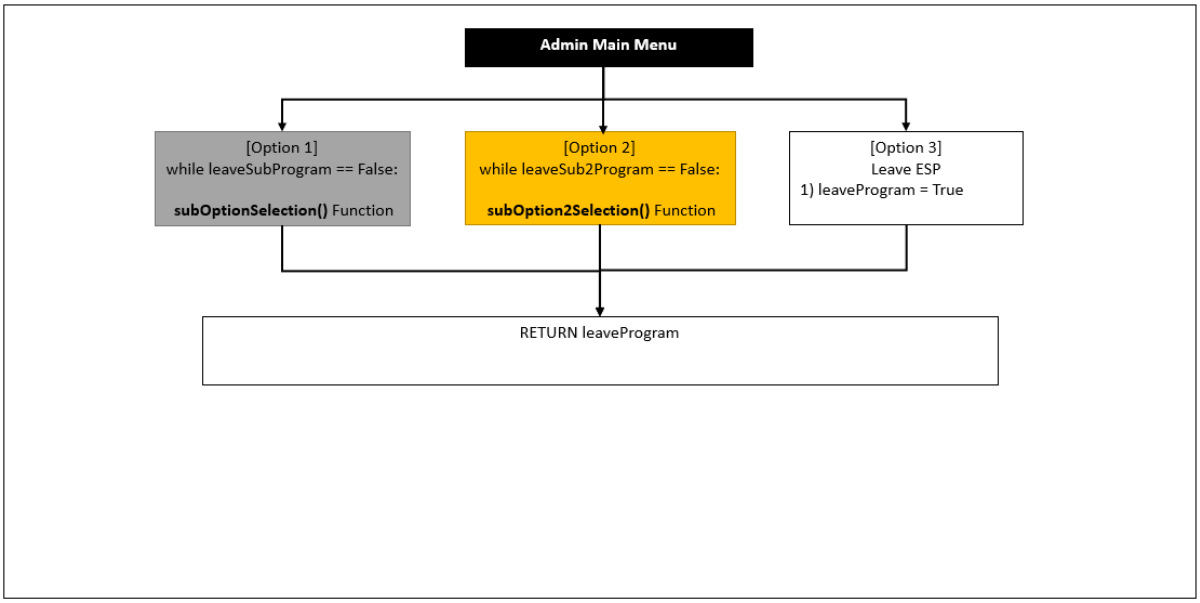
The admin program will be executed using the “adminClient.py” file.

“adminClient.py” File Part 1 – **Validate Admin:**



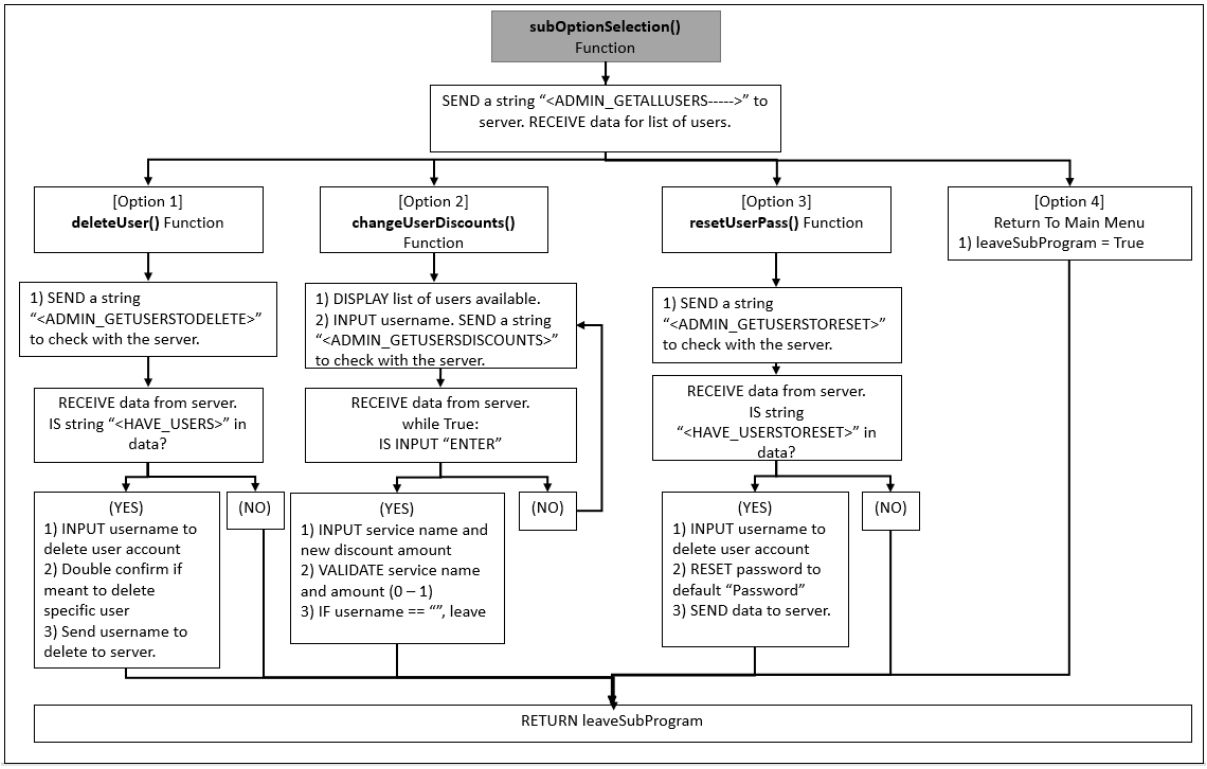
- **getPassword()** Function
    - For each character user inputs, it converts it into an “*”. If the input is “ENTER”, to finish the password, it breaks the loop.
- INPUT username and password of admin before SEND to server a string “<ADMIN_VALIDATEUSER---->” with username and password appended to the end. RECEIVE data sent back from server.
    - IF data contains a string “<VALIDATED_ADMIN>”, isValidUser = True.
    - Else, DISPLAY “Invalid credentials.”

“adminClient.py” File Part 2 – **Admin Main Menu:**



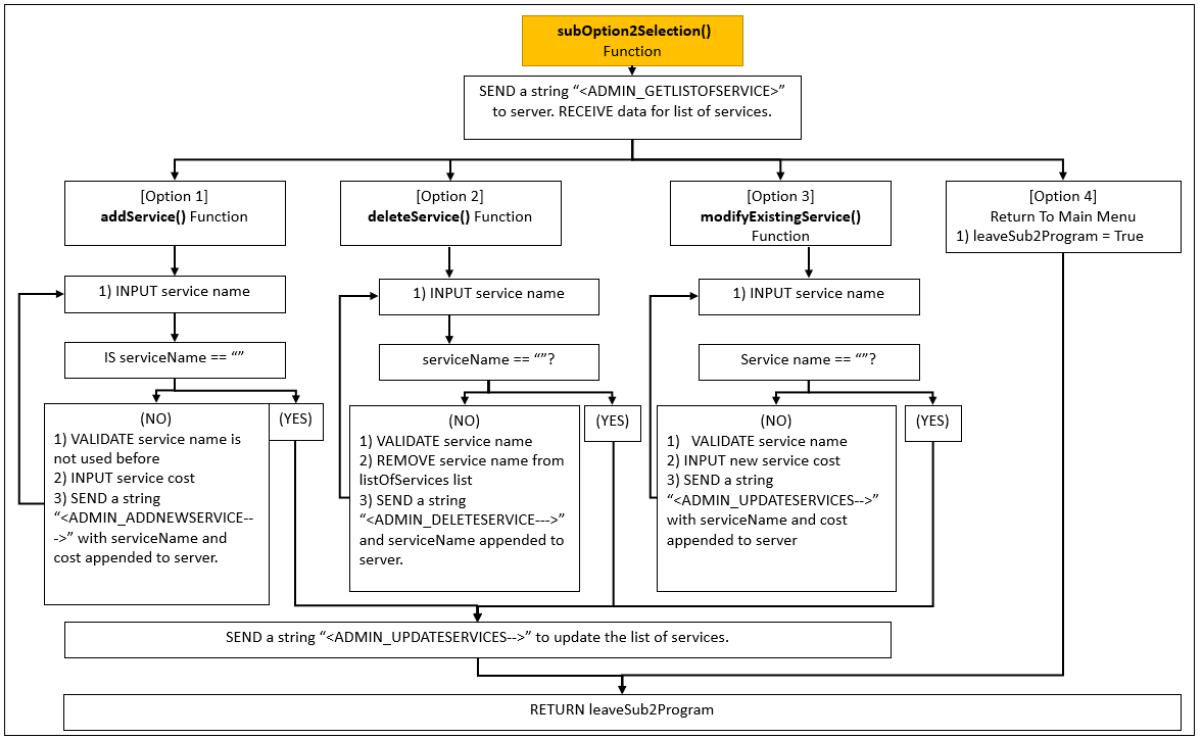
- **subOptionSelection()** Function
    - INPUT “1” for the admin to execute user-related functions.
- **subOption2Selection()** Function
    - INPUT “2” for the admin to execute service-related functions.

“adminClient.py” File Part 3 – **subOptionSelection:**



- **deleteUser()** Function
  - SEND a string “<ADMIN_GETUSERSTODELETE>” to server and RECEIVE data from server.
    - IF “<HAVE_USERS>” in data, DISPLAY users pending deletion. INPUT username to delete.
      - IF username is valid, SEND a string “<ADMIN_DELETEUSER------>” and username appended to it. RECEIVE “<USER_DELETED>” IF user has been deleted.
      - Else, leave the function.
    - Else, print “No users pending deletion”.
- **changeUserDiscounts()** Function
  - INPUT username to modify discounts.
  - IF username is valid:
    - SEND a string “<ADMIN_GETUSERDISCOUNTS>” and username appended to it. RECEIVE a dictionary of the discounts for that user.
    - While input IS NOT “ENTER”, input service name and cost.
  - Else, print “Invalid username”.
- **resetUserPass()** Function
  - SEND a string “<ADMIN_GETUSERSTORESET->” to server and RECEIVE data from server.
    - IF “<HAVE_USERSTORESET>” from server, DISPLAY all users pending reset password. INPUT username to delete.
      - IF username is valid, SEND a string “<ADMIN_RESETPASSWORD--->” and username appended to server.
    - IF “<NO_USERSTORESET-->” from server, print “No users pending reset password”.

“adminClient.py” File Part 4 – **subOption2Selection:**

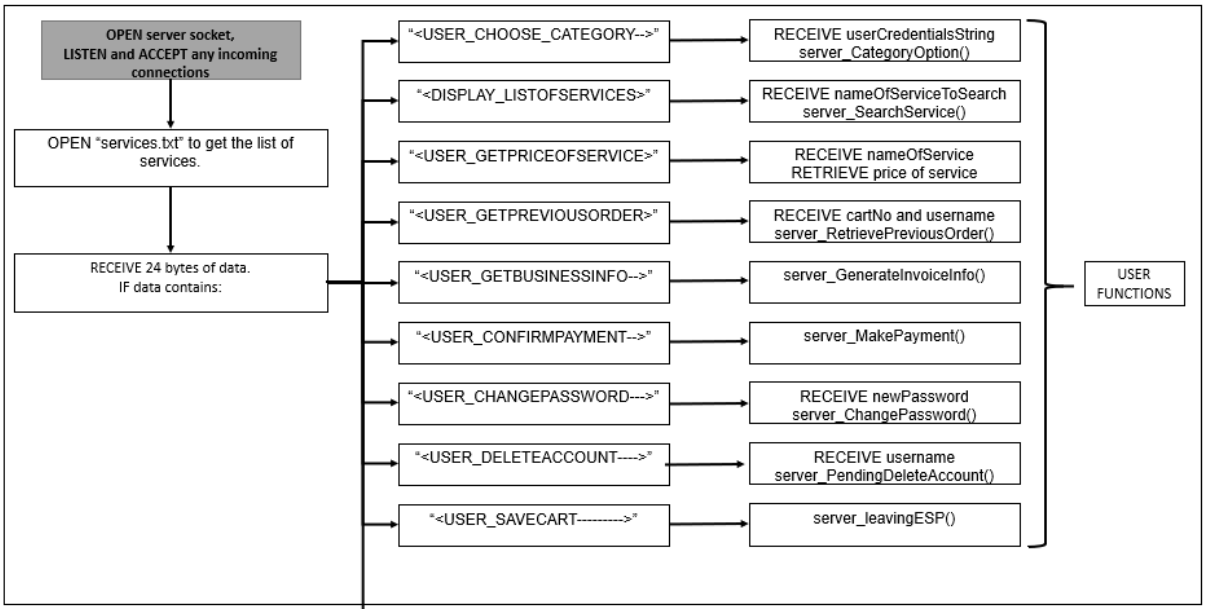


- **addService()** Function
  - INPUT new service name and cost.
  - While the service name is not “ENTER”,
    - IF new service name and cost IS VALID, send a string “<ADMIN_ADDNEWSERVICE--->” with service name and cost to server.
    - IF new service name or cost is INVALID, REINPUT a new name or cost.
- **deleteService()** Function
  - INPUT service name to delete.
  - While the service name is not “ENTER”,
    - IF service name is VALID, send a string “<ADMIN_DELETESERVICE--->” with service name to server.
    - IF service name is INVALID, REINPUT service name to delete.
- **modifyExistingService()** Function
  - INPUT service name and cost.
  - While the service name is not “ENTER”,
    - IF service name and cost is VALID, send a string “<ADMIN_UPDATESERVICES-->” with service name and cost to server.
    - IF service name or cost is INVALID, REINPUT a valid name or cost.

7.3 Server Program

The server program will be executed using the “generalServer.py” file.

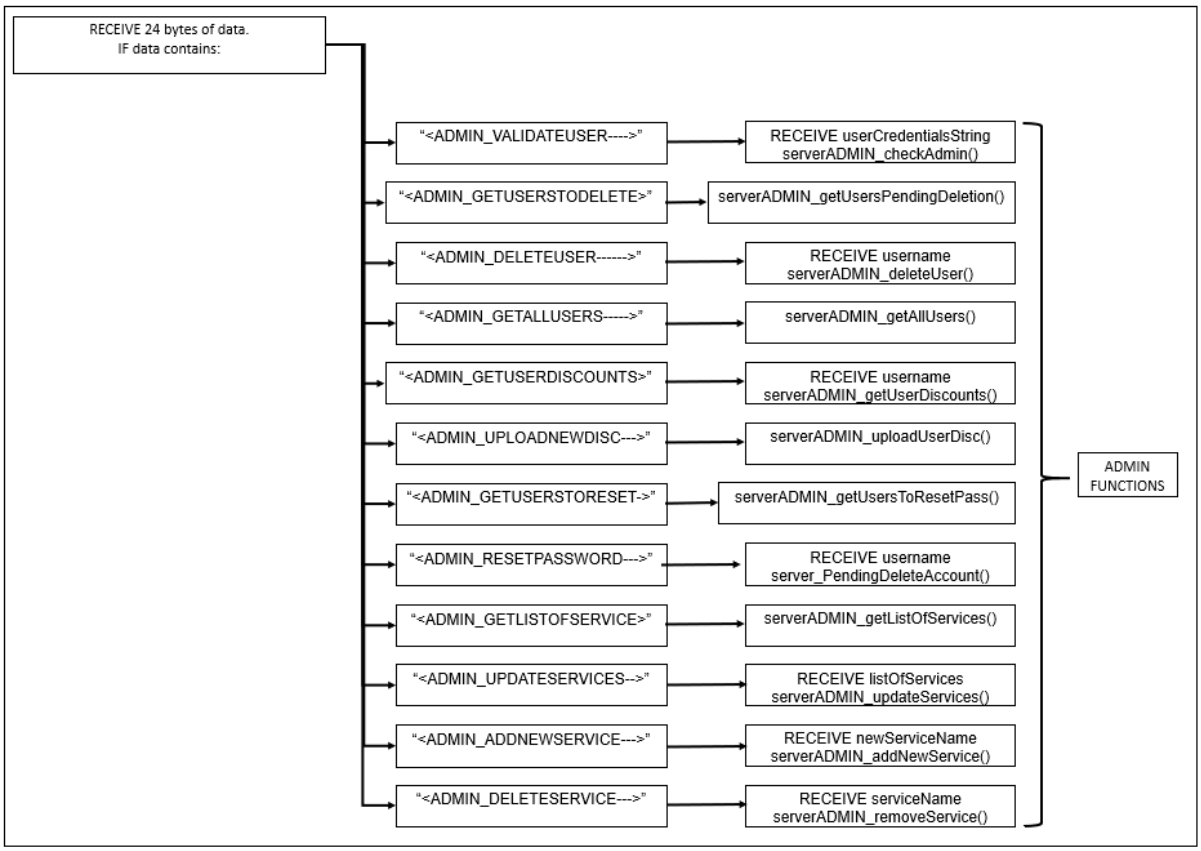
“generalServer.py” File Part 1 – **User Functions**



- **server_CategoryOption()** Function
  - The userCredentialsString is split into the category option, username, and password.
  - The **checkUser()** function is executed to check whether the credentials are validated.
    - IF valid user, check if it is from create new user.
      - IF new user, execute **createNewUser()** (from generateNewUser.py) to
      - SEND the **User** object to the client.
    - Else, SEND the reason for invalid credentials to the client.
- **server_SearchService()** Function
  - The nameOfServiceToSearch is used to search the listOfServices to find any available services.
    - IF there are any services, append it from an array into a string.
    - IF there are no services, send a “<NOAVAILABLESERVICES>”.
- **server_RetrievePreviousOrder()** Function
  - OPEN the “orderInCart.txt” file to search for the correct cart order and username.
    - IF there is the correct cart order with the correct username, the cart is retrieved and SENT to the client.
    - IF there is a correct cart order but for a different user, the cart is not retrieved.
- **server_GenerateInvoiceInfo()** Function
  - OPEN the “winstonBusinessInfo.txt” file to get the business’s name and location.
  - OPEN the “invoices.txt” file to get the newest invoice number.
  - SEND this information to the client to generate the invoice.

- **server_MakePayment()** Function
    - Receives the **User** object from the client and uses it to create a new invoice.
        - First, OPEN the “invoice.txt” file to add the new invoice.
        - Second, OPEN the “subscriptions.txt” file to update the new subscriptions.
        - Third, OPEN the “orderInCart.txt” to check if the cart is still inside. IF still inside, remove the cart order.
- **server_ChangePassword()** Function
    - OPEN the “passwd.txt” file to search for the username.
    - When found, update the “passwd.txt” to change the default password of “Password”.
- **server_PendingDeleteAccount()** Function
    - OPEN the “usersPendingDeletion.txt” to update the user using the username.
    - OPEN the “passwd.txt” to remove the credentials for that user.
- **server_leavingESP()** Function
    - RECEIVE the **User** object.
    - OPEN the “orderInCart.txt” file to add the cart order to the file.

“generalServer.py” File Part 2 – **Admin Functions**



NOTE: This is in the same IF, ELIF, ELSE statements as the USER FUNCTIONS in the previous two pages.

- **serverADMIN_checkAdmin()** Function
  - OPEN the “passwd.txt’ file to validate the credentials received from the client.
    - IF credentials are valid, send a “<VALIDATED_ADMIN>” to the client.
    - IF credentials are invalid, send a “<INVALID_CREDENTIALS>” to the client.
- **serverADMIN_getUsersPendingDeletion()** Function
  - OPEN the “usersPendingDeletion.txt” file to check if there are any users pending deletion.
    - IF there are users pending deletion, send a string “<HAVE_USERS>” with a string of the name of users to the client.
    - IF there are no users pending deletion, send a “<NO_USERS-->” string to the client.
- **serverADMIN_deleteUser()** Function
  - OPEN the “usersPendingDeletion.txt” file to remove the username of the user to delete.
  - OPEN the “discounts.txt” and “subscriptions.txt” files to remove the discounts and subscriptions for that user.
  - OPEN the “orderInCart.txt” file to remove any cart previously saved by that user.
  - SEND the client a “<USER_DELETED>” string to inform the client that the user has been deleted.

- **serverADMIN_getAllUsers()** Function
    - OPEN the “passwd.txt” file to get the list of users that are inside the ESP system.
    - SEND the list of users to the client.
- **serverADMIN_getUsersDiscounts()** Function
    - OPEN the “discounts.txt” file to grab the discounts for the username provided.
    - SEND the discounts of the user to the client.
- **serverADMIN_uploadUserDisc()** Function
    - RECEIVE the discounts sent by the client.
    - OPEN the “discounts.txt” file to update the discounts for that user.
    - SEND a string “<UPDATED_USERDISCOUNTS>” to client to inform of updated discounts.
- **serverADMIN_getUsersToResetPass()** Function
    - OPEN the “usersForgotPassword.txt” file to check if there are any users pending reset password.
        - IF there are users pending password reset, send a string “<HAVE_USERSTORESET>” with a string of the name of users to the client.
        - IF there are no users pending password reset, send a string “<NO_USERSTORESET-->” to client.
- **serverADMIN_resetPassword()** Function
    - OPEN the “passwd.txt” file to search for the user and the previous password. Afterwards, change the password to the default password of “Password”.
    - OPEN the “usersForgotPassword.txt” file to remove the username from the file.
- **serverADMIN_getListOfServices()** Function
    - OPEN the “services.txt” file to get all the list of services provided by the ESP.
    - SEND the list of services to the client.
- **serverADMIN_updateServices()** Function
    - OPEN the “services.txt” file to update the list of services provided by the ESP.
    - SEND a string “<UPDATED_LISTOFSERVICES>” to the client to inform the client that the services has been updated.
- **serverADMIN_addNewService()** Function
    - OPEN both the “discounts.txt” and “subscriptions.txt” file to create a new section for each service.
    - SEND a string “<UPDATED_DISC_SUBS>” to the client.
- **serverADMIN_removeService()** Function
    - OPEN both the “discounts.txt” and “subscriptions.txt” file to remove the section for the service which is being removed.
    - SEND a string “<UPDATED_DISC_SUBS>” to the client.

8. Sample Runs

8.1 User Program

This sample run will be simulated as a registered user of the program.
NOTE: This sample run is through registered user “Winston”.

Client Output [userClient.py]	**Server Output** [generalServer.py]

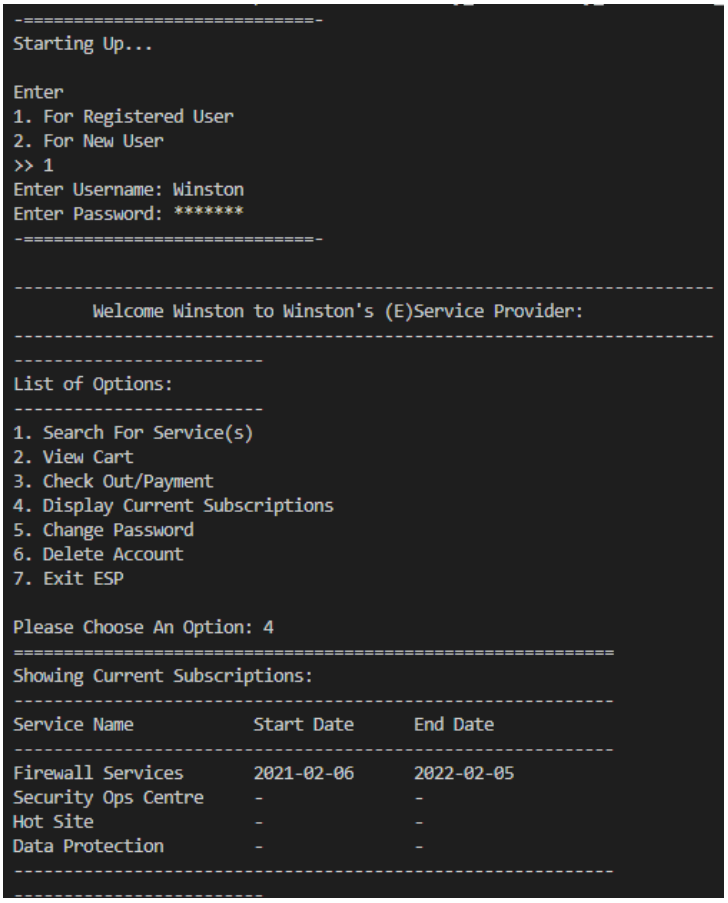
Scenario 1: Buying Service From ESP
1) Entering The Credentials
`-==========-`
`Starting Up...`
 	
`Enter`
`1. For Registered User`
`2. For New User`
`>> 1`
`Enter Username: Winston`
`Enter Password: *******`
`-==========-`
	1) Validate Credentials
	`Connected to: 127.0.0.1:60375`
	`Thread Number: 2`
	`Active threads: 0`
	`Received Category Choice`
	`Processing...`
	`Category Option 1: Registered User...`
	`Valid Credentials Sent By Client.`
	`Closed connection for: 127.0.0.1:60375`

2) Main Menu
`Welcome Winston to Winston's (E)Service Provider:`
`List of Options:`
`1. Search For Service(s)`
`2. View Cart`
`3. Check Out/Payment`
`4. Display Current Subscriptions`
`5. Change Password`
`6. Delete Account`
`7. Exit ESP`
 	
`Please Choose An Option:`
	2) **<NO OUTPUT>**

3) Search Service *[Main Menu Option 1[*
`Please Choose An Option: 1`
`Please Enter Name Of Service: fire`
`==========`
`Service(s) Found`
`==========`
`1. Firewall Services`
 	
`=========`
`Suboption Menu 1:`
`=========`
`1. Add To Cart (No.)`
`2. Search Again`
`3. View Cart`
`4. Return To Main Menu`
 	
`Enter A Number:`
	3) Search For Available Services
	`Connected to: 127.0.0.1:60447`
	`Thread Number: 3`
	`Active threads: 0`
	`Main Menu Option 1 Selected: Search For Service`
	`Closed connection for: 127.0.0.1:60447`

Client Output [userClient.py]	**Server Output** [generalServer.py]
4) Add To Cart =--------------= Suboption Menu 1: =--------------= 1. Add To Cart (No.) 2. Search Again 3. View Cart 4. Return To Main Menu Enter A Number: 1 Processing Input... ========== List Of Service(s) Available: ---------- 1. Firewall Services ---------- Enter The Number Of The Service or ENTER to stop: 1 ---------- Added Services: 1. Firewall Services ========== ========== List Of Service(s) Available: ---------- 1. Firewall Services ---------- Enter The Number Of The Service or ENTER to stop: 1 ---------- Added Services: 1. Firewall Services 2. Firewall Services ========== ========== List Of Service(s) Available: ---------- 1. Firewall Services ---------- Enter The Number Of The Service or ENTER to stop: Finish Adding Services... =--------------= Suboption Menu 1: =--------------= 1. Add To Cart (No.) 2. Search Again 3. View Cart 4. Return To Main Menu Enter A Number:	4) **<NO OUTPUT>**
5) View Cart *[Main Menu Option 2]* =--------------= Suboption Menu 1: =--------------= 1. Add To Cart (No.) 2. Search Again 3. View Cart 4. Return To Main Menu Enter A Number: 3 Processing Input... Grabbing Cart... ----------=Current Cart:=---------- Name Of Service: Quantity Discount Cost Firewall Services 2 0.0% 2400.00 Total Cost: $2400.00 ---------- =--------------= Suboption Menu 2: =--------------= 1. Remove A Service 2. Retrieve Order From Previous Session 3. Check Out/Payment 4. Return To Main Menu ========== Enter An Suboption No.:	5) Get Price Of Each Service Connected to: 127.0.0.1:60493 Thread Number: 4 Active threads: 0 Get Price For Service - Firewall Services Closed connection for: 127.0.0.1:60493

Client Output [userClient.py]	**Server Output** [generalServer.py]
6) Remove A Service =--------------= Suboption Menu 2: =--------------= 1. Remove A Service 2. Retrieve Order From Previous Session 3. Check Out/Payment 4. Return To Main Menu ========== Enter An Suboption No.: 1 Processing Input... ---------- Remove Service: Enter Service Name (To Be Removed): firewall services Processing Input... Enter Quantity For Service (To Be Removed): 1 Loading Cart... ----------=Current Cart:=---------- Name Of Service: Quantity Firewall Services 1 ----------	6) **<NO OUTPUT>**
7) Check Out/Payment *[Main Menu Option 3]* =--------------= Suboption Menu 2: =--------------= 1. Remove A Service 2. Retrieve Order From Previous Session 3. Check Out/Payment 4. Return To Main Menu ========== Enter An Suboption No.: 3 Processing Input... ========== ---------- Business Name: Winston's (E)Service Provider Business Location: 500 Dover Rd Invoice Number: I8 User: Winston Date: 2021-02-06 ---------- Service Name: Quantity: Cost: Expiration Date: Firewall Services 1 1200.00 2022-02-05 ---------- Total Cost: $1284.00 *Inclusive Of GST ========== Confirm Order? (Y/N):	7) Grab Business Information Connected to: 127.0.0.1:60620 Thread Number: 5 Active threads: 0 Main Menu Option 3 Selected: Grab Businesss Information Closed connection for: 127.0.0.1:60620
8) Confirm Payment ---------- Business Name: Winston's (E)Service Provider Business Location: 500 Dover Rd Invoice Number: I8 User: Winston Date: 2021-02-06 ---------- Service Name: Quantity: Cost: Expiration Date: Firewall Services 1 1200.00 2022-02-05 ---------- Total Cost: $1284.00 *Inclusive Of GST ========== Confirm Order? (Y/N):Y Order Made. An Invoice Will Be Sent To You Shortly. Thank You For Shopping With Winston's (E)Service Provider!	8) Retrieve User Object Connected to: 127.0.0.1:54760 Thread Number: 6 Active threads: 0 Main Menu Option 3 Selected: Make Payment User Object Retrieved. Closed connection for: 127.0.0.1:54760

Client Output [userClient.py]	**Server Output** [generalServer.py]
9) Logging Back In, View Current Subscriptions *[Main Menu Option 4]* 	9) **<NO OUTPUT>**

```
-----------------------------
Starting Up...

Enter
1. For Registered User
2. For New User
>> 1
Enter Username: Winston
Enter Password: *******
-----------------------------

------------------------------------------------------------------------
        Welcome Winston to Winston's (E)Service Provider:
------------------------------------------------------------------------
------------------------
List of Options:
------------------------
1. Search For Service(s)
2. View Cart
3. Check Out/Payment
4. Display Current Subscriptions
5. Change Password
6. Delete Account
7. Exit ESP

Please Choose An Option: 4
============================================================
Showing Current Subscriptions:
------------------------------------------------------------
Service Name            Start Date      End Date
------------------------------------------------------------
Firewall Services       2021-02-06      2022-02-05
Security Ops Centre     -               -
Hot Site                -               -
Data Protection         -               -
------------------------------------------------------------
------------------------
```

Client Output [userClient.py]	**Server Output** [generalServer.py]
Scenario 2: Saving To Cart *[Main Menu Option 7]* 1) Enter and Add Into Cart	1) Output From Server

```
Enter The Number Of The Service or ENTER to stop:
Finish Adding Services...

=--------------=
Suboption Menu 1:
=--------------=
        1. Add To Cart (No.)
        2. Search Again
        3. View Cart
        4. Return To Main Menu

        Enter A Number: 3
Processing Input...
Grabbing Cart...
----------------------------=Current Cart:=----------------------------
Name Of Service:        Quantity        Discount        Cost
Security Ops Centre     1               0.0%            4200.00
Data Protection         1               0.0%            10000.00
Total Cost: $14200.00
--------------------------------------------------------------------------
```

```
Connected to: 127.0.0.1:54873
Thread Number: 7
Active threads: 0
Received Category Choice
Processing...
Category Option 1: Registered User...
Valid Credentials Sent By Client.
Closed connection for: 127.0.0.1:54873

Connected to: 127.0.0.1:55082
Thread Number: 8
Active threads: 0
Main Menu Option 1 Selected: Search For Service
Closed connection for: 127.0.0.1:55082

Connected to: 127.0.0.1:55085
Thread Number: 9
Active threads: 0
Get Price For Service - Security Ops Centre
Closed connection for: 127.0.0.1:55085

Connected to: 127.0.0.1:55086
Thread Number: 10
Active threads: 0
Get Price For Service - Data Protection
Closed connection for: 127.0.0.1:55086

Connected to: 127.0.0.1:55363
Thread Number: 11
Active threads: 0
Main Menu Option 7 Selected: Leaving ESP
Closed connection for: 127.0.0.1:55363
```

2) Save To Server

```
=--------------=
Suboption Menu 2:
=--------------=
        1. Remove A Service
        2. Retrieve Order From Previous Session
        3. Check Out/Payment
        4. Return To Main Menu
==========================================================================
Enter An Suboption No.: 4
Processing Input...
------------------------
List of Options:
------------------------
1. Search For Service(s)
2. View Cart
3. Check Out/Payment
4. Display Current Subscriptions
5. Change Password
6. Delete Account
7. Exit ESP

Please Choose An Option: 7
There Are Still Item(s) In Cart!
Do You Want To Save? (Y/N): Y
Enter 'A4' To Retrieve This Cart Order Next Time!

Thanks For Using Winston's (E)Service Provider!
```

8.2 Admin Program

This sample run will be simulated as an administrator of the ESP.

Client Output [adminClient.py]	**Server Output** [generalServer.py]

1) Entering The Credentials
`-==========-`
`Starting Up...`
`Enter Admin's Username: Admin`
`Enter Admin's Password: *****`
	1) Validate Credentials
	`Server starts listening...`
	`Connected to: 127.0.0.1:57887`
	`Thread Number: 1`
	`Active threads: 0`
	`Recevied Admin Credentails`
	`Processing...`
	`Closed connection for: 127.0.0.1:57887`

2) Manage User Option *[Main Menu Option 1]*
`Welcome Admin to Winston's (E)Service Provider:`
`List Of Options:`
`1. Manage User`
`2. Manange Services`
`3. Exit ESP`
`Please Choose An Option: 1`
`#----------#`
`Suboption Menu 1`
`#----------#`
`1. Delete Registered User`
`2. Modify User Discounts`
`3. Reset User Password`
`4. Return To Main Menu`
`Please Choose An Option:`
	2) Get List Of User
	`Connected to: 127.0.0.1:57900`
	`Thread Number: 2`
	`Active threads: 0`
	`Grabbing List Of Users.`
	`Closed connection for: 127.0.0.1:57900`

3) Delete User *[Suboption 1 Option 1]*
`List Of Options:`
`1. Manage User`
`2. Manange Services`
`3. Exit ESP`
`Please Choose An Option: 1`
`#----------#`
`Suboption Menu 1`
`#----------#`
`1. Delete Registered User`
`2. Modify User Discounts`
`3. Reset User Password`
`4. Return To Main Menu`
`Please Choose An Option: 1`
`Users Pending Deletion`
`1. Ryan`
`Please Enter Name Of User To Delete:`
	3) Get List Of Users Pending Deletion
	`Connected to: 127.0.0.1:57953`
	`Thread Number: 3`
	`Active threads: 0`
	`Grabbing List Of Pending Deletions`
	`Closed connection for: 127.0.0.1:57953`

Client Output [adminClient.py]	**Server Output** [generalServer.py]
(Continued)	**(Continued)**

```
=-----------------------------------=
       Users Pending Deletion
=-----------------------------------=
1. Ryan
=-----------------------------------=
Please Enter Name Of User To Delete: Ryan
Are You Sure To Delete User Ryan (Y/N): Y
User Ryan Has Been Deleted.
-------------------------------------

Returning To Suboption Menu 1...


#-----------------#
 Suboption Menu 1
#-----------------#
        1. Delete Registered User
        2. Modify User Discounts
        3. Reset User Password
        4. Return To Main Menu

Please Choose An Option:
```

```
Connected to: 127.0.0.1:57977
Thread Number: 4
Active threads: 0
User Ryan Has Been Deleted.
Closed connection for: 127.0.0.1:57977
```

4) Change User Discounts *[Suboption 1 Option 2]*

4) Grab the discounts and Update Them Accordingly

```
#-----------------#
 Suboption Menu 1
#-----------------#
        1. Delete Registered User
        2. Modify User Discounts
        3. Reset User Password
        4. Return To Main Menu

Please Choose An Option: 2
-------------------------
No.     Username
-------------------------
1.      Betty
2.      Carl
3.      Donovan
4.      Edward
5.      Faith
6.      Winston
-------------------------
=----------------------=
 Modify User Discounts
=----------------------=
Enter A Username To Change Their Discounts: Winston
-----------------------------------------------------
Showing User Winston Discounts:
-----------------------------------------------------
Service Name:           Discount(0 - 1)
Firewall Services       0
Security Ops Centre     0
Hot Site                0
Data Protection         0
-----------------------------------------------------
Enter A Discount Name To Modify or ENTER To Leave: firewall services
Enter A Discount Value For Firewall Services: 0.7
-----------------------------------------------------
Showing User Winston Discounts:
-----------------------------------------------------
Service Name:           Discount(0 - 1)
Firewall Services       0.7
Security Ops Centre     0
Hot Site                0
Data Protection         0
-----------------------------------------------------
Enter A Discount Name To Modify or ENTER To Leave: data protection
Enter A Discount Value For Data Protection: 0.5
-----------------------------------------------------
```

```
Connected to: 127.0.0.1:58107
Thread Number: 4
Active threads: 0
Grabbing Discounts For User Winston.
Closed connection for: 127.0.0.1:58107

Connected to: 127.0.0.1:58108
Thread Number: 5
Active threads: 0
Updated Discounts For User Winston.
Closed connection for: 127.0.0.1:58108

Connected to: 127.0.0.1:58109
Thread Number: 6
Active threads: 0
Grabbing Discounts For User Winston.
Closed connection for: 127.0.0.1:58109

Connected to: 127.0.0.1:58122
Thread Number: 7
Active threads: 0
Updated Discounts For User Winston.
Closed connection for: 127.0.0.1:58122
```

Client Output [adminClient.py]	**Server Output** [generalServer.py]
(Continue) Showing User Winston Discounts: ----- Service Name: Discount(0 - 1) Firewall Services 0.7 Security Ops Centre 0 Hot Site 0 Data Protection 0.5 ----- Enter A Discount Name To Modify or ENTER To Leave: Discounts For User Winston Has Been Updated. Returning To Suboption Menu 1... #----------# Suboption Menu 1 #----------# 1. Delete Registered User 2. Modify User Discounts 3. Reset User Password 4. Return To Main Menu Please Choose An Option:	<OUTPUT FROM PREVIOUS PAGE>
5) Reset Password For User *[Suboption 1 Option 3]* #----------# Suboption Menu 1 #----------# 1. Delete Registered User 2. Modify User Discounts 3. Reset User Password 4. Return To Main Menu Please Choose An Option: 3 =----------= Reset User Password =----------= 1. Winston ---------- Enter A Username: Winston The Password For User Winston Has Been Reset. Returning To Suboption Menu 1... #----------# Suboption Menu 1 #----------# 1. Delete Registered User 2. Modify User Discounts 3. Reset User Password 4. Return To Main Menu Please Choose An Option:	5) Grab List Of Users Pending Reset Password And Reset User “Winston” Password. Connected to: 127.0.0.1:58183 Thread Number: 8 Active threads: 0 Grabbing List Of Pending Password Resets. Closed connection for: 127.0.0.1:58183 Connected to: 127.0.0.1:58184 Thread Number: 9 Active threads: 0 Password Reset Successful For User Winston. Closed connection for: 127.0.0.1:58184

Client Output [adminClient.py]	**Server Output** [generalServer.py]

6) Manage Services Option *[Main Menu Option 2]*
 	
`-------------------------`
`List Of Options:`
`-------------------------`
`1. Manage User`
`2. Manange Services`
`3. Exit ESP`
 	
`Please Choose An Option: 2`
 	
`=------------------=`
`Suboption Menu 2`
`=------------------=`
`1. Add New Service`
`2. Delete Current Service`
`3. Modify Service`
`4. Return To Main Menu`
`--------------------`
 	
`Please Choose An Option:`
	6) Get List Of Services
	
	`Connected to: 127.0.0.1:58257`
	`Thread Number: 10`
	`Active threads: 0`
	`Grab List Of Services.`
	`Closed connection for: 127.0.0.1:58257`

7) Add New Service *[Suboption 2 Option 1]*
 	
`=-----------------=`
`Suboption Menu 2`
`=-----------------=`
`1. Add New Service`
`2. Delete Current Service`
`3. Modify Service`
`4. Return To Main Menu`
`-------------------`
 	
`Please Choose An Option: 1`
`=---------------------=`
`Adding New Service`
`=---------------------=`
`==========`
`Current Services: Cost ($):`
`==========`
`Firewall Services 1200`
`Security Ops Centre 4200`
`Hot Site 8500`
`Data Protection 10000`
`==========`
`Enter New Service Name Or ENTER To Leave: water bottle`
`Enter New Service Cost: $1234`
`Adding New Service...`
`----------`
`==========`
`Current Services: Cost ($):`
`==========`
`Firewall Services 1200`
`Security Ops Centre 4200`
`Hot Site 8500`
`Data Protection 10000`
`Water Bottle 1234`
`==========`
`Enter New Service Name Or ENTER To Leave:`
`Service File Has Been Updated.`
`==========`
`Current Services: Cost ($):`
`==========`
`Firewall Services 1200`
`Security Ops Centre 4200`
`Hot Site 8500`
`Data Protection 10000`
`Water Bottle 1234`
`==========`
	7) Update the Discounts, Subscriptions, and Services
	
	`Connected to: 127.0.0.1:58284`
	`Thread Number: 11`
	`Active threads: 0`
	`Discounts And Subscriptions Files Have Been Updated.`
	`Closed connection for: 127.0.0.1:58284`
	
	`Connected to: 127.0.0.1:58288`
	`Thread Number: 12`
	`Active threads: 0`
	`Services File Has Been Updated.`
	`Closed connection for: 127.0.0.1:58288`

Client Output [adminClient.py]	**Server Output** [generalServer.py]
8) Delete Service *[Suboption 2 Option 2]*	8) Update the Discounts, Subscriptions, and Services

```
=-----------------=
 Suboption Menu 2
=-----------------=
        1. Add New Service
        2. Delete Current Service
        3. Modify Service
        4. Return To Main Menu
--------------------

Please Choose An Option: 2
=----------------------=
 Delete Current Service
=----------------------=
================================================
Current Services:                       Cost ($):
================================================
Firewall Services                       1200
Security Ops Centre                     4200
Hot Site                                8500
Data Protection                         10000
Water Bottle                            1234
================================================
Enter Service Name To Delete Or ENTER To Leave: water bottle
Removing Service Water Bottle
---------------------------------------
================================================
Current Services:                       Cost ($):
================================================
Firewall Services                       1200
Security Ops Centre                     4200
Hot Site                                8500
Data Protection                         10000
================================================
Enter Service Name To Delete Or ENTER To Leave:
Service File Has Been Updated.
================================================
Current Services:                       Cost ($):
================================================
Firewall Services                       1200
Security Ops Centre                     4200
Hot Site                                8500
Data Protection                         10000
================================================
```

```
Connected to: 127.0.0.1:58426
Thread Number: 13
Active threads: 0
Discounts And Subscriptions Files Have Been Updated.
Closed connection for: 127.0.0.1:58426

Connected to: 127.0.0.1:58427
Thread Number: 14
Active threads: 0
Services File Has Been Updated.
Closed connection for: 127.0.0.1:58427
```

Client Output [adminClient.py]	**Server Output** [generalServer.py]
9) Modify Service *[Suboption 2 Option 3]*	9) Update the Services File

```
=================
 Suboption Menu 2
=================
        1. Add New Service
        2. Delete Current Service
        3. Modify Service
        4. Return To Main Menu
-------------------

Please Choose An Option: 3
=====================
 Modify Current Service
=====================


================================================
Current Services:                   Cost ($):
================================================
Firewall Services                   1200
Security Ops Centre                 4200
Hot Site                            8500
Data Protection                     10000
================================================
Enter Service Name To Modify Or ENTER To Leave: data protection
Enter A New Cost For Data Protection: $1234
Service File Has Been Updated.


================================================
Current Services:                   Cost ($):
================================================
Firewall Services                   1200
Security Ops Centre                 4200
Hot Site                            8500
Data Protection                     1234
================================================
Enter Service Name To Modify Or ENTER To Leave:
================================================
Current Services:                   Cost ($):
================================================
Firewall Services                   1200
Security Ops Centre                 4200
Hot Site                            8500
Data Protection                     1234
================================================
```

```
Connected to: 127.0.0.1:58427
Thread Number: 14
Active threads: 0
Services File Has Been Updated.
Closed connection for: 127.0.0.1:58427

Connected to: 127.0.0.1:58445
Thread Number: 15
Active threads: 0
Services File Has Been Updated.
Closed connection for: 127.0.0.1:58445
```

10) Leave Program *[Main Menu Option 3]*

```
-------------------------
List Of Options:
-------------------------
1. Manage User
2. Manange Services
3. Exit ESP

Please Choose An Option: 3
Leaving Admin Client Program...
```

End Of Documentation