Cairo University Faculty of Computers and Artificial Intelligence



**Software design specification document**

**2022**

**Project Team**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Email** |
| 20200163 | Haneen Ehdaa Ibrahim Almarsafy | haneen.ehdaa@gmail.com |
| 20201074 | Rawda Mohamed Hussein | rawda14moh@gmail.com |
| 20201136 | Farah Ashraf Wafaa | Farahashraf77788@gmail.com |
| 20201218 | Walaa Soudy Ibrahim | walaasoudy36@gmail.com |

Contents

[Instructions[To be removed] 2](#_Toc120811426)

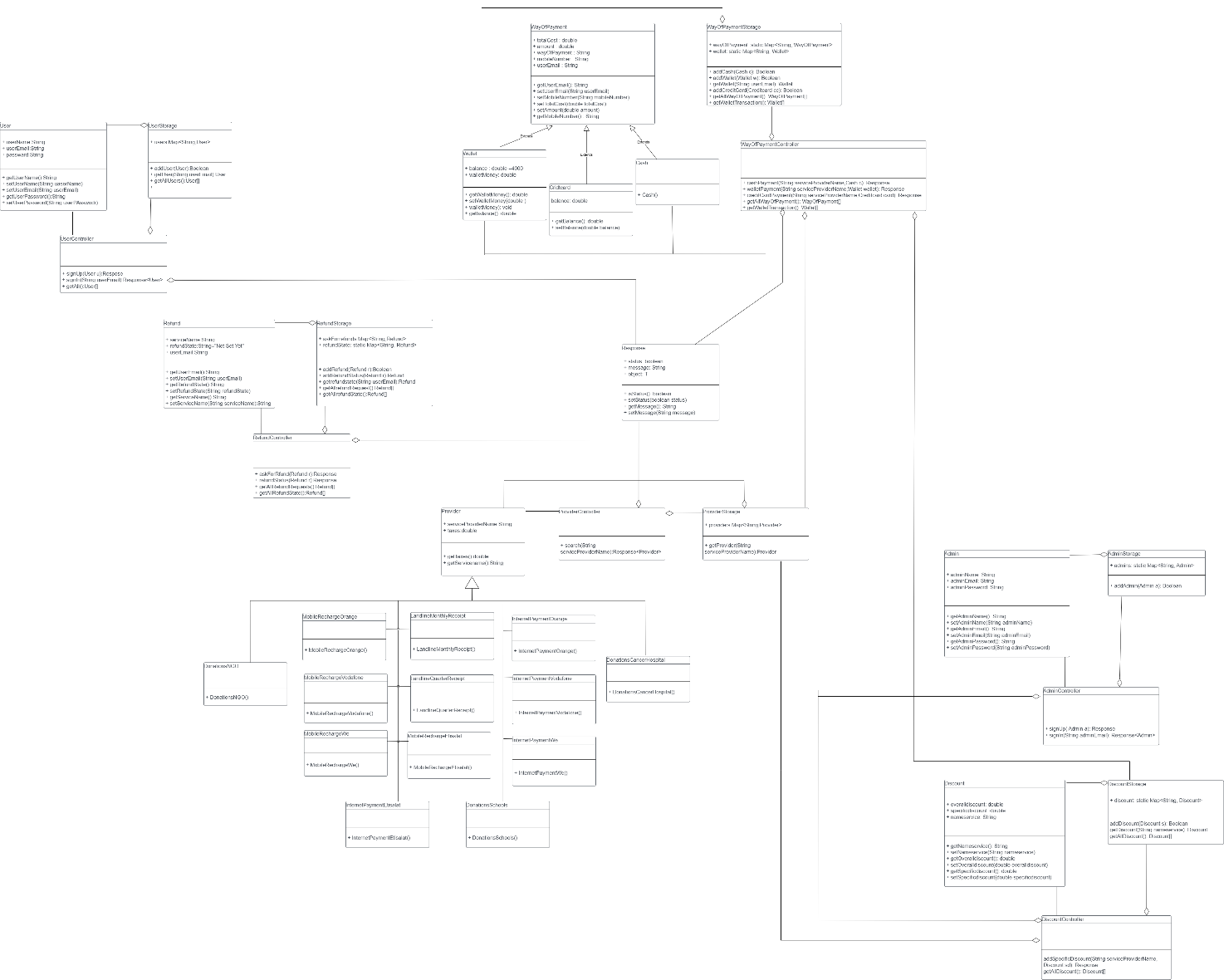
[Class diagram design 2](#_Toc120811427)

[Class diagram Explanation 3](#_Toc120811428)

[Sequence diagram design 3](#_Toc120811429)

[Github repository link 4](#_Toc120811430)

# Class diagram design:

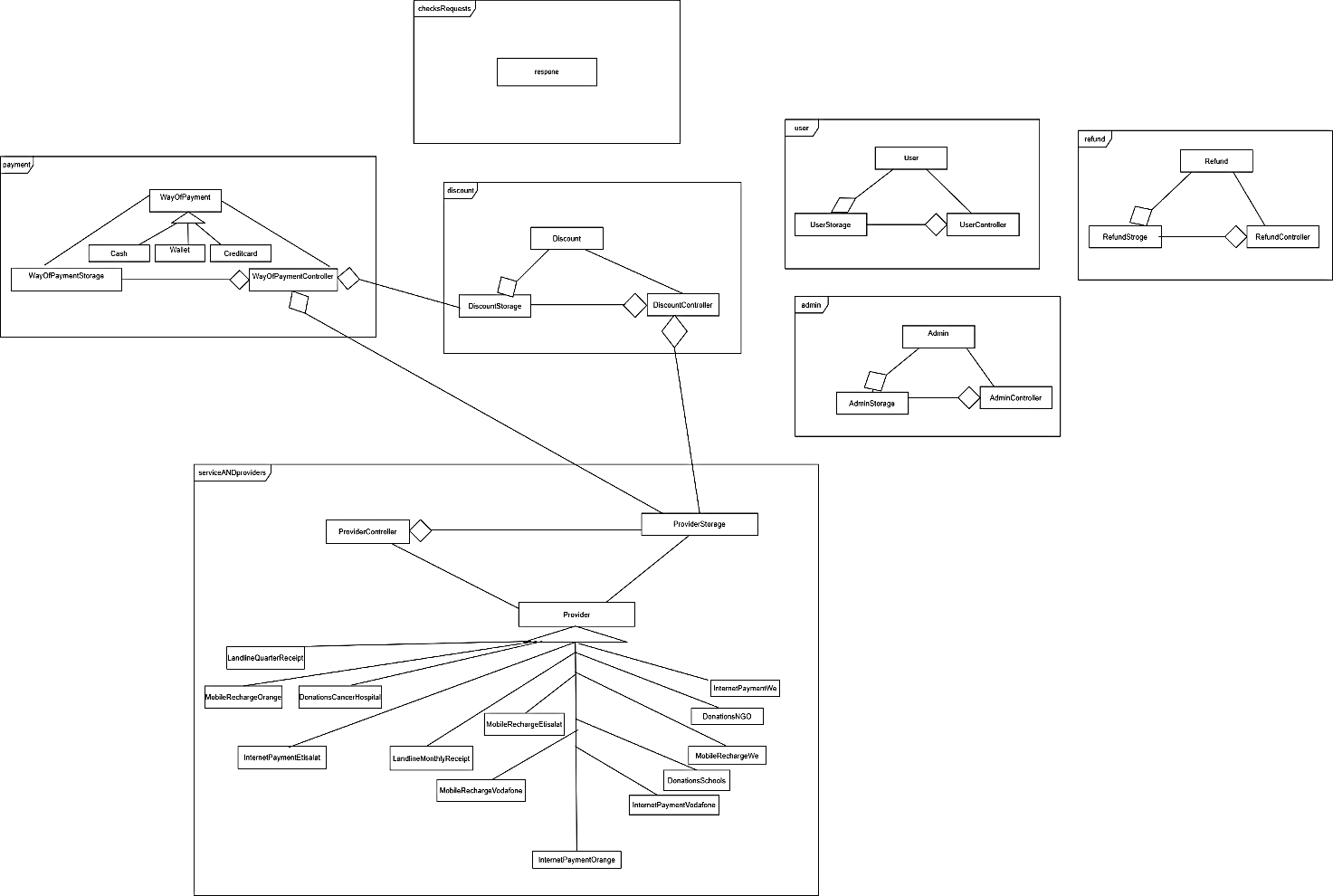


# Class diagram Explanation:

1. We have 3 classes (user , userController, userStorage) in the same package called User class has 3 attributes(user email , user name , password) user controller class which controllers user sign in and sign up user storage which has a map stores user information.
2. We have 3 classes (refund , refundController, refundStorage) in the same package called Refund class has 3 attributes(service name, refund state,user email) refund controller class which controllers refunds requests and state and refund storage which has a map stores refund request information and refund state information.
3. We have 3 classes (overall discount , specific discount, name service) in the same package called discount class has 3 attributes(service name, refund state,user email) refund controller class which controllers refunds requests and state and refund storage which has a map stores refund request information and refund state information.

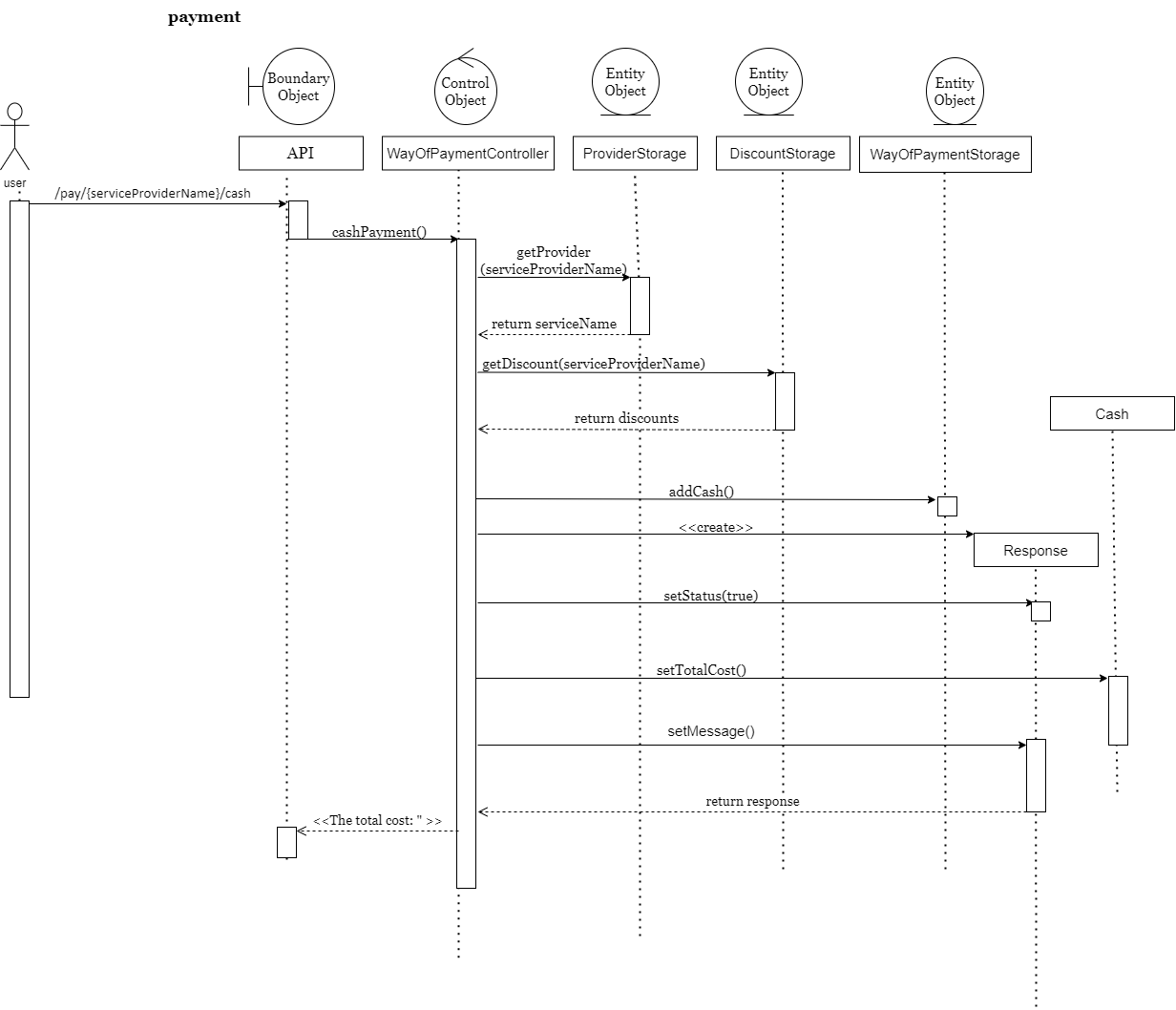
# 

# decomposition diagram

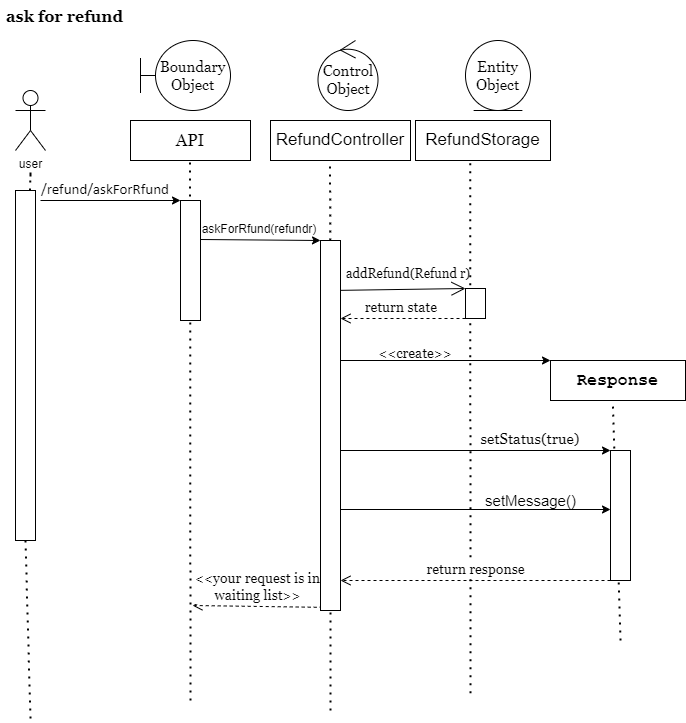


# Sequence diagram design

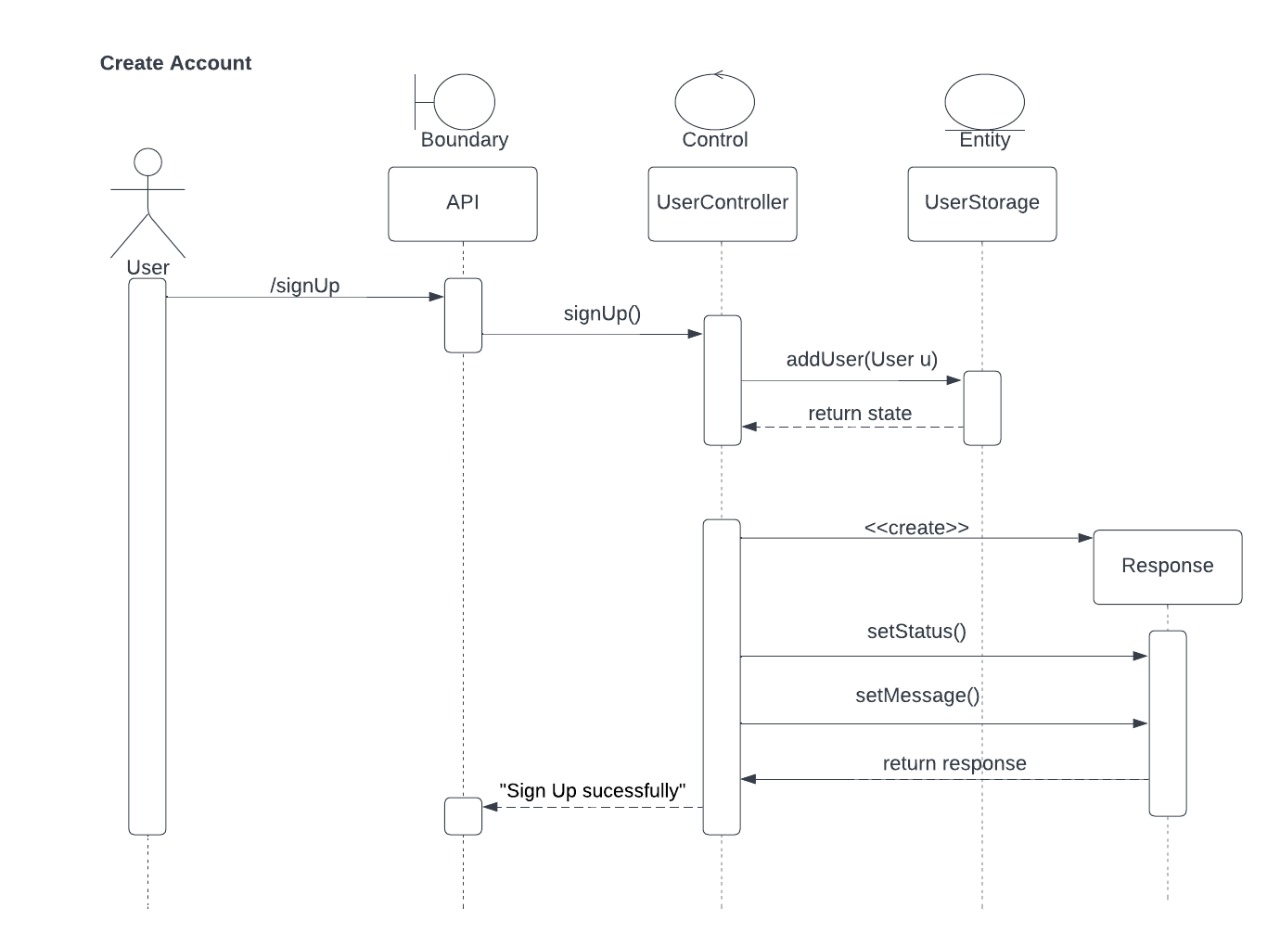
**Payment:**



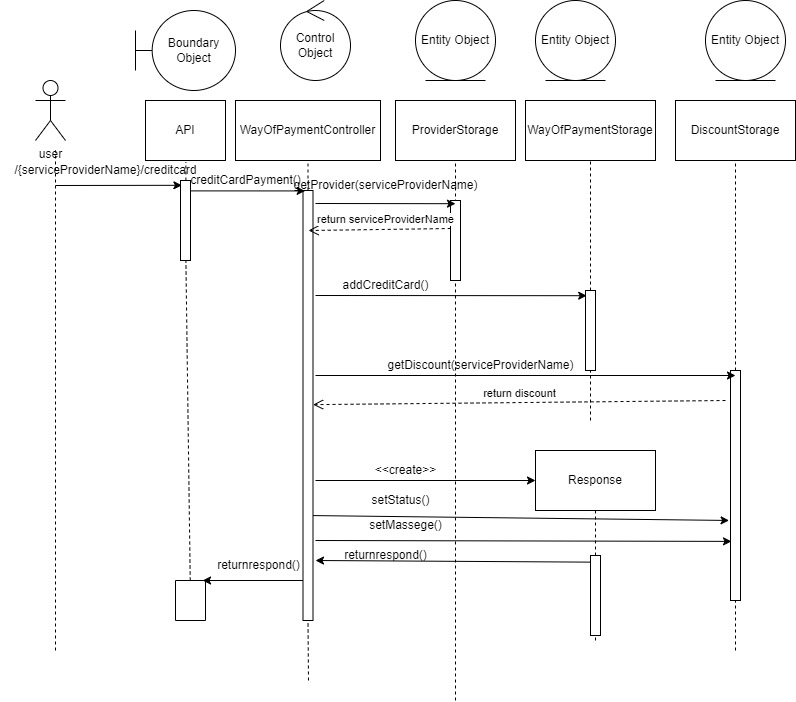
**Ask for refund:**



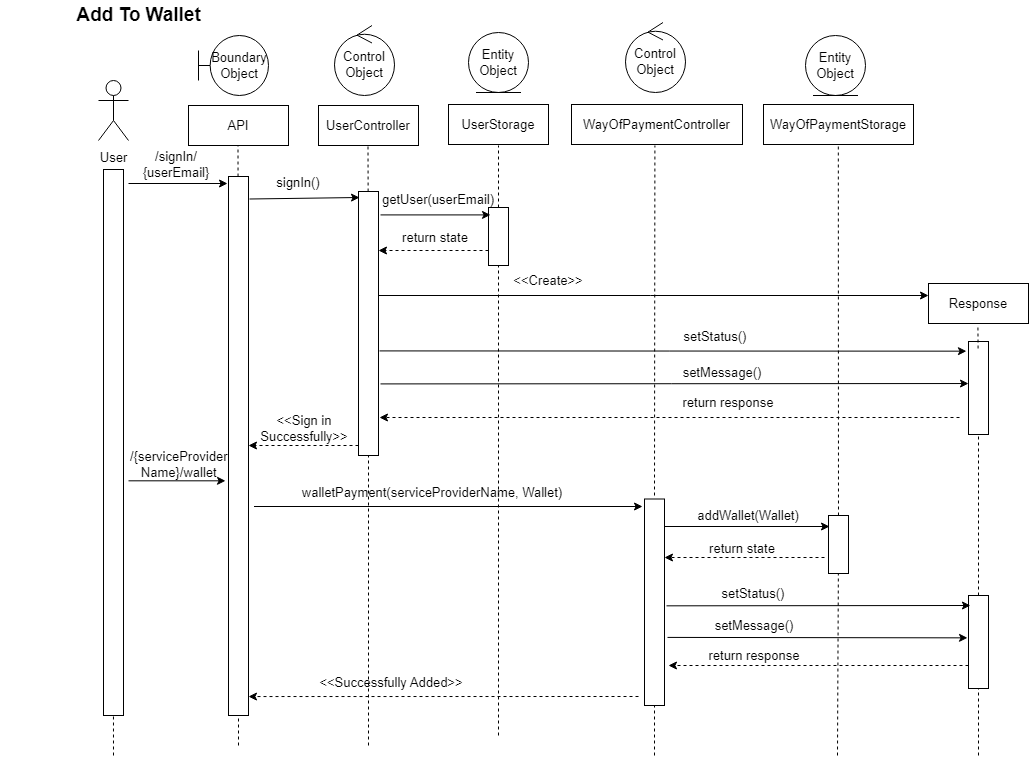
**Create account:**



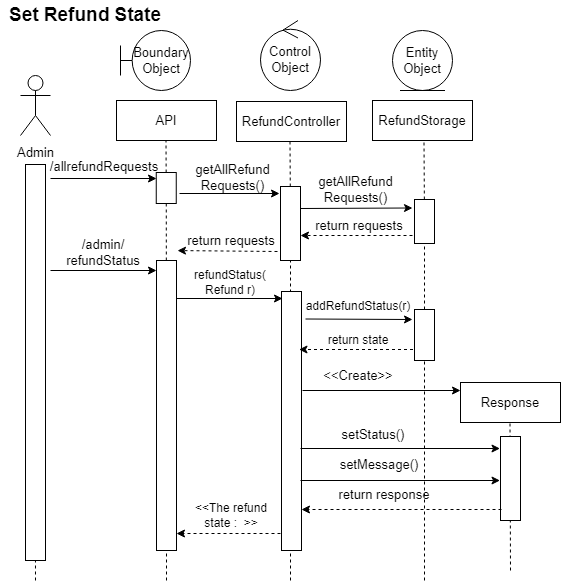
**Payment with discount:**



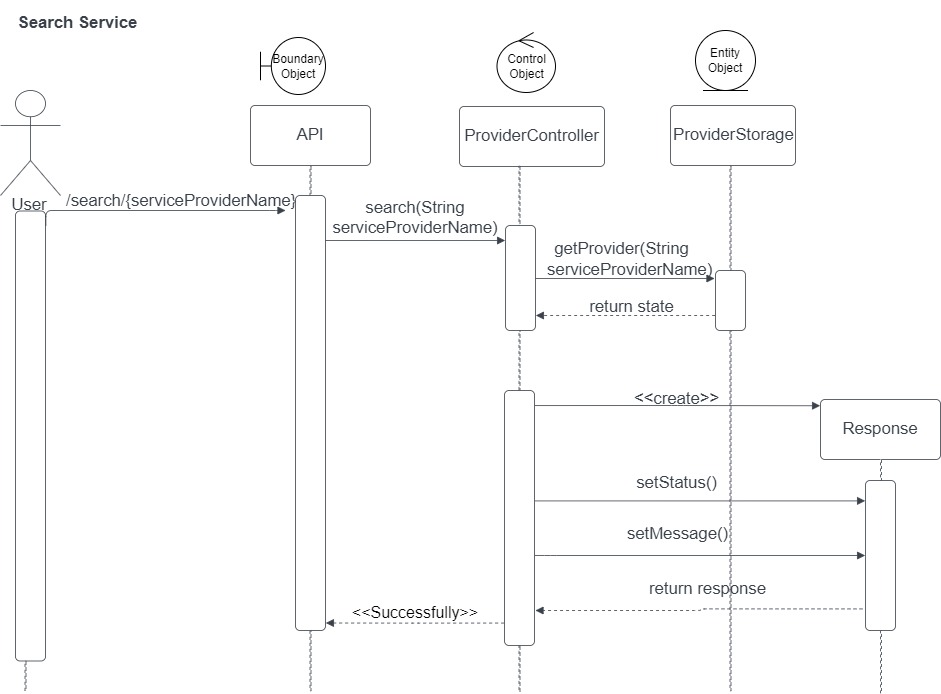
**Add to wallet:**



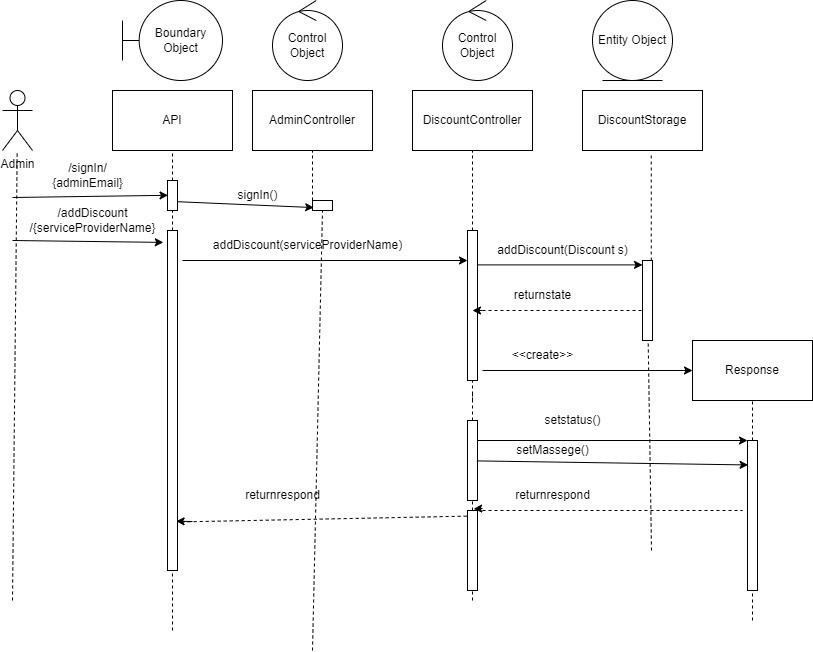
**Set refund state:**



**Search service:**



**Add discount:**



**Part 1:**

**have a good experience in our project :**

The sequence for URLs to have a good experience in our project.

1. Post /admin/signUp --------> admin
2. Post/addDiscount/{serviceProviderName} --------> admin
3. Post/user/signUp ------------> user
4. Get/provider/search/{serviceProviderName} ------------> user
5. Get/addDiscount/checkDiscount ------------> user
6. **Post** /pay/{serviceProviderName}/ creditcard ------------> user
7. Post/refund/askForRfund ------------> user
8. Get /refund/allrefundRequests --------> admin
9. Post/refund/admin/refundStatus --------> admin

**Part 2:**

|  |  |  |
| --- | --- | --- |
| Requirement |  | Exposed API |
| 1. The user should be able to sign up to the system. The user should provide his username, email and password. The system should check if the username or the email is registered before, if they are not registered before then the signup process should complete successfully, if not, the system will show an error to the user |  | 1-Post/user/signUp The user should put username,email and userPassword Inputs: userName, userEmail, userPassword  Note:  All inputs are string  {  "userName": "haneen",  "userEmail": "lll",  "userPassword": "3456"  } |
| 2. The user should be able to sign-in to the system. Given the user’s email and a password, the user can login to the system and use any of the system functionalities |  | 1-Get/user/signIn/{userEmail} Get the user by user email |
| 3. The user should be able to search for any service in the system. The user can type the service name and the system will return all services that match the user query. |  | 1.Get/provider/search/{serviceProviderName} The user can search by service name  Note:  For example:  Get/provider/search/InternetPaymentEtisalat‏  Get/provider/search/InternetPaymentOrange‏‏  Get/provider/search/InternetPaymentVodafone‏‏  Get/provider/search/InternetPaymentWe‏‏  Get/provider/search/MobileRechargeWe‏‏  Get/provider/search/MobileRechargeEtisalat‏‏  Get/provider/search/MobileRechargeVodafone‏‏  Get/provider/search/MobileRechargeOrange‏  Get/provider/search/Landline‏QuarterReceipt‏‏  Get/provider/search/Donations‏NGO‏‏  Get/provider/search/Landline‏Monthlyreceipt‏‏  Get/provider/search/Donations‏CancerHospital‏‏  Get/provider/search/DonationsDSchools‏‏ |
| 4. The user can pay for any service in the system. The system should prompt the user to the payment form when the user asks to pay for any service. The default way is to pay via credit card. The system should allow the user to consume from the wallet (check Req. 6) for this payment. If the service that should to receive the payment accepts cache on delivery, then this option should be visible too. |  | 1.Post /pay/{serviceProviderName}/cash 2. **Post** /pay/{serviceProviderName}/creditcard  3. **Post** /pay/{serviceProviderName}/wallet  Service name as we mentioned before  Inputs for cash:  {  "amount": 100.0,  "mobileNumber": "2345",  "userEmail" : "rawda"  }  Inputs for creditcard:  {  "amount": 197.0,  "mobileNumber": "1234",  "balance":200.0,  "userEmail" : "farah"  }  Inputs for wallet:  {  "amount": 197.0,  "mobileNumber": "01234",  "walletMoney":200.5,  "userEmail" : "farah"  } |
| 5. The user can ask for a refund for any complete transaction to any given service. The refund request will be issued by the user and sent to the admin |  | 1.Post/refund/askForRfund The user can ask for refund by put this url  Inputs:  {  "serviceName" : "Landline‏QuarterReceipt‏",  "userEmail" : "rawda@gmail.com"  } |
| 6. . If the admin approves the refund then the refund process should complete successfully. |  | 1.Post/refund/admin/refundStatus the admin can reject or accept the refund  note: admin should get all refund request First by this  url (Get/refund/allrefundRequests) to get the email to use it to accept or reject refund request  inputs:  {  "serviceName" : "Landline‏QuarterReceipt‏",  "refundState" : "accept",  "userEmail" : "www"  }  Should the same email as allrefundRequests. |
| 7. The system maintain a wallet balance for each user. The user should be able to add any funds to the wallet. Adding funds to the wallet should be done via credit card. |  | Post /pay/{serviceProviderName}/wallet Inputs for wallet:  {  "amount": 30.5,  "mobileNumber": "1234",  "walletMoney":200.5  } |
| 8. The user should be able to check any discount for any service in the system. |  | 1.Get/addDiscount/checkDiscount The user able to check all discounts  Note:  Must be after the admin add discount |
| 9. Discounts could be added by the admin (this will be discussed later). |  | 1.Post/addDiscount/{serviceProviderName} the admin can add specific and overall discount if he didn’t want to set overall discount he should set to 0 and specific discount also  inputs:  {  "nameservice" : "Landline‏QuarterReceipt‏",  "specificdiscount" : 0.5,  "overalldiscount" : 0.25  } |
| 10. The admin should be able to list all user transactions. The transactions types are a. Payment transaction.  b. Add to wallet transaction.  c. Refund transaction |  | 1.Get/pay/getPaymentTransaction **2.Get**/pay/getWalletTransaction  **3.for refund transaction:**  **Get** /refund/allrefundRequests  User can pay, add to wallet and ask for refund |
| 11. The admin should be able to list all refund requests. Each refund request should contain the related service and the amount to be refunded. |  | 1.Get /refund/allrefundRequests admin should be able to list all refund requests. |
| 12.the admin can sign in and sign up also |  | 1.Post /admin/signUp ----------->sign up  2.Get /signIn/{adminEmail}------🡪 sign in  Inputs:  {  "dminName": "haneen",  "adminEmail": "walaa@gmail.com",  "adminPassword": "3456"  } |

# Github repository link

[**https://github.com/farahAshraf777/Phase1\_software2**](https://github.com/farahAshraf777/Phase1_software2)