Cairo University **Faculty of Computers and Artificial Intelligence**

Software design specification document

2022

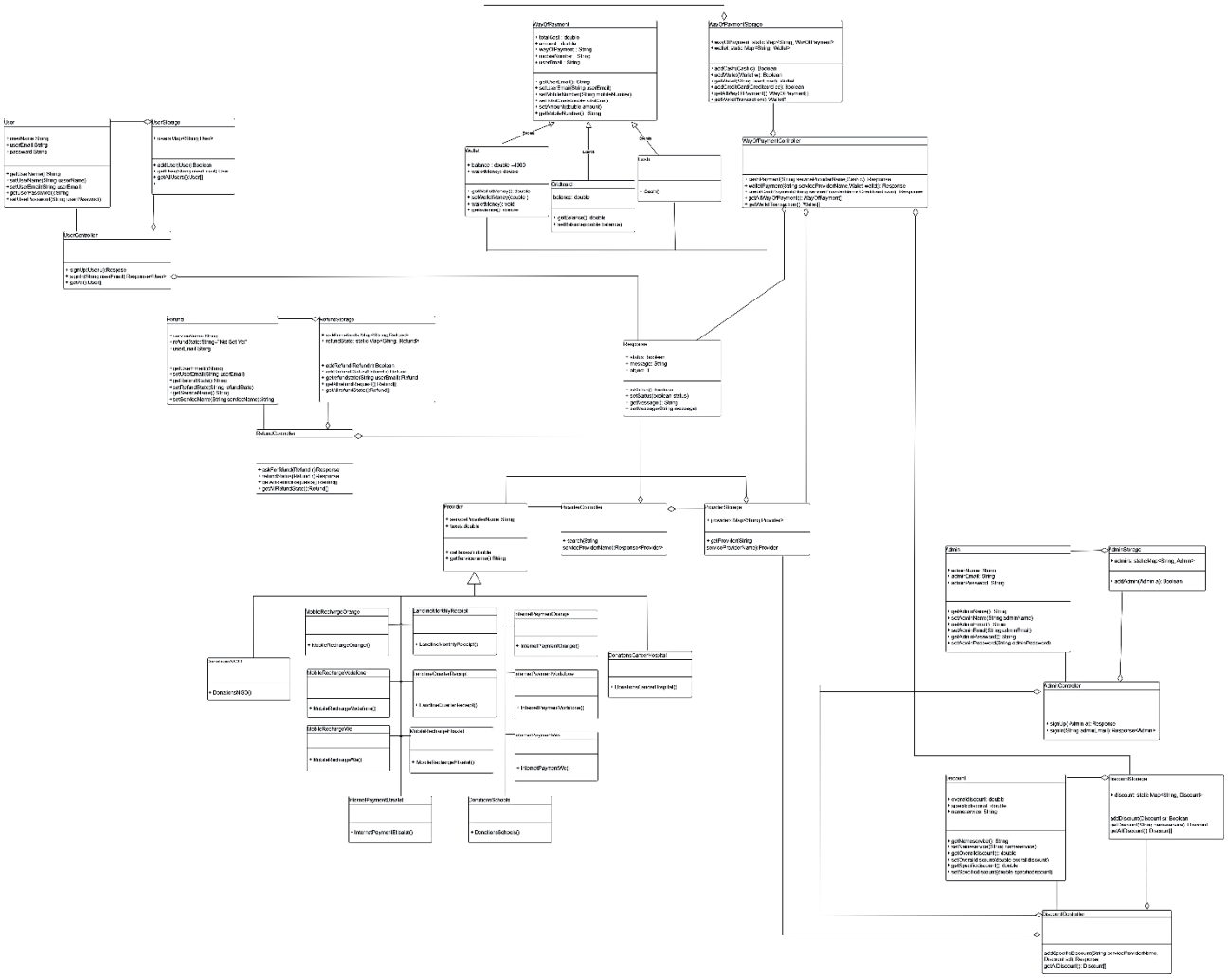
Project Team

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

Contents

SDS document

Class diagram design:



Class diagram Explanation:

1. Package(user):We have 3 classes (user , userController, userStorage) in the same package called User class has 3 attributes(user email , user name ,

SDS document

password) user controller class which controllers user sign in and sign up user storage which has a map stores user information.

2. Package(refund):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. Package(discount):We have 3 classes (discount , discount controller,discount storage) in the same package called discount class has 3 attributes(name service, overall discount ,specific discount) discount controller class which controllers when admin want to add overall discount and specific discount and discount storage which has a map stores discount vales.

4.package(serviceANDproviders): We have 16 classes (DonationsCancerHospital, DonationsSchools, DonationsNGO, InternetPaymentEtisalat, InternetPaymentVodafone, InternetPaymentOrange, InternetPaymentWe, LandlineMonthlyReceipt, MobileRechargeEtisalat, MobileRechargeOrange, LandlineQuarterReceipt, MobileRechargeVodafone, MobileRechargeWe, Provider, ProviderController, ProviderStorage) in the same package 13 of them is the types of the provider class , ProviderController class which controllers when the user search about specific service and ProviderStorage which has a map stores providers service types.

5.package(payment): We have 6 classes(WayOfPayment, Cash, Creditcard, Wallet, WayOfPaymentController, WayOfPaymentStorage) in the same package, WayOfPayment has 5 attributes(userEmail, totalCost ,amount, wayOfPayment, mobileNumber), Creditcard has attribute(balance), Wallet has 2 attributes(balance,walletMoney), WayOfPaymentController class which controllers when user want to choose the way of payment and select the discount if it is exist,

SDS document

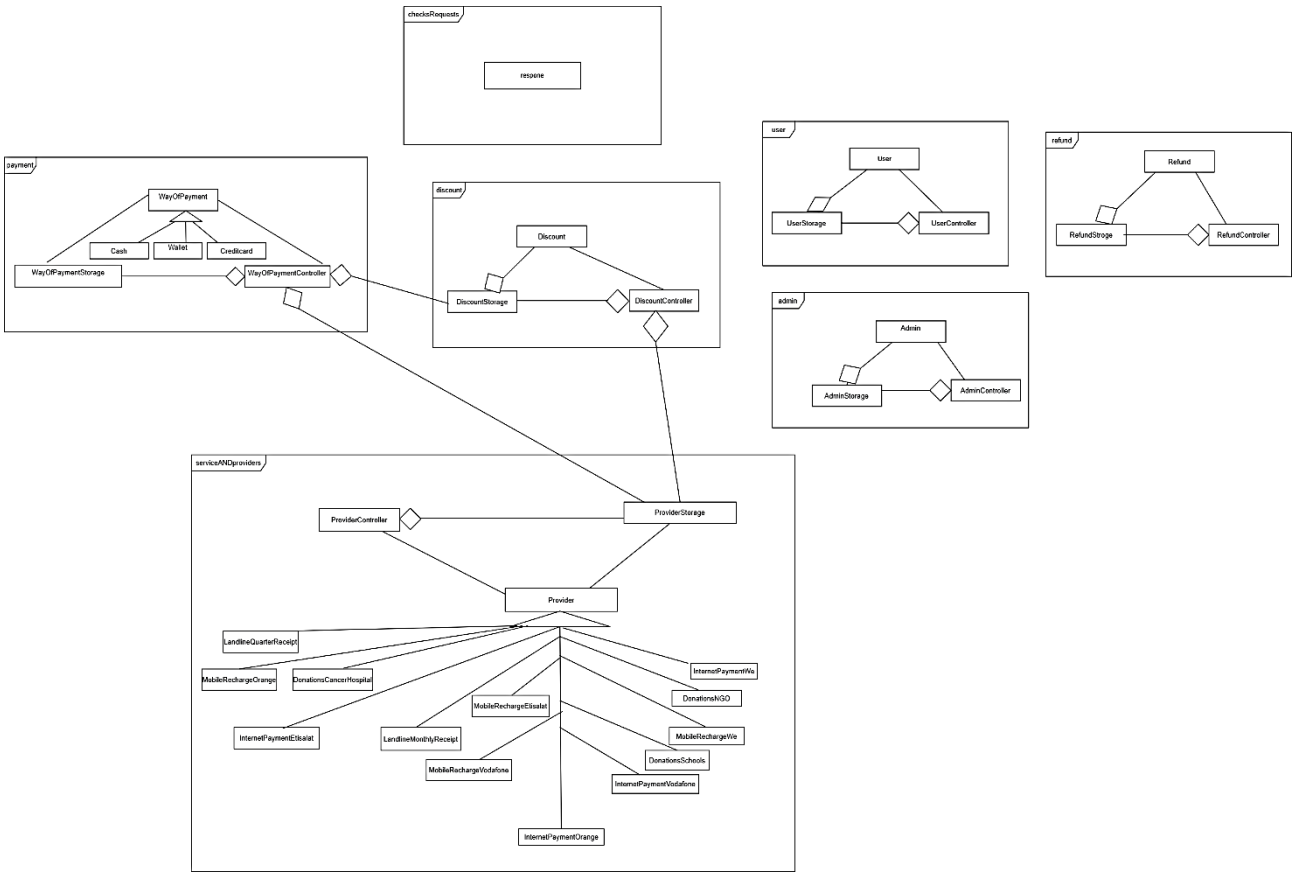
WayOfPaymentStorage which has a map stores providers service types set the amount of mony in the wallet credit cart.

6. package(admin): We have 3 classes(Admin, AdminController, AdminStorage) in the same package Admin has attributes(adminName, adminEmail, adminPassword) admin controller class which controllers admin sign in/up requests and AdminStorage which has a map stores admin data request information and refund state information

7. package (checksRequests): We have class(Response) with attributes(status, message, T object) all classes using it to send the response.

diagram

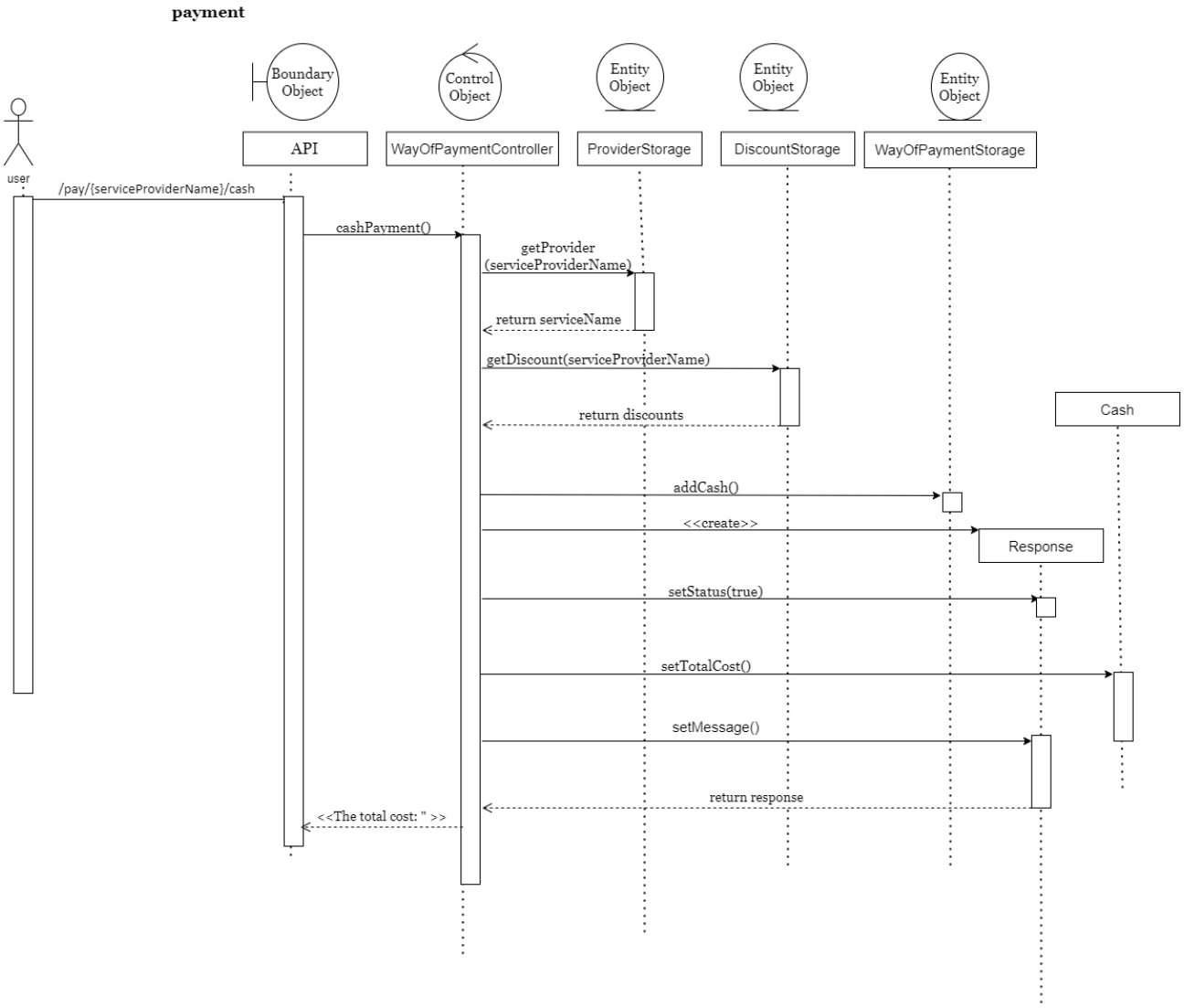
SDS document



Sequence diagram design

Payment:

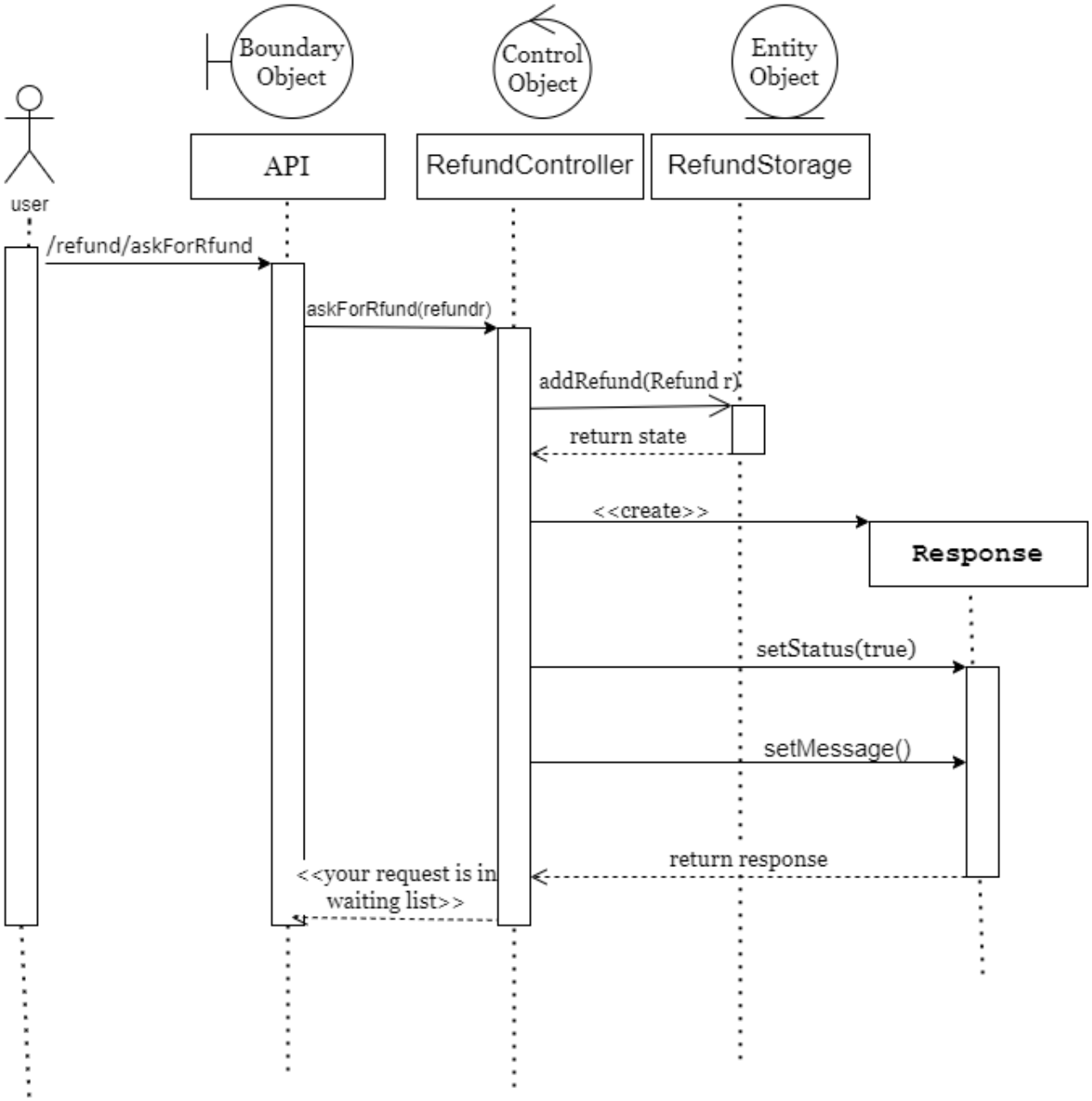
SDS document



Ask For Refund

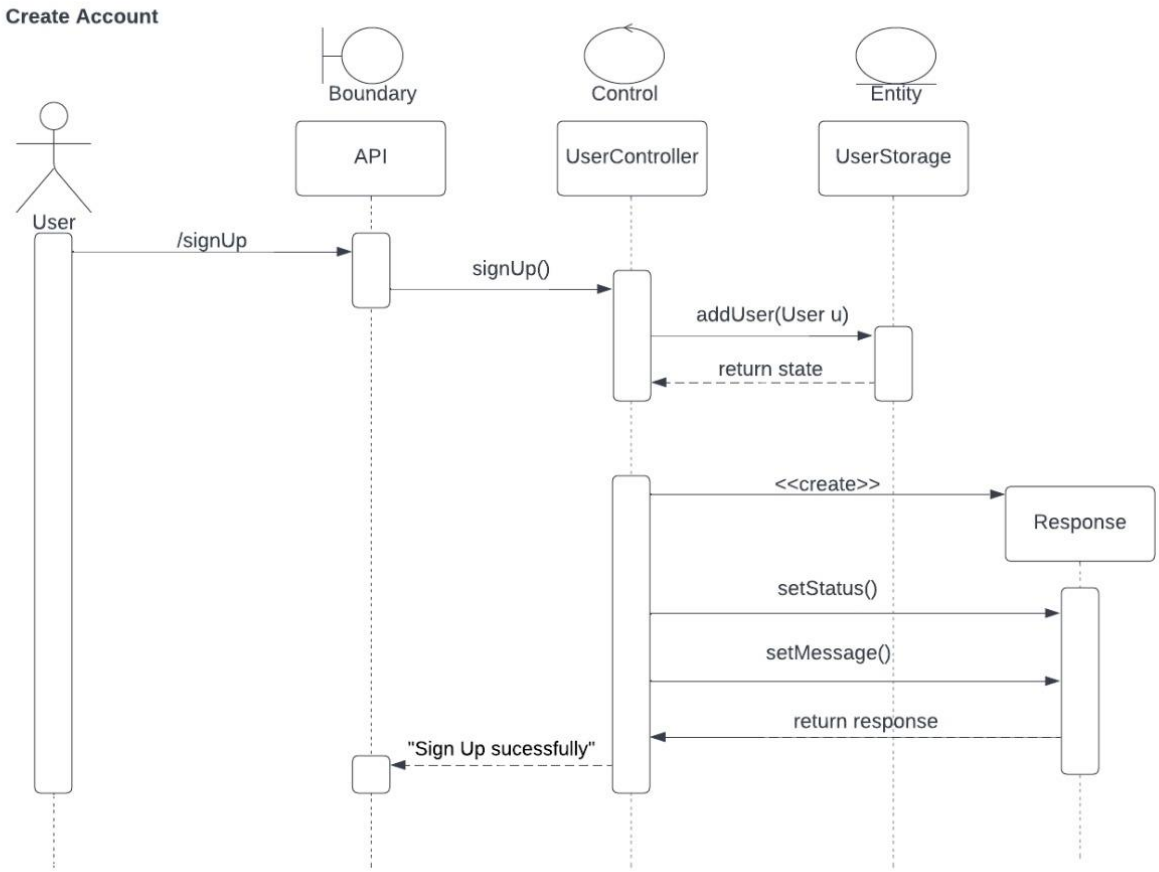
SDS document

ask for refund



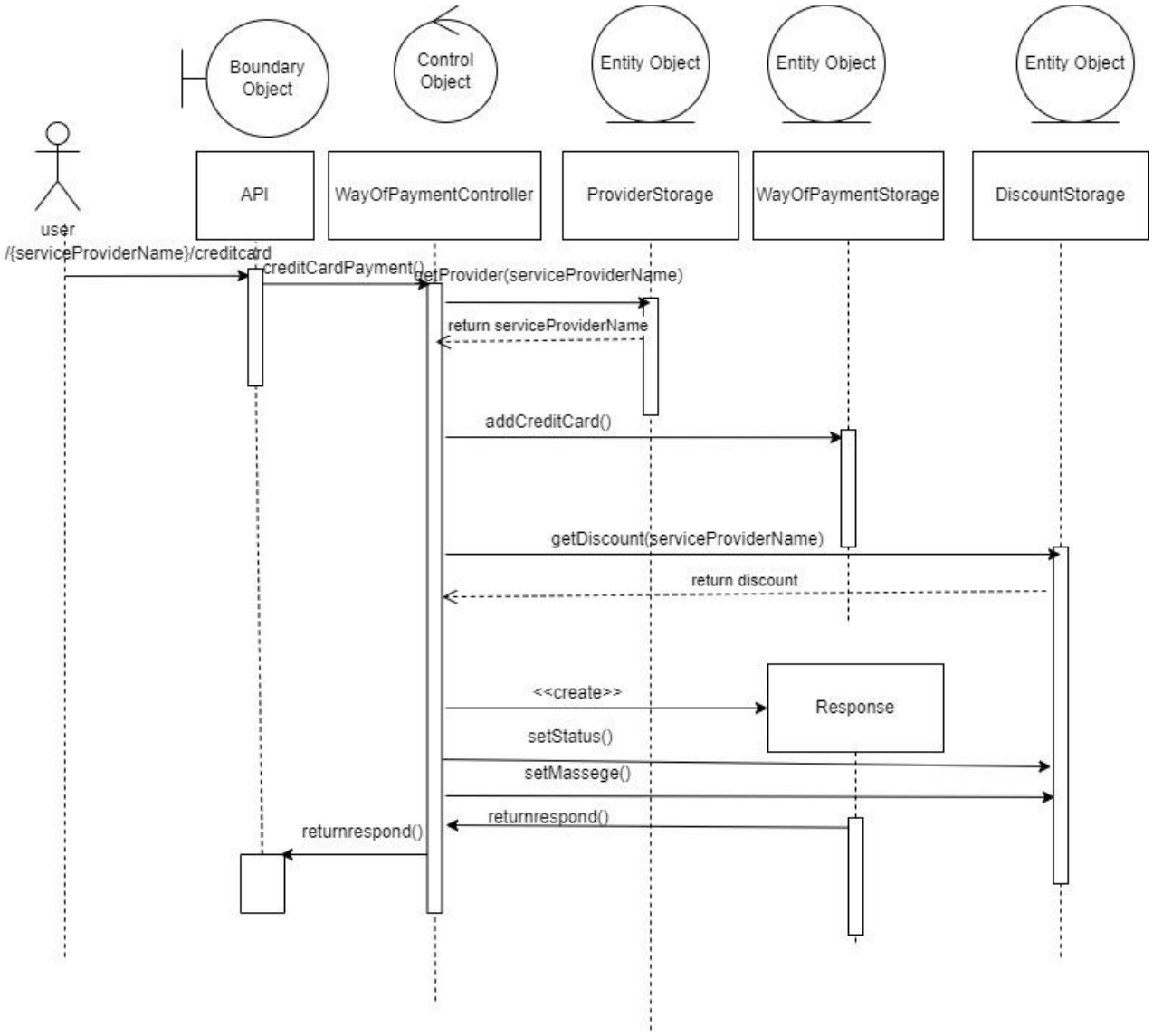
SDS document

Create account:



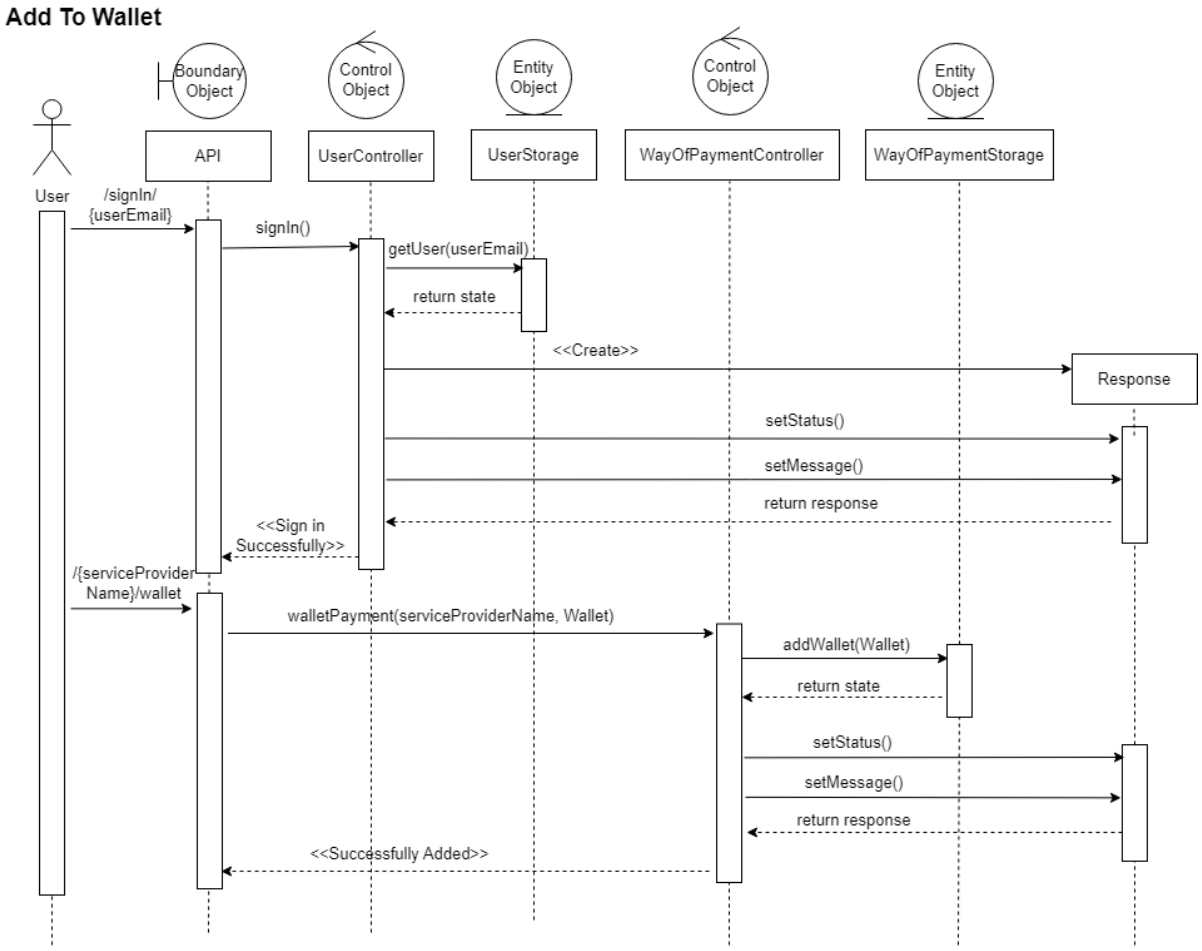
SDS document

Payment with discount:



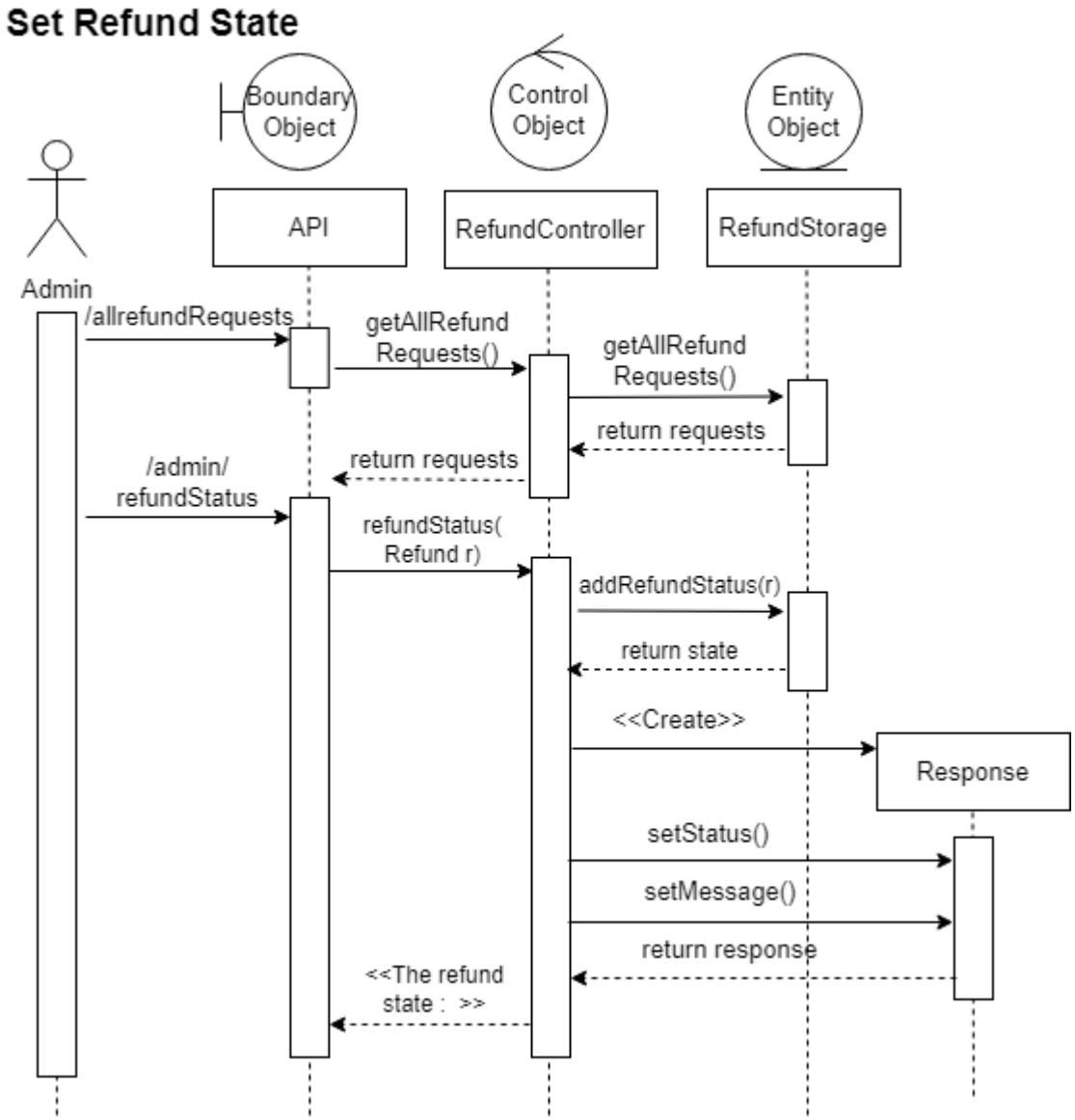
SDS document

Add to wallet:



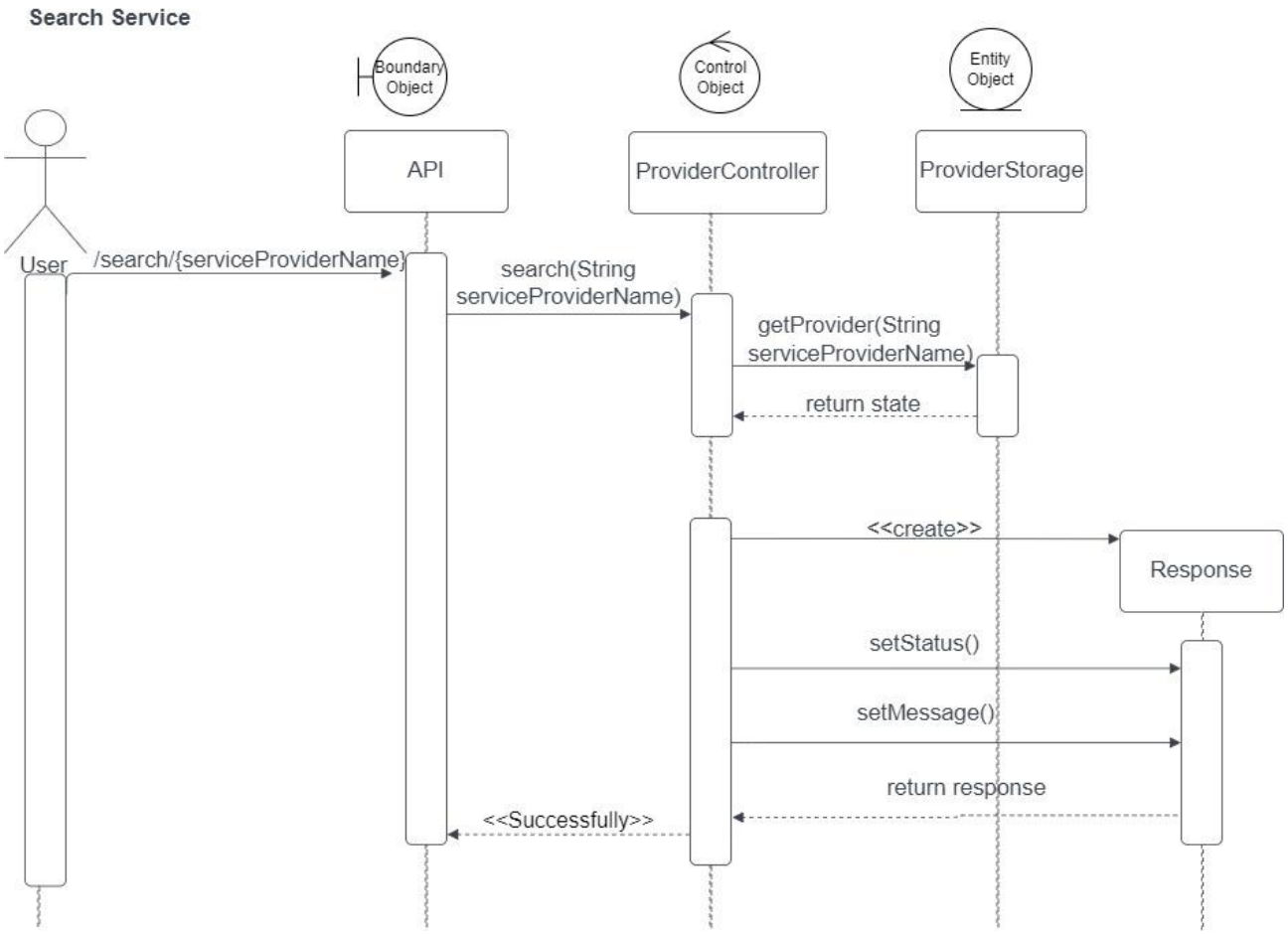
SDS document

Set refund state:



SDS document

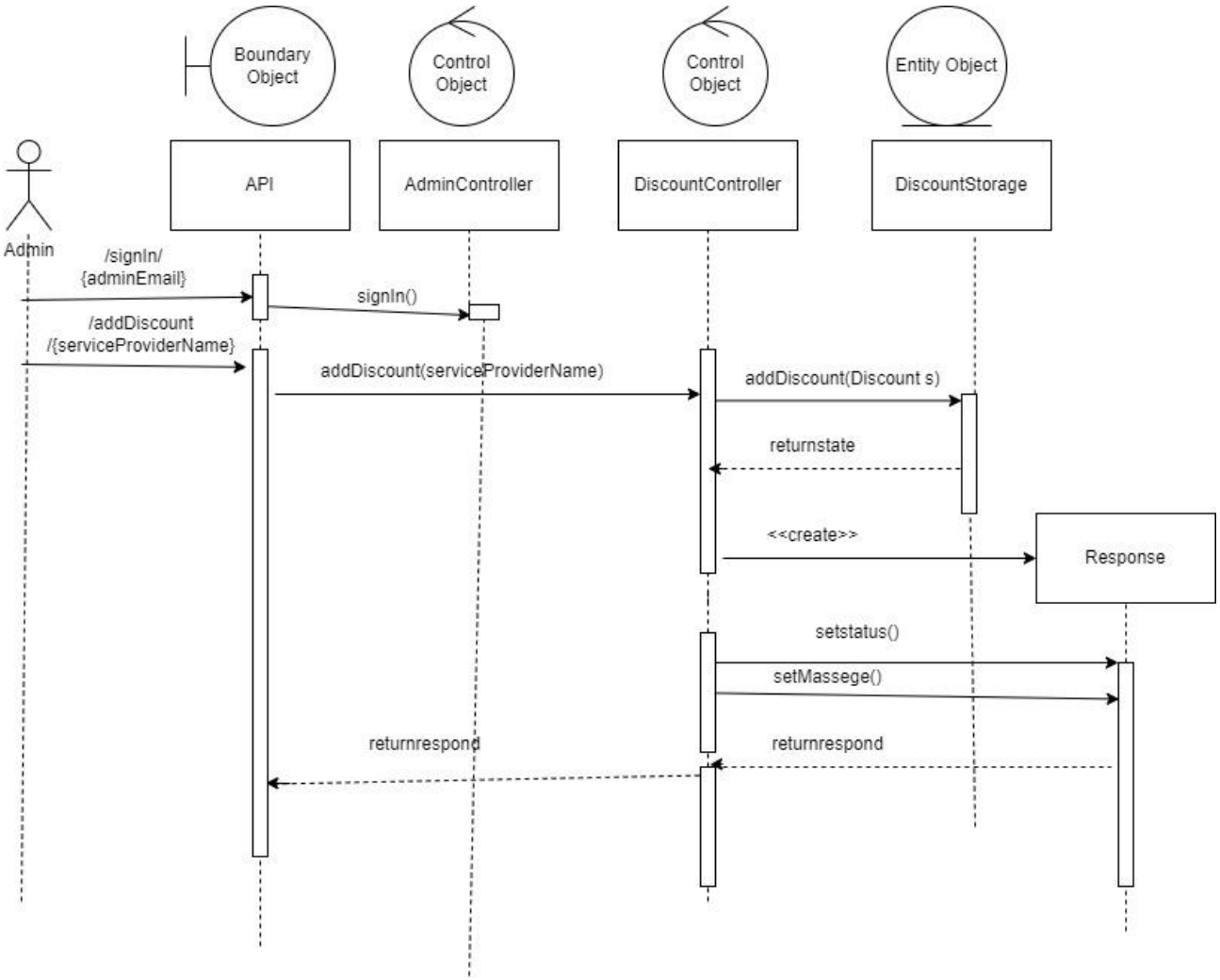
Search service:



SDS document

Add discount:

SDS document



Part 1:

SDS document

// our collection included in the folder

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

SDS document

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		**1-**Get/user/signIn/{userEmail} Get the user by user email

a password, the user can login to the system and use any of the system functionalities		
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		1.Post /pay/{serviceProviderName}/cash 2. **Post** /pay/{serviceProviderName}/creditcard 3. **Post** /pay/{serviceProviderName}/wallet

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.		Service name as we mentioned before
		Inputs for cash:
		{
		"amount": 100.0,
		"mobileNumber": "2345",
		"userEmail" : "rawda"
		}
		Inputs for creditcard:
		{
		"amount": 30.5,
		"mobileNumber": "1234",
		"balance":200.0,
		"userEmail" : "farah"
		}
		Inputs for wallet:
		{
		"amount": 30.5,
		"mobileNumber": "1234",
		"walletMoney":200.5,
		"userEmail" : "walaa"
		
		}

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		1.Post/refund/askForRfund The user can ask for refund by put this url Inputs: { "serviceName" : "LandlineQuarterReceipt", "userEmail" : "rawda@gmail.com"

SDS document

admin		}

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.		1. Post /pay/{serviceProviderName}/wallet
		Inputs for wallet:
		{
		"amount": 30.5,
		"mobileNumber": "1234",
		"walletMoney":200.5,
		"userEmail": "rawda"
		}

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:

SDS document

		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		1.Get /refund/allrefundRequests admin should be able to list all refund requests.

SDS document

the amount to be refunded.		

12.the admin can sign in and sign up also		1.Post /admin/signUp ----------->sign up
		2.Get /signIn/{adminEmail}------→ sign in
		Inputs:
		{
		"adminName": "haneen",
		"adminEmail": "walaa@gmail.com",
		"adminPassword": "3456"
		}

Github repository link

https://github.com/farahAshraf777/Phase1_software2