Cairo University Faculty of Computers and Artificial Intelligence



**Software design specification document**

**2022**

**Project Team**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Email** |
| 20200317 | Abdelrahman Fawzy Abdel-Ghaffar | bobofawzy3@gmail.com |
| 20200665 | Yousef Mohamed Hassan | muhamedyousef05@gmail.com |
| 20200659 | Youssef Magdy Mohamed | sasamego893@gmail.com |
| 20200166 | Khaled Osama Abdallah | khaleddoosama@gmail.com |

# 

# Class diagram design

Diagram

Description automatically generated

A picture containing diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram, engineering drawing

Description automatically generated

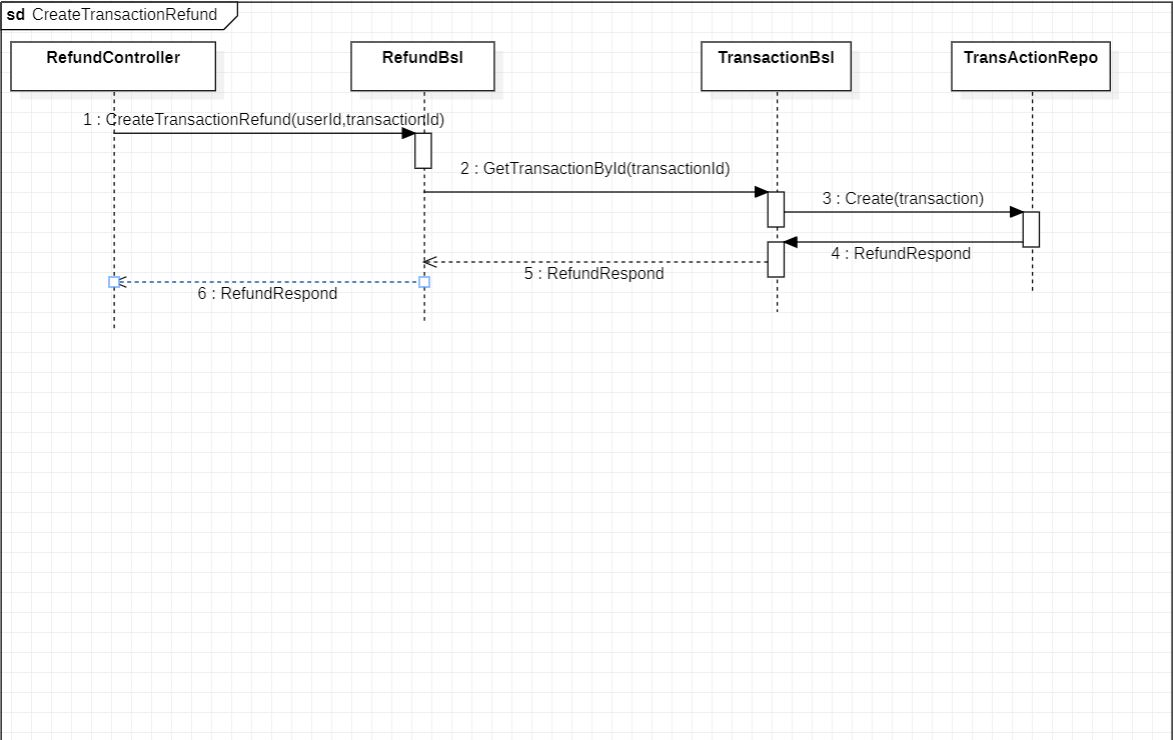
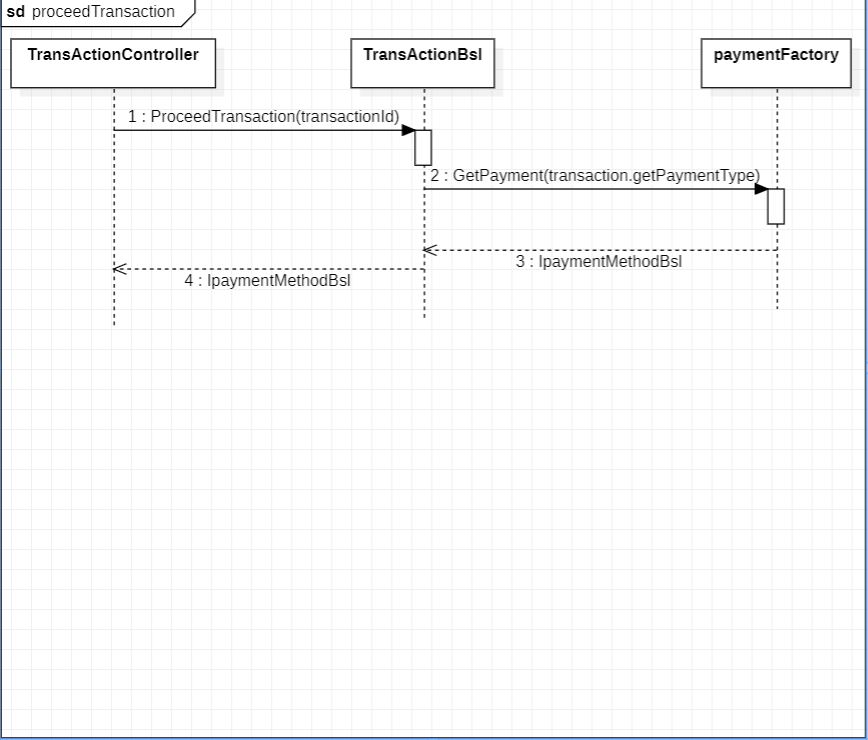
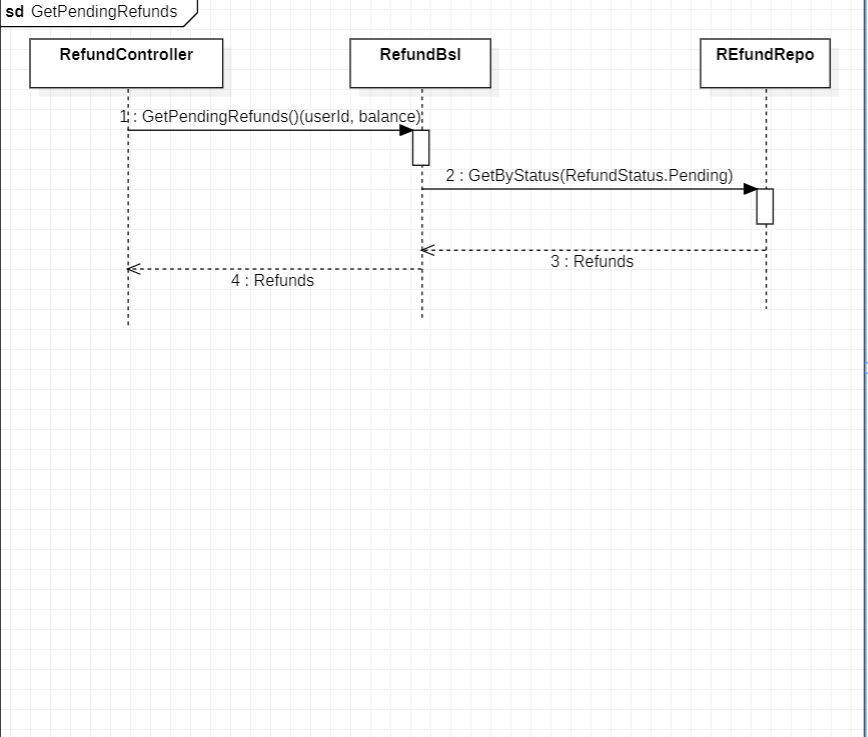
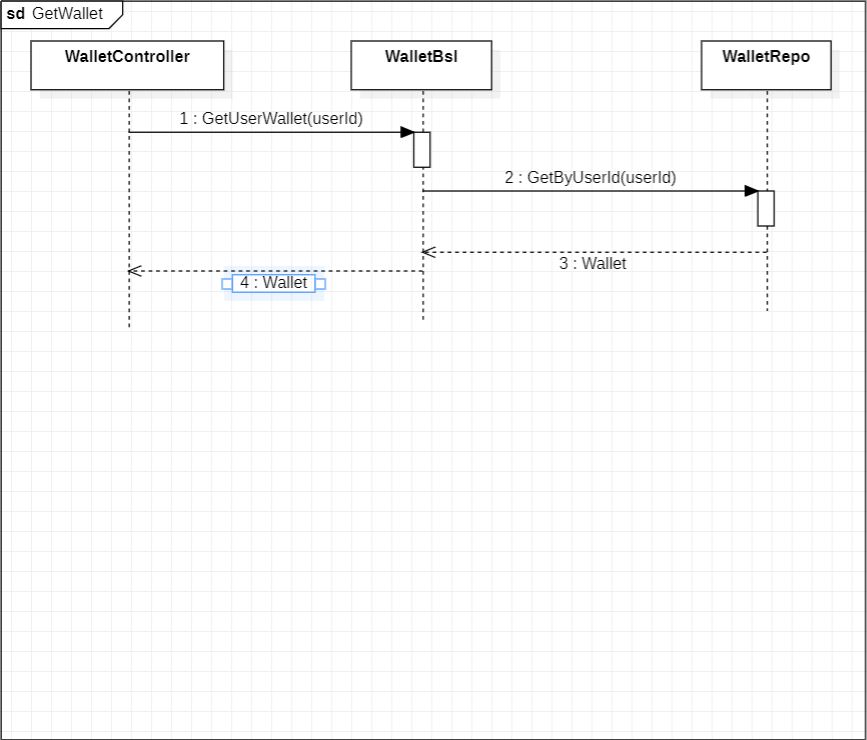
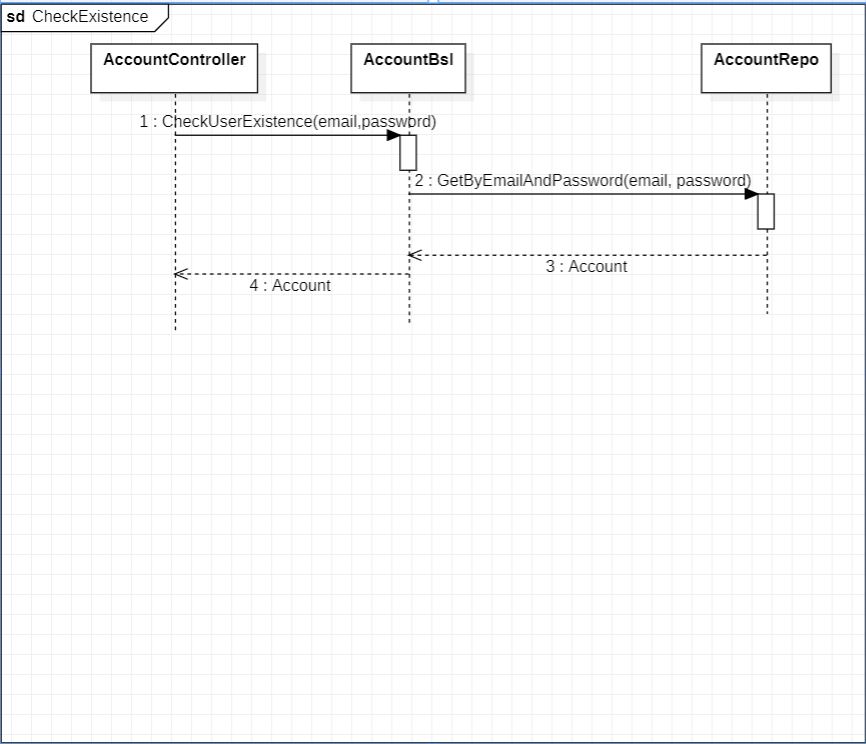
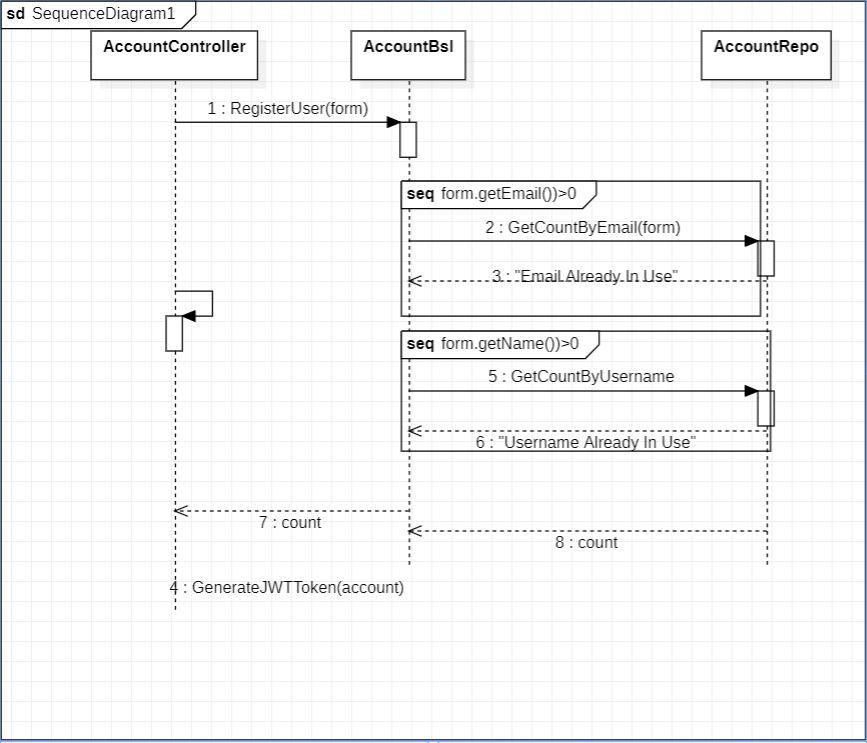
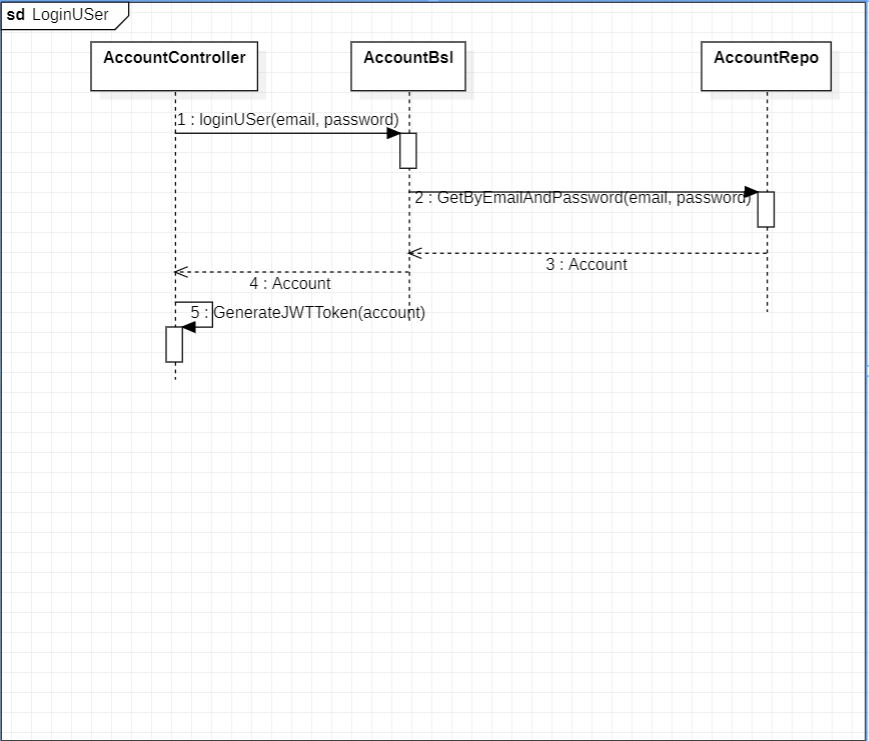
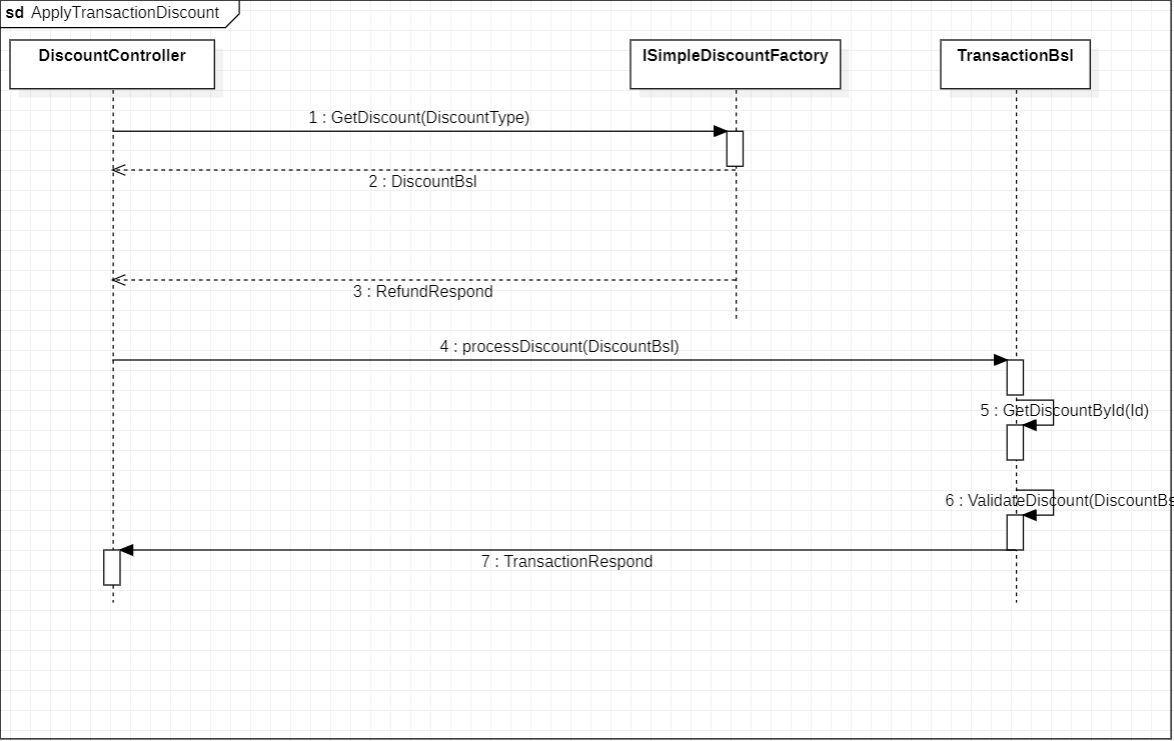
Graphical user interface, application, Word

Description automatically generated

# Class diagram Explanation

* **Strategy pattern:** 
  + **We used this pattern to set the payment method of the transaction.**
  + **We used this pattern to set the discount of the transaction and delegate its functionality.**
* **Simple Factory Pattern:**
  + **We used this pattern to initialize the type of ServiceBSL depending on the user choice.**
  + **We used this pattern also to initialize the PaymentMethodBSL depending on the choice of the user.**

# Sequence diagram design



# Requirements Exposure as Web Service API

**Part 1: Exposed Postman Collection**

* **https://api.postman.com/collections/25130992-724c1a36-ee92-4fbc-9edd-3f63d6ddccb9?access\_key=PMAT-01GNQM2S8FNYTPWBXF1RMSVDCV**

**Note:**

**For all APIs to work, you must provide the Authorization header and token in the request header, to get the Authorization token to you need to login or sign up.**

**To Add the Authorization Header:**

**Header Name: Authorization**

**Value: Bearer “token”**

|  |  |  |
| --- | --- | --- |
| Requirement |  | Exposed API |
| The user should be able to sign-in to the system. |  | 1-POST/Account/Login A login service. This service returns user account info.  Input: email , password  Example:  URL: <http://localhost:8080/Account/Login>  {      "email":"admin@gmail.com",      "password":"admin"  }  {      "email":"ahmed@gmail.com",      "password":"123456"  } |
| The user should be able to sign up to the system |  | 2-POST/Account/Register A Register service. This service returns user account info.  Input: name, username, email, password  Example:  URL: <http://localhost:8080/Account/Register>  {      "name" : "Ali ahmed",      "username" : "ali",      "email" : "ali@gmail.com",      "password" : "123456"  } |
| The system should check if the username or the email is registered before. |  | 2-POST/Account/CheckExistence A service to check if the user exists or not. This service returns all user info if exists  Input: email and password.  Example:  URL: <http://localhost:8080/Account/CheckExistence>  {      "email" : "ahmed@gmail.com",      "password" : "123456"  } |
| The user can process any service provider form |  | 4-POST/Service/ A service to process a provider service form. This service returns a form respond.  Input: form request |
| The user can process a Mobile service provider form |  | 4-POST/Service/MobileService/ProcessService/{MobileServiceProviderType} A service to process a mobile service provider form. This service returns a form respond.  Input: form request  URL: <http://localhost:8080/Service/MobileService/ProcessService/Vodafone>  {      "type" : "Vodafone",      "amount" : 10.5,      "mobileNum" : "01012345678"  } |
| The user can process an Internet service provider form |  | 4-POST/Service/InternetService/ProcessService/{InternetServiceProviderType} A service to process an Internet service provider form. This service returns a form respond.  Input: form request  Example:  URL: <http://localhost:8080/Service/InternetService/ProcessService/We>  {      "type" : "We",      "amount" : 10.5,      "serviceNum" : "0233516485"  } |
| The user can then start the transaction And choose the type of payment |  | 4-POST/Transaction/CreateTransaction A service to create user transaction and set the payment method. This service returns the transaction.  Input: transactionRequest (serviceId,PaymentMethodType)  Example:  URL: <http://localhost:8080/Transaction/CreateTransaction>  {      "serviceId" : 1,      "paymentType" : "Wallet"  } |
| The user can proceed to start the transaction |  | 4-GET/Transaction/ProceedTransaction/{transactionId} A service to proceed the user transaction. This service returns the transaction.  Example:  URL: <http://localhost:8080/Transaction/ProceedTransaction/:transactionId>  transactionId = 1 |
| The user can ask for a refund for any complete transaction to any given service. |  | 5-POST/Refund/CreateTransactionRefund/{transactionId} A service to ask for a refund for any complete transaction. This service returns the refund.  Example:  URL: <http://localhost:8080/Refund/CreateTransactionRefund/:transactionId>  transactionId = 1 |
|  |  | 6-POST/Service/AddToWalletService/ProcessService A service to add funds to the wallet form. This service returns the respond form with the wallet and credit card.  Input: Form Respond  Example:  URL: <http://localhost:8080/Service/AddToWalletService/ProcessService>  {      "amount" : 30,      "type" : "AddToWallet",      "walletId": 1,      "creditCardId" : 2  } |
| The system maintain a wallet balance for each user. |  | 6-GET/Wallet/GetWallet A service to get the user wallet. This service returns the wallet.  Example:  URL: <http://localhost:8080/Wallet/GetWallet> |
| The user should be able to select any discount for any service in the system. |  | 7-POST/Discount/ApplyTransactionDiscount A service to add a discount to a transaction. This service returns the updated transaction after the discount.  Input: discountRequest(transactionId,discountId,discountType)  Example:  URL: <http://localhost:8080/Discount/ApplyTransactionDiscount>  {      "transactionId" : 1,      "discountId" : 1,      "type" : "Overral"  } |
| The admin should be able to list all user transactions. |  | 3-GET/Transaction/GetUserTransactions/{userId} A service to get all user transactions(payment,addToWallet,Refund). This service returns a list of transactions.  Example:  URL: <http://localhost:8080/Transaction/GetUserTransactions/:userId>  userId = 1 |
| The admin should be able to list all refund requests. |  | 4-GET/Refund/GetPendingRefunds A service to get all pending refund requests (has not processed yet). This service returns a list of all pending refund requests  URL: <http://localhost:8080/Refund/GetPendingRefunds> |
| The admin should be able to process (accept or reject) any refund request |  | 4-POST/Refund/ProcessRefund A service to process (accept or reject) a refund request. The service returns the updated refund.  Input: RefundRequest(id,status)  URL: <http://localhost:8080/Refund/ProcessRefund>  {      "id" : 1,      "status" : "Accepted"  } |

# Github repository link

* **https://github.com/khaleddoosama/Fawry-system**