Cairo University Faculty of Computers and Artificial Intelligence



**Software design specification document**

**2022**

**Project Team**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Email** |
| 20206111 | Youssef Hussein Mahmoud | [youssefhusseinwahba@gmail.com](mailto:youssefhusseinwahba@gmail.com) |
| 20206092 | Youssef Ashraf Mohammadi |  |
| 20206042 | Omar Sayed Ramadan |  |
| 20206014 | George Tadros Emil |  |

Contents

[Class diagram design 2](#_Toc120811427)

[Class diagram Explanation 3](#_Toc120811428)

[Sequence diagram design 5](#_Toc120811429)

[Github repository link 11](#_Toc120811430)

# Class diagram design

Diagram, engineering drawing

Description automatically generated

# Class diagram Explanation

Design patterns used :

1. **MVC** using model view controller to represent persistence layer which contains system models (persistence layer package), view representing spring controllers, controller using business layer package.
2. **Strategy pattern** in IPerson implementing from it Admin and User - ITransaction implementing from it WalletfundTransaction, RefundTransaction and PaymentTransaction classes – IPayment implementing from it CreditCard and Wallet classes – IProviders implementing from it Etisalat, Vodafone and We – IForm implementing from it VodafoneCashForm, EtisalatInternetForm and LandlineForm – IField implementing from it different form field initially TextFiled and DropDownMenu – IHandler implementing from it different handlers for each single service initially IVodafoneCashHandler, ILandlineHandler and IEtisalatInternetHandler.
3. **Abstract factory pattern** in EtisalatInternet, Landline and VodafoneCash (Factories) which creating objects of IHandler, IForm and IProvider.

# Subsystem decomposition diagram

Diagram, engineering drawing

Description automatically generated

# Sequence diagram design

1. Accept refund request of a user (by admin).

Diagram, schematic

Description automatically generated

1. Add overall discount for first transaction only (by admins).

Diagram

Description automatically generated

1. Request refund for a done transaction (by user).

Diagram, schematic

Description automatically generated

1. Add specific discount on a service(here for example on Etisalat internet payment).

Diagram

Description automatically generated

1. Add to wallet funds from credit card (by user).

Diagram, schematic

Description automatically generated

1. Pay for a service (here for example for Etisalat internet payment).

Diagram, engineering drawing

Description automatically generated

1. Reject refund request of a user (by admin).

Diagram

Description automatically generated

1. User login (by user).

Diagram

Description automatically generated

# Requirements Exposure as Web Service API

**Part 1:**

below the link of our collection on postman on our public workspace. You can easily fork it to your workspace and use it with all its data.

[**https://elements.getpostman.com/redirect?entityId=22241636-50f466e6-2c1d-4114-8db0-19c2c6661383&entityType=collection**](https://elements.getpostman.com/redirect?entityId=22241636-50f466e6-2c1d-4114-8db0-19c2c6661383&entityType=collection)

below link drive including the JSON file to be imported directly to your workspace either of the 2 methods work fine

<https://drive.google.com/file/d/1wXDotiCWrP0TF3FoLUx5hgTpuhZQBbQ_/view?usp=sharing>

**Part 2:**

Below we exposed all APIs that we are using except for the different services as all of them intersect in the same functionalities. For example, we exposed APIs related to “Etisalat internet Service” you can find in the postman collection the different APIs for different services we implemented. We didn’t implement all the services in the description as they can be added easily at any time due to the reliable and scalable design that we tried to implement. Here we implemented only these services [Etisalat internet payment, Vodafone cash pay, landline pay]. Other services can be added easily by the time to make the design simpler and not be complex.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Requirement** | | **Exposed API** | | **Service explanation** | **Body or path variable example** | |
| The user should be able to sign-in to the system. | | POST /users/login | | Service to login the user to the system make the user able to use system functionalities by signing in the system using email and password.  -input in body as JSON-  email, password | Body:  {      "email":"youssef@gmail.com",      "password":"youssef"  }  Well tested if any credentials missing or not existing user. | |
| The user should be able to sign up to the system. | | POST /users/register | | Service to register an un registered user to the system to make him able to sign in and use the system functionalities.  -Input in body as JSON-  Username, email, password | Body:  {      "username":"ah",      "email":"ah@gmail.com",      "password": "ahmed"  }  Well tested if any credentials missing or already existing user. | |
| The user should be able to search for any service in the system. | | POST  /services/search/{query} | | Service to make the user able to search for any service in the system.  -input in path variable-  User query | Path variable:  services/search/Vodafone  return all services containing this query.  well tested if not found. | |
| The user can pay for any service in the system by default way (credit card). | | POST  /etisalat-internet  /creditCard-pay | | Service to make the user able to pay via credit card for Etisalat internet payment service (for example) by sending requirements of the service: secretInternetCode, amount (requirement for this example) | Body:  {      "secretInternetCode": "1aas71a21s8",      "amount": 200  }  Well tested if any requirement missing. | |
| The user can pay for any service in the system by wallet. | | POST  /etisalat-internet  /wallet-pay | | Service to make the user able to pay via user’s own wallet for Etisalat internet payment service (for example) by sending requirements of the service: secret Internet Code, amount (requirement for this example). | Body:  {      "secretInternetCode": "1aas71a21s8",      "amount": 200  }  Well tested if any requirement missing. | |
| The user can ask for a refund for any complete transaction to any given service. | | POST /refund/add-refund/{index}  Index = the index of the transaction of user’s done transactions  (e.g., 1st transaction so index = 1 etc.). | | Service to make the user able to request for refund of his money from any paid service before to be sent to the admin to be either accepted and return refund to user’s credit card or rejected the request. | Path variable:  /refund/add-refund/1  Request refund of first transaction done either by credit card or wallet.  Well tested if wrong index sent. | |
| The user should be able to add any funds to the wallet. | | POST /wallet-funds/add-funds/{amount}  Amount of money to be withdrawn from credit card to the wallet (in double). | | Service to make the user able to add funds to his own wallet balance from credit card if enough balance found in credit card. | Path variable:  /wallet-funds/add-funds/100  Add 100$ to the wallet from credit card.  Well tested if there is not enough balance in credit card. | |
| The admin should be able to add Overall discounts for the first user’s transaction only. | | POST /admins/overall-discount/{percentage}  Percentage is given (e.g., 50% = 0.5). | | Service to make the user accept discount percentage, by admin, on his first transaction only regardless to the service type. | Path variable:  /admins/overall-discount/0.5  Apply 50% discount on first user’s transaction regardless to the service type.  Well tested. | |
| The admin should be able to add specific discounts for a given service. | | POST /etisalat-internet/add-discount/{percentage}  For example, adding to Etisalat internet service a specific percentage of discount for the user. | | Service to make the user accept discount percentage, by admin, on specific type of transactions only. | Path variable:  /etisalat-internet/add-discount/0.5  Apply 50% discount on Etisalat internet payment.  Well tested. | |
| The admin should be able to list all user’s payment transactions. | | GET /payment-transaction/list  Done by admins only to list all user’s payment transactions. | | Service to make the admin list all user’s payment transactions for all services. | /payment-transaction/list  Well tested. | |
| The admin should be able to list all user’s add to wallet transactions. | | GET /wallet-funds/list  Done by admins only to list all user’s add to wallet transactions. | | Service to make the admin list all user’s add to wallet transactions | /wallet-funds/list  Well tested. | |
| The admin should be able to list all user’s refund transactions requests. | | GET /refund/list  Done by admins only to list all user’s refund transaction requests. | | Service to make the admin list all user’s refund transactions requests. | /refund/list  Well tested. | |
| The admin should be able to accept users’ refund requests for any service. | | POST /refund/accept/{index}  Index = the index of the refund transaction requested by the user (e.g., 1st transaction so index = 1 etc.) | | Service to make the admin to accept the users’ refund requests and returning the refund balance to the user’s credit card. | /refund/accept/1  Accepts first request refund listed from /refund/list endpoint.  Well tested if index not found or no refund requests found. | |
| The admin should be able to reject users’ refund requests for any service. | | POST /refund/reject/{index}  Index = the index of the refund transaction requested by the user (e.g., 1st transaction so index = 1 etc.). | | Service to make the admin to reject the users’ refund requests and removing the request from the list. | /refund/reject/1  Rejects first request refund listed from /refund/list endpoint.  Well tested if index not found or no refund requests found. | |
| **All the previous endpoints are the required from our phase the following endpoints are created to help us in testing and showing our results to make the project more realistic.** | | | | | | |
| Get all users in the system  (Admin only). | GET /users/list | | Service to make the admin able to list all users’ information including their wallet and credit cards balance. | | | /users/list  Well tested. |
| Signs out any user from the system  (User and admin). | GET /users/signout | | Service to make either the user or admin to log out from the system. | | | /users/signout  Well tested. |
| List all services available in the system (User and admin) | GET /services/list | | Service to make any user or admin who is already signed in to list all services in the system (here we implemented 3 services to be able to make the design more simple and not filled with too much services and it is too easy to add any service easily at any time) | | | /services/list  Well tested. |
| Log in the admin to the system (Admins only). | POST admins/login | | Service to log in the admin to his account and use his own functionalities in the system  -input in body as JSON-  Email,password | | | Body :  {      "email":"y@gmail.com",      "password":"11111"  }  Well tested. |
| List all admins in the system (Admins only). | GET /admins/list | | Service to list the admins in the system. | | | /admins/list  Well tested. |
| Removes any added overall discount (Admins only). | GET /admins/overall-discount/remove | | Service to make the admin able to remove overall discounts from the system | | | /admins/overall-discount/remove  Well tested. |
| Get all signed in user payment transaction to help him send the index of the transaction if he needs a refund (users only). | GET /payment-transaction/user-transactions | | Service to make the user able to list all his transactions | | | /payment-transaction/user-transactions  Well tested. |
| Remove the specific discount from a certain service in the system (Admins only). | GET /etisalat-internet/remove-discount | | Service to make the admin able to remove the specific discount on certain service (here we remove specific discount from the Etisalat internet service and for each service has its own endpoint to do such functionality). | | | /etisalat-internet/remove-discount  Well tested. |
| Lists the requirements of a specific service (users and admins). | GET /Etisalat-internet/list-requirements | | Service to make users and admins to list requirements of a service to complete a transaction to it (here we list requirements for Etisalat internet service and for each service has its own endpoint to do such functionality) | | | /Etisalat-internet/list-requirements  Well tested. |

# Github repository link

* **URL:** [**https://github.com/Youssef-Wahba/Fawry-Demo**](https://github.com/Youssef-Wahba/Fawry-Demo)
* **GIT**: https://github.com/Youssef-Wahba/Fawry-Demo.git