Cairo University Faculty of Computers and Artificial Intelligence



**Software design specification document**

**2022**

**Project Team**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Email** |
| 20200457 | Mohamed Abdelaal Mahmoud | mohamed.abdelaaaaaal@gmail.com |
| 20200272 | Tarek Mohamed Amin | desiretarek@gmail.com |
| 20200032 | Ahmed Qadry Taha | ahmedqadry72@gmail.com |
| 20200539 | Mustafa Saad Abdel Aleem | mustafaaleem757@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

Diagram, schematic

Description automatically generated

# Class diagram Explanation

* **Singleton: to Create one instance as a database for system** **(Entity)**
* **Factory: to allows the sub-classes to choose the type of objects to create When a class doesn't know what sub-classes will be required to create away from the user)ٍServices-Landline-Mobile-Internet-Donation-Ifactory-ServiceFactory)**
* **Decorator: to apply discounts to amount (payment-Cash-Wallet-Discount-FirstDiscount-SpecDiscount)**

# Sequence diagram design

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram, schematic

Description automatically generated

Diagram, schematic

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

# Requirements Exposure as Web Service API

**Part 1:** <https://api.postman.com/collections/25139825-78fc8fdf-80a4-46c2-956f-4b11b4b4e988?access_key=PMAT-01GNPTZQM66SREQ1VTF0AEMECK>

**Part 2:**

**Explain here the exact mapping between every single requirement and its corresponding web service API operation. A sample example is provided to better explain the concept.**

|  |  |  |
| --- | --- | --- |
| Requirement |  | Exposed API |
| 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 |  | - POST AddClient  -http://localhost:8080/signUp  -A service to check if the user email or username already exists or not. If exists, this service return a message (“This email or username is used try new one”) else (“Add successfully”)  Input: email, password and username.  -Body->raw->json  Example:  {   "email":"mohamed@gmail.com",   "password":"123456",   "username":"mohamed"  } |
| The user can login to the system and use any of the system functionalities. |  | - POST Client\_login  -http://localhost:8080/ClientLogin  -A service returns all user information if exists else return 404 Not Found  Input: email and password  -Body->raw->json  Example:  {   "email":"mohamed@gmail.com",   "password":"123456"  } |
| 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. |  | - GET Search  -http://localhost:8080/Search  -A service return all matched services to user input  Input: Service\_name  -Body->raw->json  Example:  {      "Service\_name":"mobile"  } |
| The user can pay for any service in the system.The user should be able to check any discount for any service in the system. |  | - POST AddService  -http://localhost:8080/addService  -A service take email and password to check if user logged in system  Then service name, provider name, amount of money, discounts if available and payment and return all transaction information  Input: email, password, serviceName, providerName,paymentMethod,amount,  firstDiscount, SpecialDiscount  -Body->raw->json  Example:  {      "email":"mohamed@gmail.com",      "password":"123456",      "serviceName":"mobile",      "providerName":"we",      "firstDiscount”: true,      "specialDiscount":true,      "paymentMethod":"creditCred",      "amount":100.0  } |
| The user should be able to add any funds to the wallet. Adding funds to the wallet should be done via credit card. |  | - POST AddWallet  -http://localhost:8080/AddWallet  - A service take email and password to check if user logged in system, then amount of money(balance) want to add and serial number of credit card  Input:email,password,balance and serialNumber  -Body->raw->json  Example:  "email":"mohamed@gmail.com",      "password":"123456",      "balance":20.0,      "serialNumber":"123456789" |
| The user can ask for a refund for any complete transaction to any given service. |  | - POST AddRefund  -http://localhost:8080/addRefund  - A service take email and password to check if user logged in system, then id (position) of service in user`s services and add it to refund requests and return message (“Request done”) if all inputs valid else (“Failed)  Input: email, password and id  -Body->raw->json  Example:  {      "email":"mohamed@gmail.com",      "password":"123456",      "id":1  } |
| The admin should be able to add discounts to the system. |  | - POST AddDiscount  -http://localhost:8080/AddDiscount  -A service take email and password to check if admin logged in system, then percentage of discount  Note: Admin email and password added in code in class Entity  - Body->raw->json  Example:  {   "email":"mohamed",      "password":"256",      "percentage":20.0  } |
| The admin should be able to list all payment transactions. |  | -GET GetTranzaction  -http://localhost:8080/history  -A service return all payment transactions |
| The admin should be able to list all add to wallet transactions. |  | -GET AddWalletsTranzaction  -http://localhost:8080/AddWallets  -A service return all Add to wallet transactions |
| The admin should be able to list all refund transactions. |  | -GET GetRefunds  -http://localhost:8080/refunds  -A service return all refund transactions |
| Show all user in system |  | -GET GetClients  -http://localhost:8080/clients  -A service return all Clients in system |

# Github repository link

* **https://github.com/tarkeem/software-project.git**