Cairo University
Faculty of Computers and Artificial Intelligence

Project Description
2022/2023

Version 2.0

Introduction

- In this project you will design and implement a non-trivial software system. You will practice the concepts you learned during the course.
- Project will be at least 2 phases
- In each phase we will focus on designing and implementing some requirements.
- In each phase you are required to deliver the following deliverables **through Google classroom as one zip file**, named with your student IDs as follows: **Phase2_LabGroupNumber_Phase1_StudentID1_StudentID2_StudentID3_StudentID4**. Following such naming is a MUST. If you don't follow that naming convention, your submission will NOT be graded.
- The project must:
  1. **Be developed in the Java Programming Language.**
  2. **Use Eclipse IDE for Java EE (not SE).**
  3. **Use Spring boot (you can use spring initializer https://start.spring.io/)**
  4. **Application Server: Tomcat (to be installed as an Eclipse plugin)**
  5. **Postman for testing the API.**
- The deliverables are:
  1. **Updated** class diagram
  2. **Updated** sequence diagrams for the **most complex** scenarios. The submitted sequence diagrams should be **2 x the size of the team**, where each team member would be responsible **for submitting two sequence diagrams**.
  3. Subsystem decomposition diagram
  4. Link for Git repository for the developed source code project. Your Git repository should be private and it is your responsibility to add your TA to your project Git repository to be able to assess your proper usage of Git within your project development.
  5. A table that clearly explains the exact mapping between every single requirement and its corresponding web service API operation. If you do not provide such table with clear details for **every single requirement**, the TA will NOT be able to mark your project. The TA needs to know the exact web service calls to test the project.
  6. A Postman collection for the designed webservice.
  7. Zipped copy of the source code project. Note that your submitted source code project should work properly on any TA's machine that has Eclipse IDE.
- **Deliverables 1, 2, 3, 4, 5, 6 must be part of the provided project SDS template.**
- **All seven deliverables MUST be included in your submitted zip file through Google classroom.**

If you have any questions about the project details, please use the following form to submit your questions

https://docs.google.com/forms/d/e/1FAIpQLSdmWwb5vA8ehAPs2ncYDAYj3EK9Y1e6LHlMTz-x0edjSBHb1Q/viewform

Project Logistics

1. If a team is formed out of more than 4 members, their submission will be rejected, and they would get a zero for the phase.
2. If you cannot find a team, you need to submit your name through the form https://docs.google.com/forms/d/e/1FAIpQLSdJmvI8Cy0WF8YwJmjPVPwNIZSOdX7XNJVnqqaccbZrE1iRxw/viewform

   so that we can fit you within a team

3. Y**ou CANNOT change teams** across the phases. If your project has less than 4 team members, you can merge teams **BUT** the overall team size should still 4 team members at most.

4. Academic honesty is assumed. All work submitted must be original and written by your team (Not copied from students, the net, outside sources). Plagiarism will be penalized.

   - Soon, you will be our colleague and we will be proud of you.
   - Professional conduct and practice is essential in your career.

Project Phases:

Phase	**Deliverables**	**Deadline**	**Mark**
Phase 1 submission on **the course's Google classroom**	**Design and implement Phase 1 requirements (mentioned below)** Submit all the required deliverables (mentioned above)	Dec. 2nd, 2022 Late submission is not allowed	
Phase 2 submission on **the course's Google classroom**	**Design and implement Phase 2 requirements (mentioned below)** Submit all the required deliverables (mentioned above)	Dec. 30th, 2022 Late submission is not allowed	

Project overview

Description

In this project you will work on building something similar to Fawry system. This system should be user by users to pay for different services. The initial services are

a. Mobile recharge services.

    i. Vodafone
    ii. Etisalat
    iii. Orange
    iv. We

b. Internet Payment services.
    i. Vodafone
    ii. Etisalat
    iii. Orange
    iv. We

c. Landline services
    i. Monthly receipt
    ii. Quarter receipt

d. Donations
    i. Cancer Hospital
    ii. Schools
    iii. NGOs (Non profitable organizations)

Requirements

User

1. 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.
2. 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
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.
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.
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. If the admin approves the refund then the refund process should complete successfully.
6. 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.
7. The user should be able to check any discount for any service in the system. Discounts could be added by the admin (this will be discussed later).

Admin

1. The admin should be able to add any new service provider to the system. Provider consists of a form to be sent to the user and a handler for this form. So for example Vodafone Cash provider consists of a form (Mobile number, amount) and a handler for this form (given the user answers to this form then provider will handle the user request). The form could consist of any number of field. The type of fields that are supported are text field and drop down field
2. The admin should be able to add discounts to the system. There are two types of discounts.
    a. Overall discounts. For example the user should have 10% discount for the first transaction (regardless the service)
    b. Specific discount.. For example the admin can apply 20% discount for all mobile recharge services.

    For any given service. All overall discounts and specific discounts for this service should apply.
3. 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.
4. The admin should be able to list all refund requests. Each refund request should contain the related service and the amount to be refunded. The admin should be able to accept or reject any refund request and if any refund request got accepted a refund transaction should be processed.

By now you should be able to develop web services and APIs. In this phase you are required to develop a RESTful API for this software. Hence, all the existing requirements (from phase 1) need to be exposed as a RESTful API in phase 2.

For example, to be able to achieve the second requirement from the user requirements (highlighted in yellow above), you would need to provide the following API functions within phase 2:

1- GET /user/check
A service to check if the user exists or not. This service returns all user info if exists
Input: email and password.

Note: The web service API should be structured in terms of Resources and actions. So you should have a section for each resource and each resource should contain the corresponding actions.

Evaluation Criteria

1. Properly working functionality as per the phase requirements.

2. Quality of project configuration (i.e. actual realistic usage of Git throughout the phase development by all team members)

3. Consistency between the various submitted system models.

4. Consistency between the submitted system models, and the working product.

5. Quality of the design in terms of its usage of appropriate design patterns, and SOLID principles as needed.

6. Quality of the web service API design