Object-Orientated Analysis and Design Report

Members:

Abdelrahman Mahmoud Fangary	17P6006
Ahmed Abdel Nasser Korany	17P8113
Ahmed Essam Mohamed Ramzy	17P8229
Moaaz Mohamed Abd Elaziz	17P1023
Mohamed Ahmed Hashem	17P3067
Youssef Emad Saad	17P8042

Software Requirements Specification

Functional requirement:

User requirements:

- User make an account and apply as a customer or a chef.
- Customer can manipulate his account information in all different ways.
- Customer enters his location or the area he wants, gets all the chefs and their menus operating in that area.
- Customer chooses to make an order after reviewing the available menus.
- Customer can manage his cart in all different ways, proceed to checkout and confirm payment.
- Customer can rate his experience after every dish he tried and provides any comments.
- Customer can access his cart to view order information.
- Chef uploads dishes he/she cook on the system and its details.
- Chef receives customer orders instantly.

Software specifications:

- Website provides a page for all available dishes, and filters for dishes types.
- System provides a chef name alongside with his/her rated dishes
- system provides live chat for complains and comments.
- System may provide promocodes to the customer and deals (offers, promotions and discounts).
- System provides cart management for customer.
- Checkout page .

System requirements:

- Internet connection and browser availability.

Non-functional requirements:

- database stores system information.
- System must be reliable and secure its customer data.
- System should follow country regulations

Accurate and complete UML diagrams

- All UML diagrams are placed in file deliverables as PDF.

Where and how object-oriented concepts are applied?

Encapsulation: All classes encapsulates attributes and methods, where attributes accessibility is private, and can only be accessed through public setters and getters for the principle of data hiding, reflected on class diagram and implementation.

Associations: All Transactions types associated with other classes.

Inheritance: Customer_transacations,Chef_transacations and its children types.

Polymorphism: In transacationApplication, and any abstract class.

```
public TransactionApplication(TransactionSource source) {
        this.source = source;
        // source reference to child object.
    }
public void run() {
        Transaction trans;
        while ((trans = source.getTransaction()) != null) {
            trans.execute();
            // depending of the transactions source,it calls its child
        }
```

Composition:

Customer_account with payment_method/Membership, Visa_Payment with visa.Chef_Account with Dish. Web Customer with cart.

Abstraction: Abstract classes.

Where and how SOLID principles are applied?

SRP: All classes conform to Single Responsibility Principle, example: Customer_Account.

OCP: All transactions types are open for extensions, closed for any modification in the parent.

LSP: Any inheritance model, the child doesn't remove the parent behavior.

ISP: There is no "fat" interfaces.

DIP: Architecture is layered. No dependences between layers. Associations depends on abstractions.

Where and how design patterns are applied?

Command: All transactions, where every operation, request, logs or task expressed in this pattern.

Template: excute() function is implemented in parent and inherited in children.

```
@Override
    public void execute() {
        CustomerAccount cs= db.getCustomerAccount(CustomerId);
        if (e != null) {
            change(e);
        }
    }
```

Strategy: Payment,Different strategy for payment method.

Façade: Database

Mediator: Live Chat

Factory: Factory class.

Null Object Pattern: This pattern is used in the membership interface.

Show examples of how the design is reflected to the implementation?

Code is to be checked for OOP Part. association, composition, aggregation, inheritance relationships.

Role of each team member and answering oral questions?

Every member worked on a specific diagram,

Except for the use case, class diagram, analysis and design as the whole team contributed in each part.

Code was written by the whole team.