Grocery Shopping Application
Analysis and Design Document

Cordea Corina
Group 30432

Revision History

Date	Version	Description	Author
04/04/2018	1.0	Domain Model and Architectural Design	Cordea Corina
25/04/2018	1.1	Design Model and Data Model	Cordea Corina

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 5

2.1 Conceptual Architecture 5
2.2 Package Design 5
2.3 Component and Deployment Diagrams 6

III. Elaboration – Iteration 1.2 7

1. Design Model 7

1.1 Dynamic Behavior 7
1.2 Class Design 7

2. Data Model 8

3. Unit Testing 9

IV. Elaboration – Iteration 2 9

1. Architectural Design Refinement 9

2. Design Model Refinement 9

V. Construction and Transition 9

1. System Testing 9

2. Future improvements 9

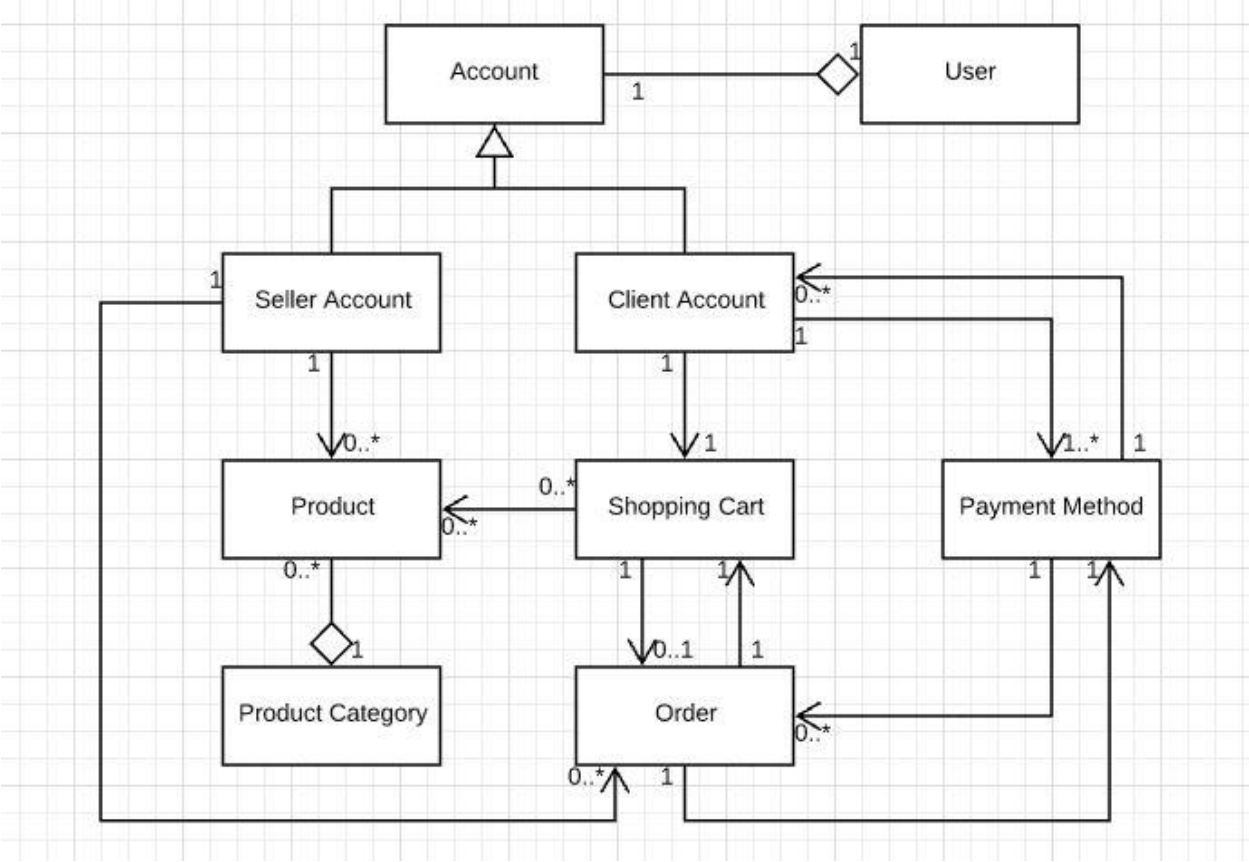
VI. Bibliography 9

I. Project Specification

The Grocery Shopping is an application where clients can purchase and order groceries online. The system is developed with a user-friendly and attractive GUI. Users have to first login into the system to view the groceries and add them into their cart. They can then order it by choosing a payment method. Also, the seller can use the application to add his products and sell them.

II. Elaboration – Iteration 1.1

1. Domain Model

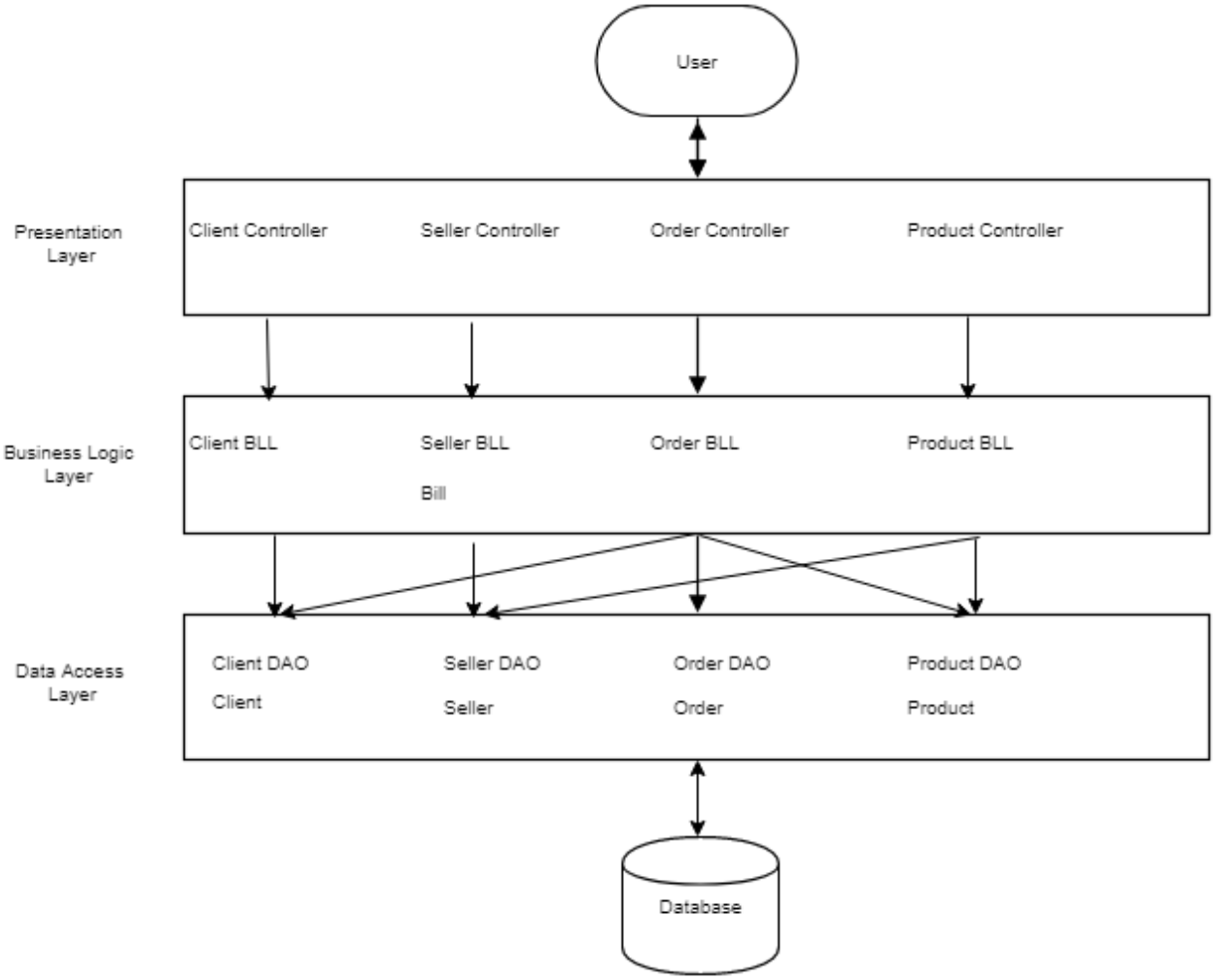


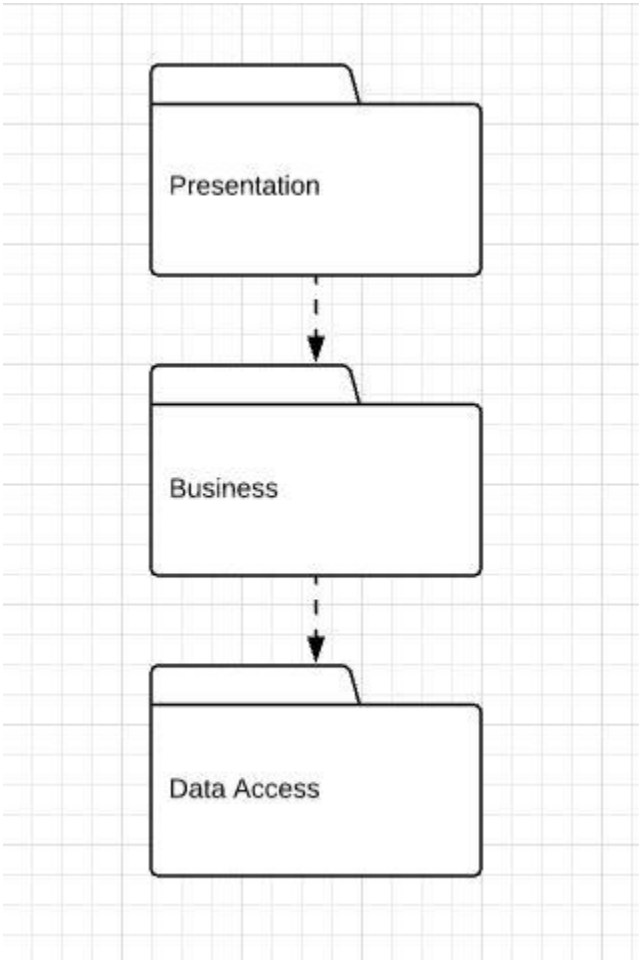
2. Architectural Design

2.1 Conceptual Architecture

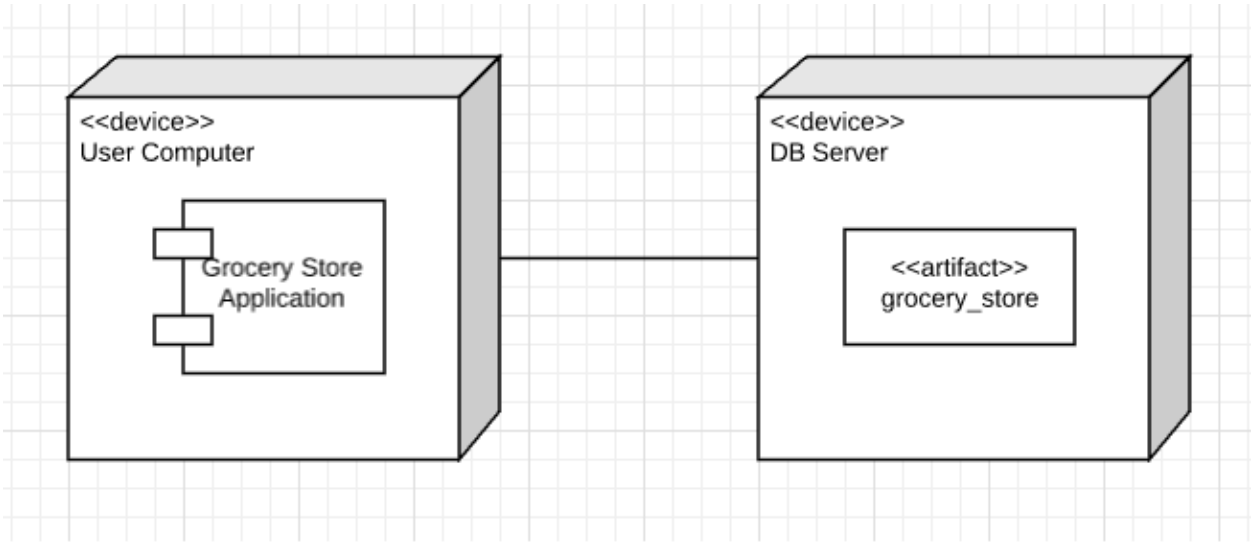
The Layers architectural pattern will be used to structure the application by dividing it into groups of subtasks based on their functional responsibility. More specifically, the application is divided in 3 layers (bottom to top): data layer which provides access to the data stored in a database, business logic layer which implements the main functionality of the system and presentation layer which contains the user oriented functionality and manages the interaction between the user and the system. Each layer can only access the one beneath it. By using this pattern, the maintainability of the application and the reusability of components are considerably increased.

2.2 Package Design





2.3 Component and Deployment Diagrams



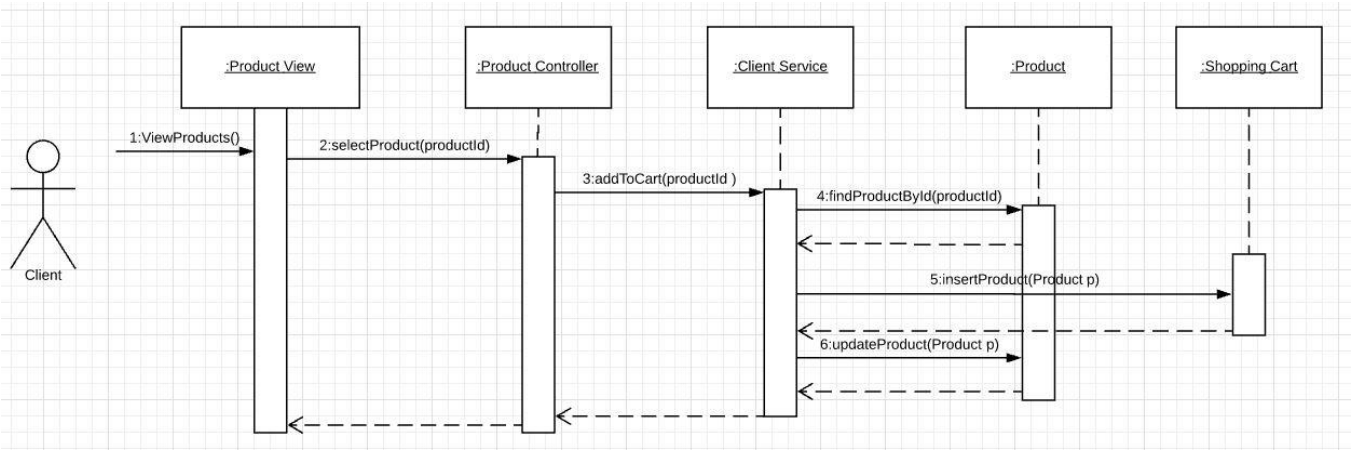
III. Elaboration – Iteration 1.2

1. Design Model

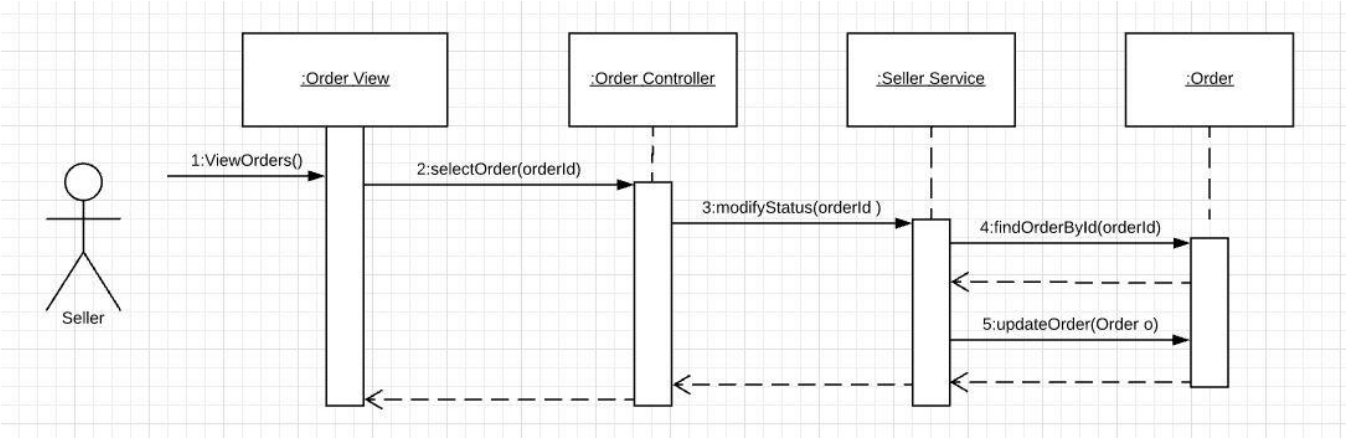
1.1 Dynamic Behavior

Sequence diagrams

- Add product to shopping cart



- Modify order status



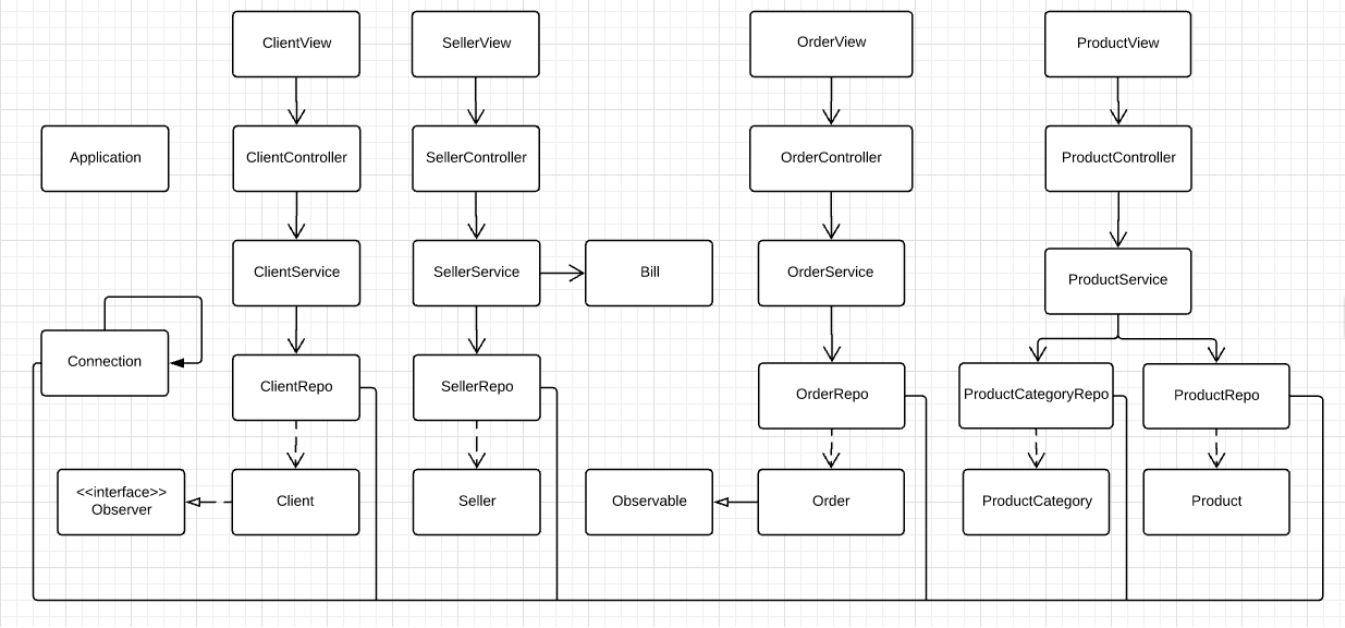
1.2 Class Design

The **observer pattern** is a software design pattern in which an object, called the **subject**, maintains a list of its dependents, called **observers**, and notifies them automatically of any state changes, usually by calling one of their methods.

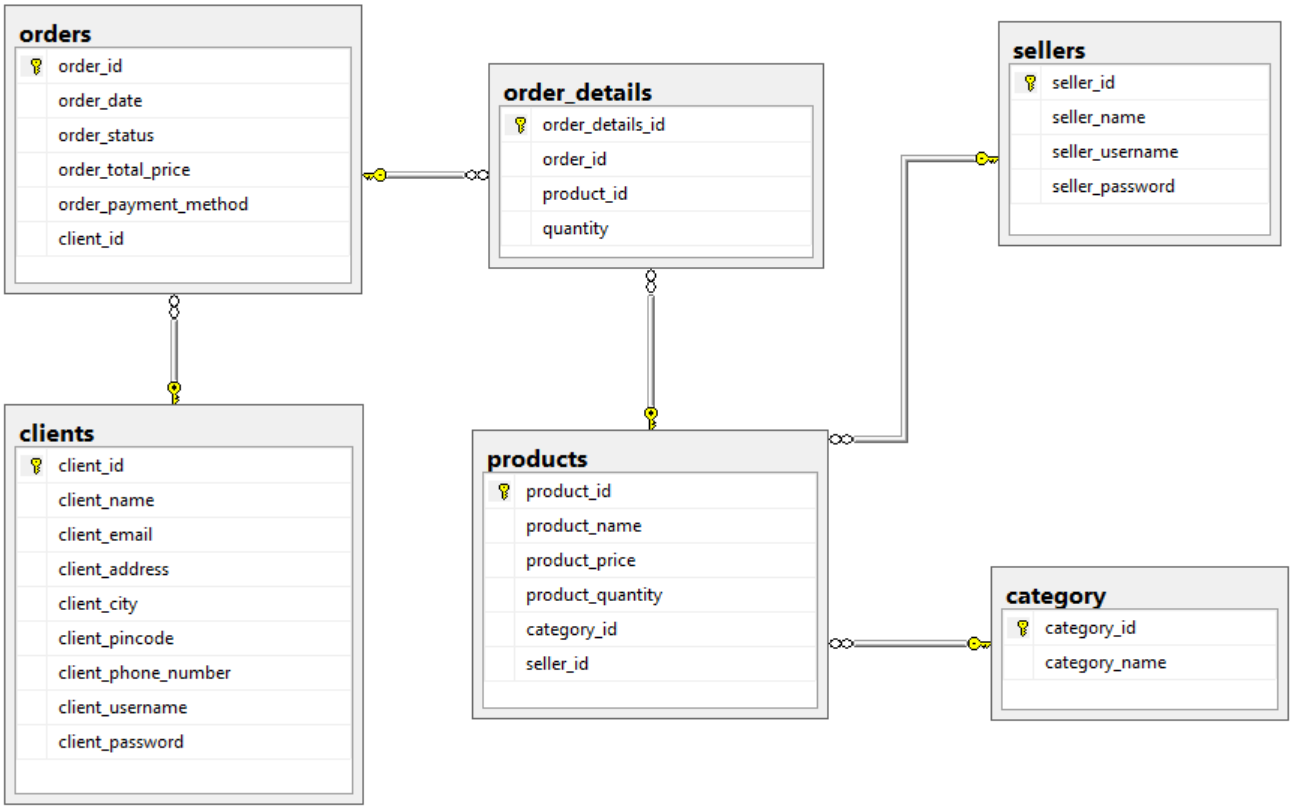
In this application, the clients are the observers, which will be notified when the status of their order is modified, and the subject is the order placed by the client.

Builder design pattern is a creational pattern that is used to simplify the creation of complex objects. It

allows us to decompose clearly the object construction by using internal builder object that passes the values to a parent class. In this application it may be used to create some complex objects, like Client.



2. Data Model



3. Unit Testing

Unit testing will be performed to test the application. This involves the testing of small individual units of code, such as methods. In order to do this, Junit and Mockito frameworks will be used.

IV. Elaboration – Iteration 2

1. Architectural Design Refinement

[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]

2. Design Model Refinement

[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]

V. Construction and Transition

1. System Testing

[Describe how you applied integration testing and present the associated test case scenarios.]

2. Future improvements

[Present future improvements for the system]

VI. Bibliography