**Online Pizza Ordering System Analysis and Design Document Student: Andreea Ionutas Group: 30432**

**Revision History**

**Date Version Description Author**

19/04/18 1.0 Andreea Ionutas

15/05/18 2.0 Andreea Ionutas

**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 6

2.3 Component and Deployment Diagrams 7

III. Elaboration – Iteration 1.2 8

1. Design Model 8

1.1 Dynamic Behavior 8

1.2 Class Design 9

2. Data Model 10

3. Unit Testing 10

IV. Elaboration – Iteration 2 10

1. Architectural Design Refinement 10

2. Design Model Refinement 10

V. Construction and Transition 10

1. System Testing 10

2. Future improvements 10

VI. Bibliography 11

1. **Project Specification**

Online Pizza ordering system is a web-based application which enables customers to order their pizzas online for home delivery or pick up from the pizzeria.

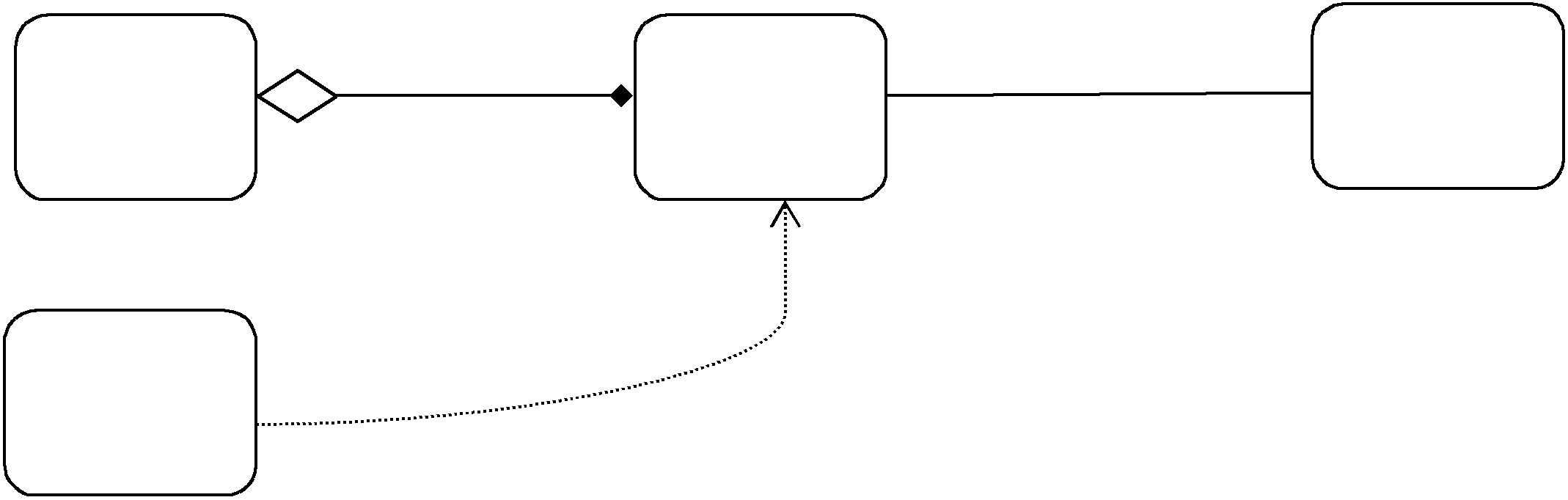
The main objectives of this ordering system are:

* Build your own pizza – This system will help customers in ordering custom pizzas. So the customer will pick exactly the things which he/she wants in their pizza. This will surely enhance the image of the pizzeria and customer satisfaction will be more.
* Reduce Paper Work- As most of the things will be performed online, it will reduce the usage of paper for the pizzeria.
* Improves Efficiency- This system will make things easier for staff as whole ordering process is done by the customer only.

**II. Elaboration – Iteration 1.1**

**1. Domain Model**

\* \*



Client Order Pizza

1 \*

Set status Edit stock

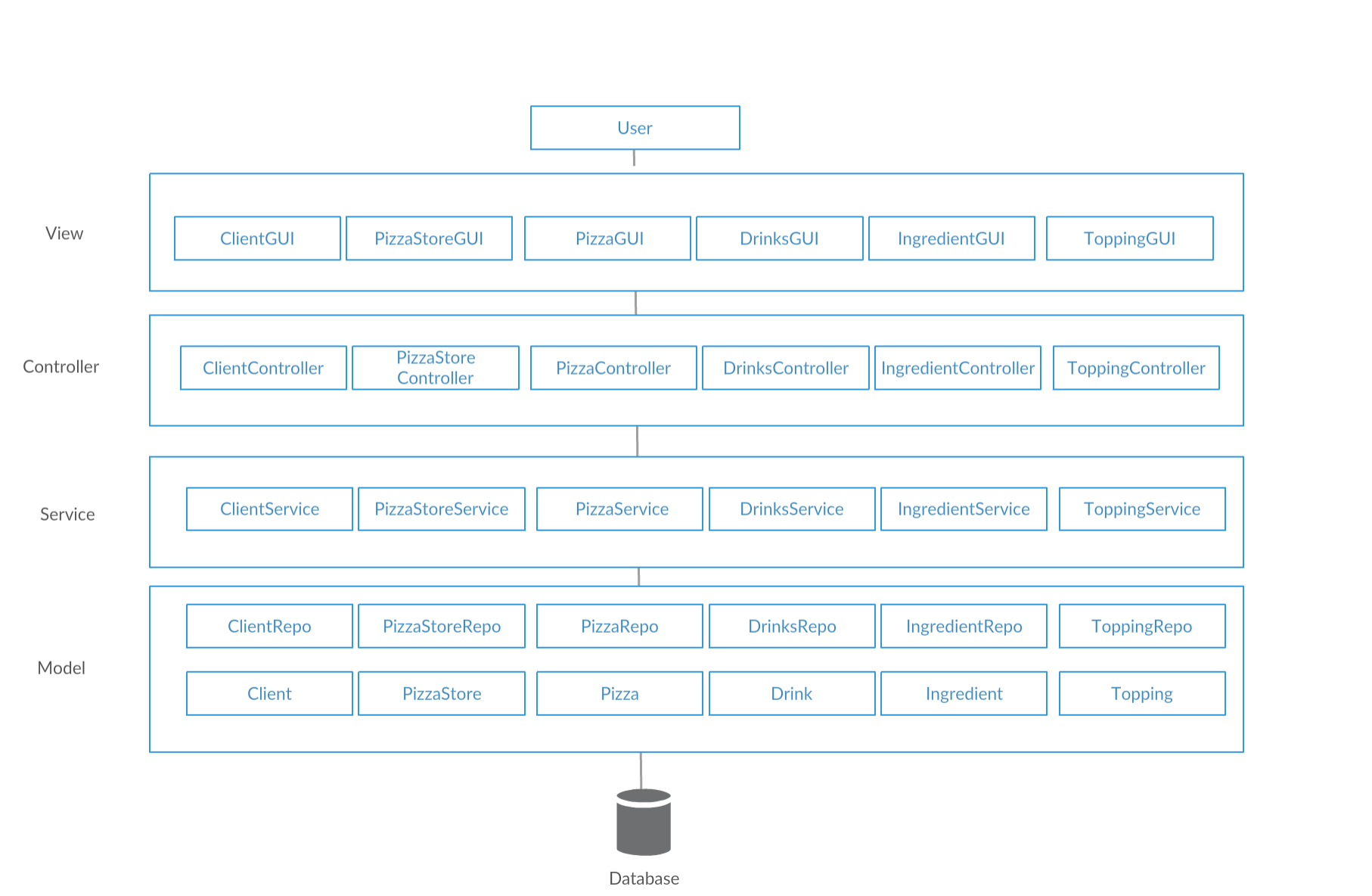
Pizza

Store

uses

**2. Architectural Design**

**2.1 Conceptual Architecture**

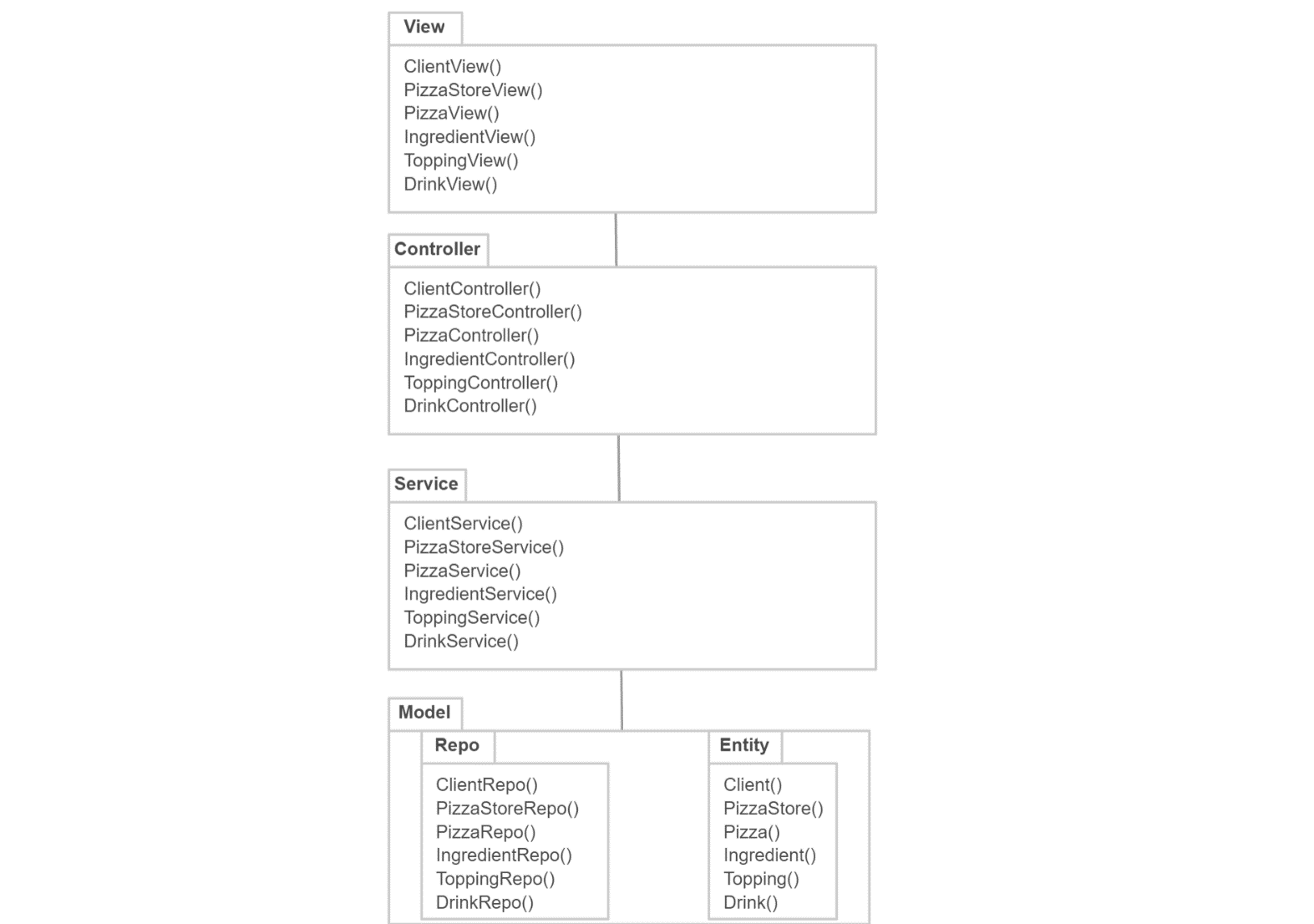


Model View Controller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts −

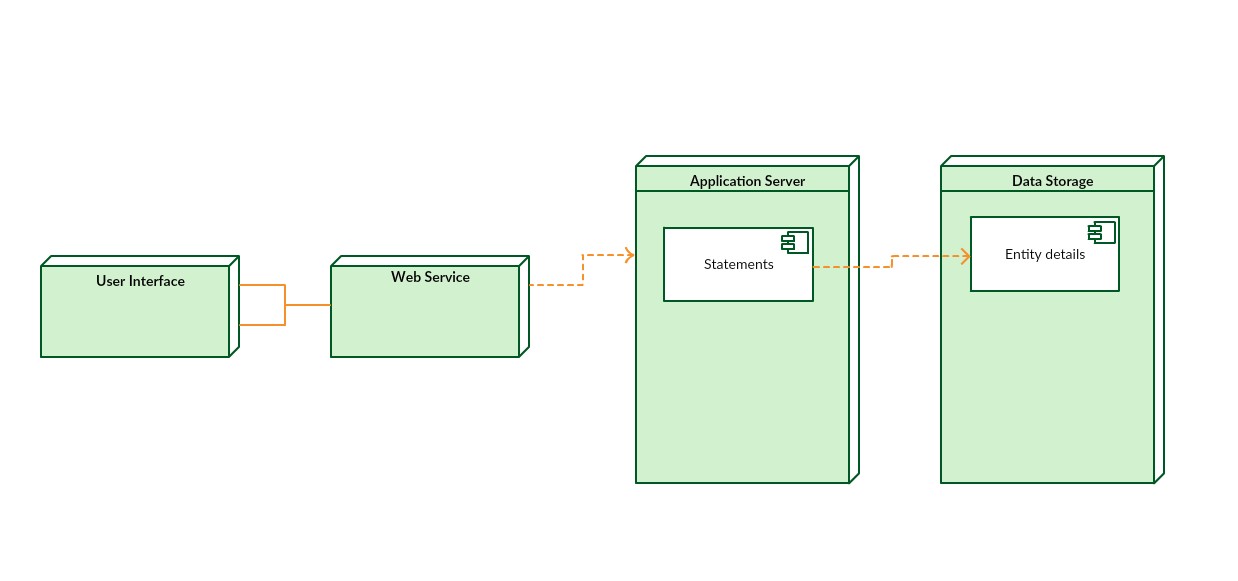
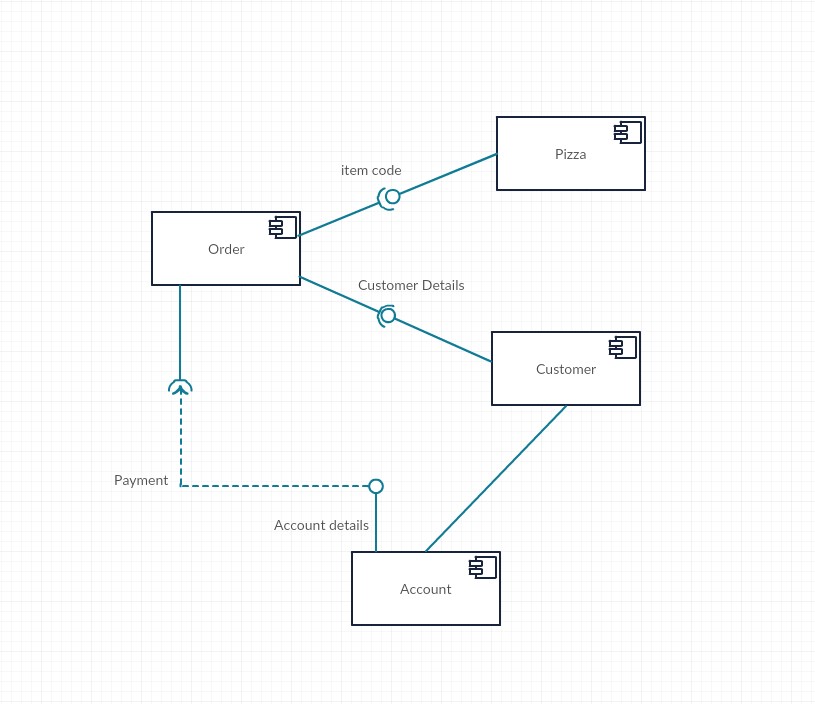
* Model − The lowest level of the pattern which is responsible for maintaining data.
* View − This is responsible for displaying all or a portion of the data to the user.
* Controller − Software Code that controls the interactions between the Model and View.

There were some refinement necessities as we needed to have access to some other entities from the DB. The MVC flow was redesign to manage the CRUD operations on all entities.

**2.2 Package Design**



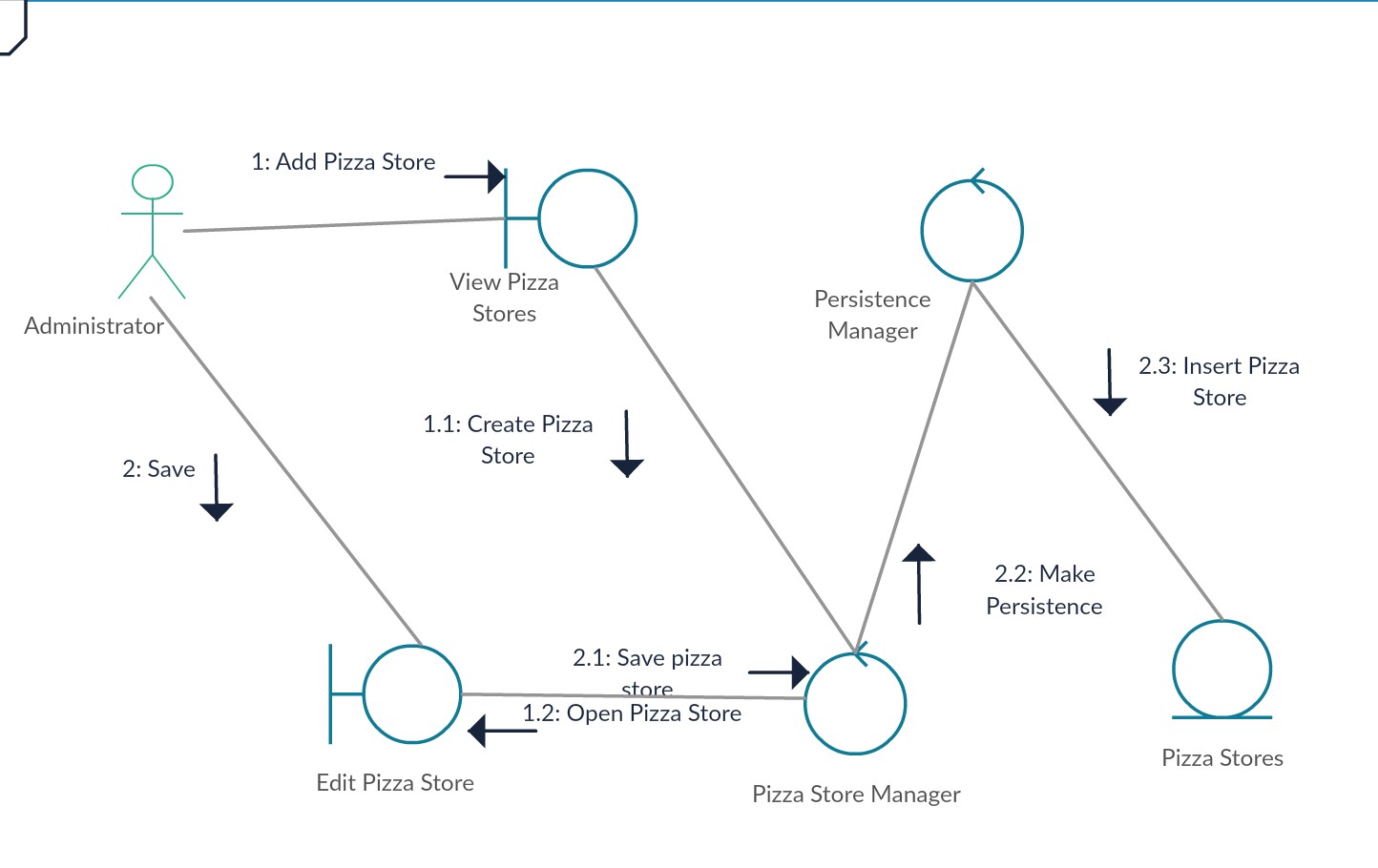
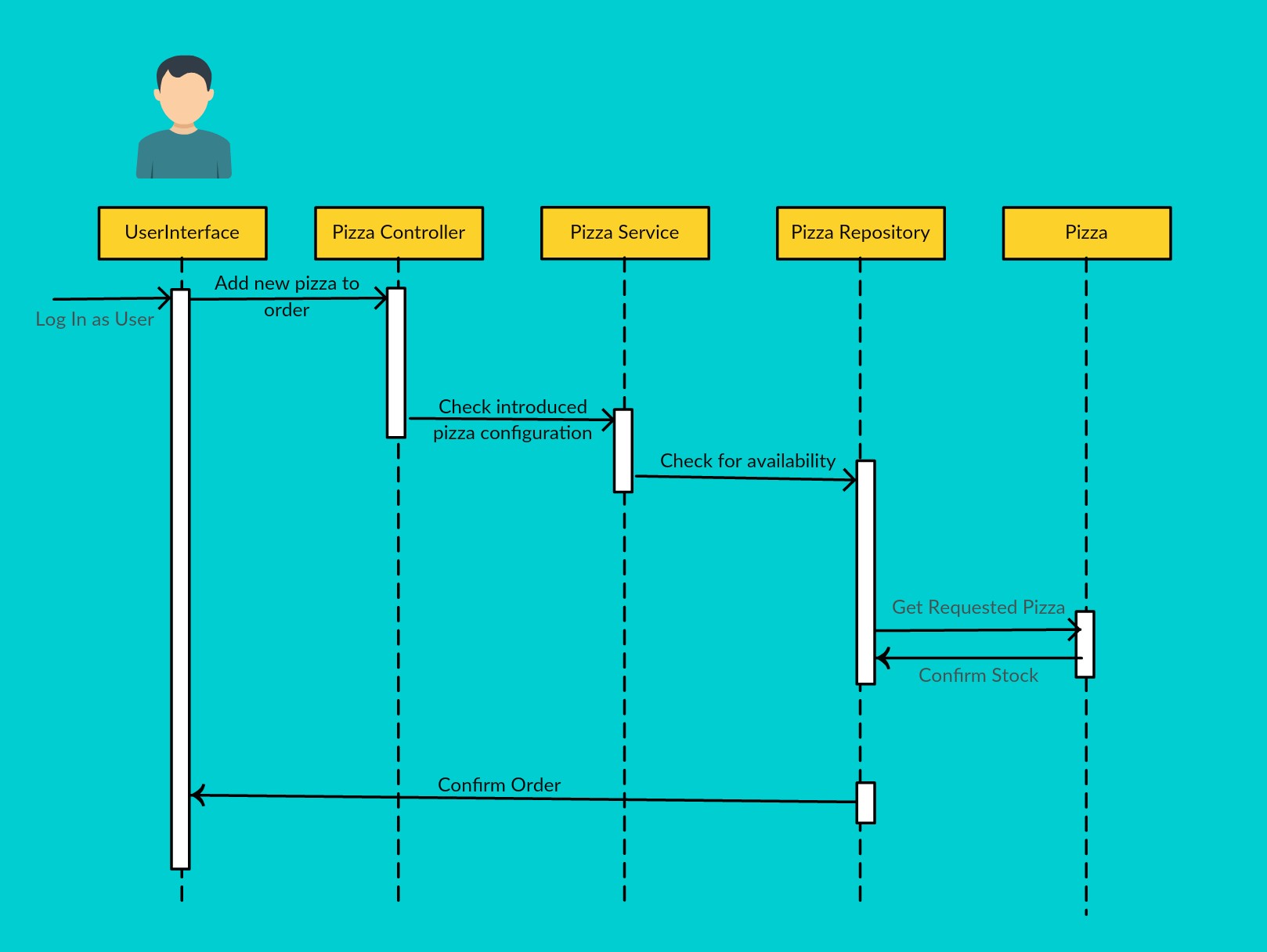
**2.3 Component and Deployment Diagrams**



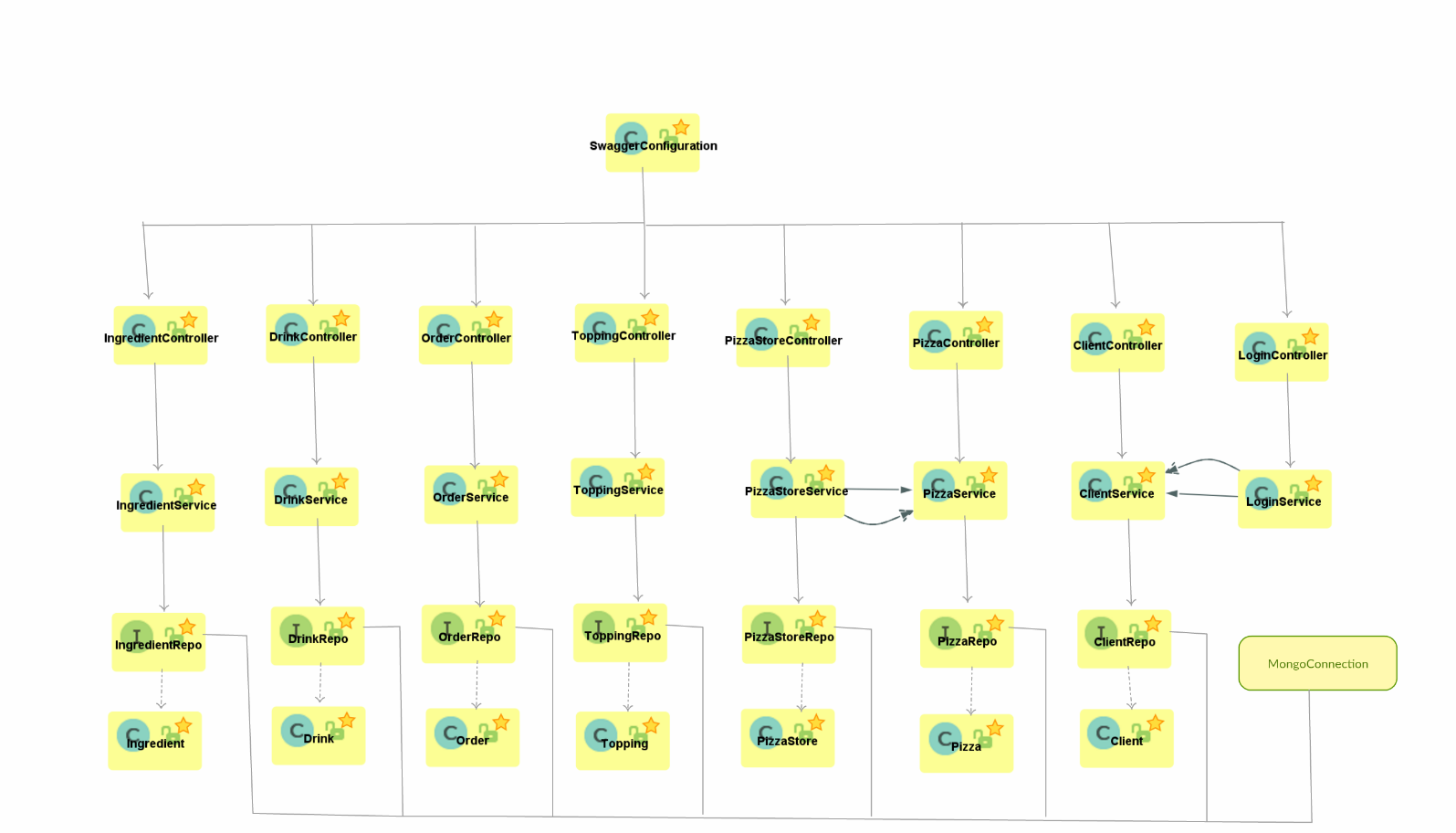
**III. Elaboration – Iteration 1.2**

**1. Design Model**

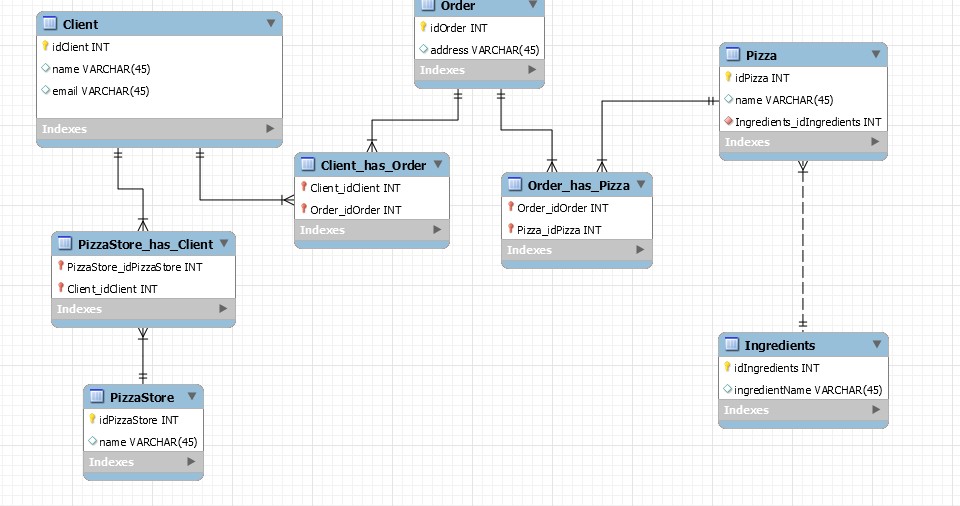
**1.1 Dynamic Behavior**



**1.2 Class Design**



**2. Data Model**



**3. Unit Testing**

The system will be tested with UnitTests and Integration Testing. For testing, Mockito will be used for mocking services.Test cases: testing some of the apis. For example, we want to test if the api which requests the findAllClients() method returns a list of clients, or if the order saved after calling the addNewOrder method matched the mocked one.

**VI. Bibliography**