<Online Bookstore>

Analysis and Design Document

Student: Craciun Sergiu

**Group:30442**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <17/03/20> | <1.0> | Version 1 | <Craciun Sergiu> |
| <19/05/20> | <1.1> | Version 1.1 | <Craciun Sergiu> |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

# The online bookstore application is a tool that helps customers to buy the books online.

# The advantages of such a system include the stock, because physical stores are restricted by

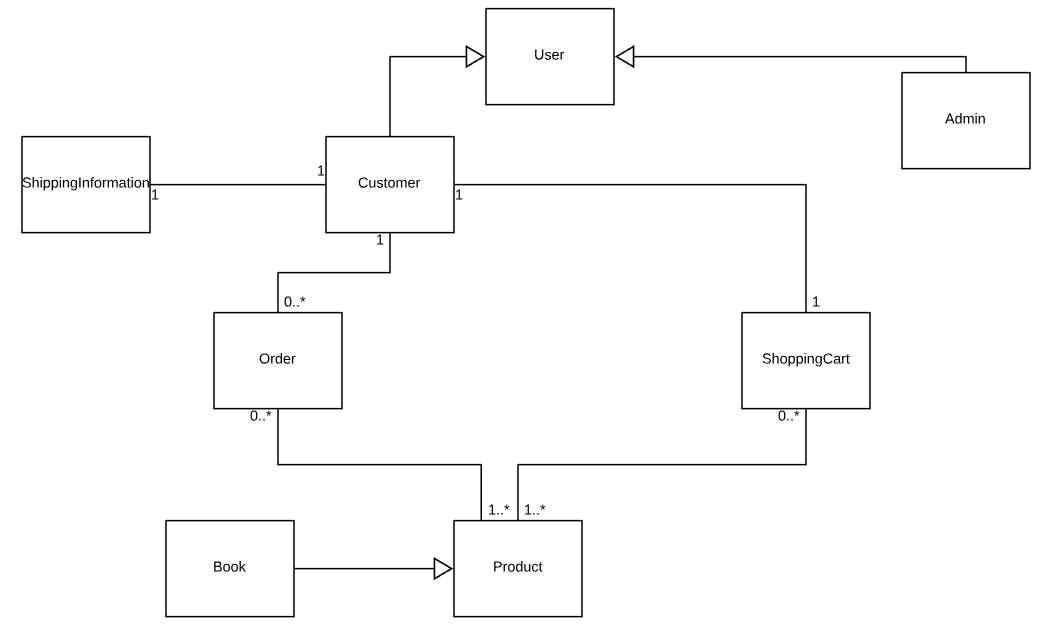
# space limitations and budgets, the convenience of that you can easily order books without the

# restrictions of specific store timings and long queues while sitting in the comfort of your home,

# but also the cost, because the customers can hunt for great deals, big discounts and free shipping.

# Elaboration – Iteration 1.1

# Domain Model



# Architectural Design

## Conceptual Architecture

The architectural pattern used is Layers. Components within this pattern are organized

into horizontal layers, each layer performing a specific role within the application. Although it

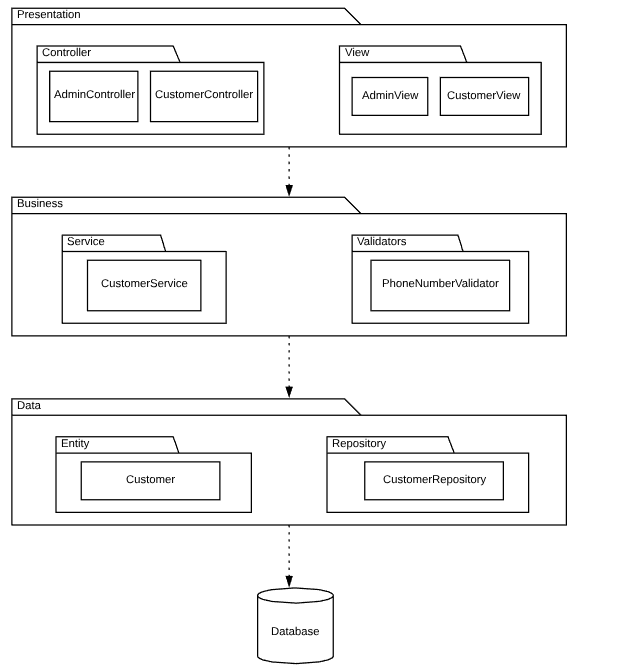
does not specify the number and types of layers that must exist in the pattern, most layered

architectures consist of three standard layers: presentation, business and database.

MVC Pattern stands for Model-View-Controller Pattern and will be used to create the presentation of the project. This pattern is used to separate application’s concerns: the Model will be used to store the data of the application; the View represents the GUI; and the Controller has the purpose of controlling the data coming from the users.

The main reason I chose to work with layers is because of the flexibility and maintainability they bring to the application. In order to be able to keep the code well organized, I also chose MVC, which will allow me to independently test and re-use the code written.

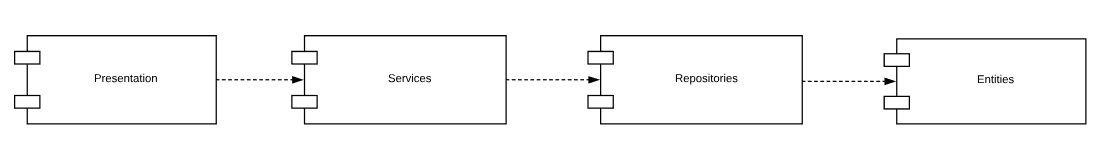
## Package Design



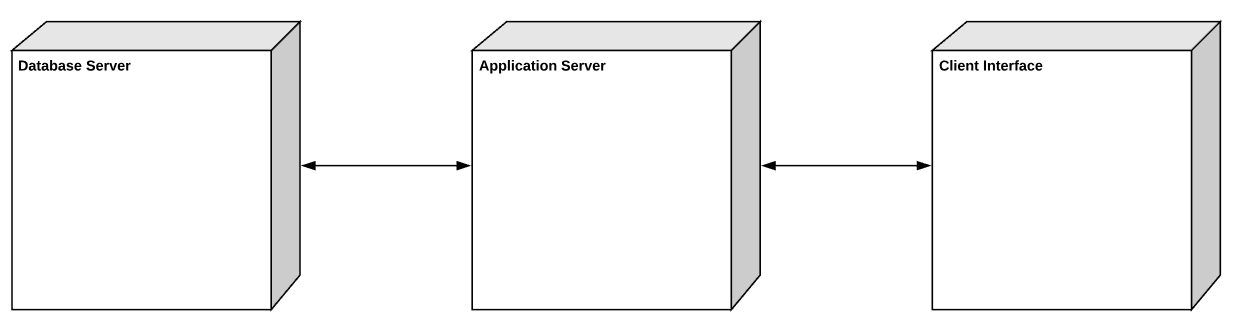
## Component and Deployment Diagrams

# Component Diagram:

]



Deployment Diagram:



# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*

# Data Model

*[Create the data model for the system.]*

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# Elaboration – Iteration 2

# 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.]*

# Design Model Refinement

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

# Construction and Transition

# System Testing

For testing the application and increasing the certainty that the system is working correctly, unit tests were written.

# Future improvements

The system can be improved in the future by adding more functionalities both to the admin and to the customer side(for example, choosing the number of the same books added to the cart).

To add flexibility, the application should be converted into a web application and security should be implemented.

# Bibliography