Video Game Shop

Analysis and Design Document

Student: Bar Luca-Narcis

**Group: 30431/1**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <11/04/23> | <1.0> | <details> | <Bar Luca-Narics > |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

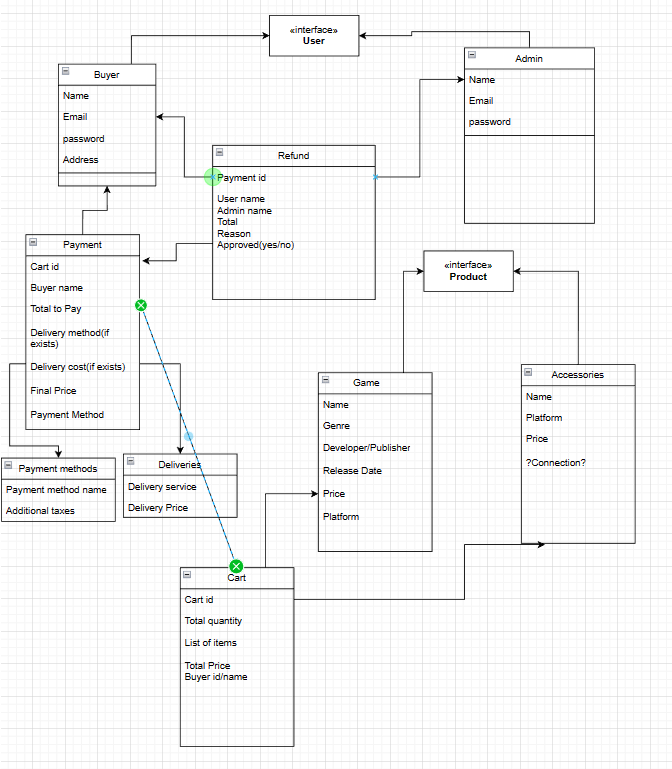
# Project Specification

*[Present the project specification]*

# Elaboration – Iteration 1.1

# Domain Model

***Conceptual Class Diagram***

******

# Architectural Design

## Conceptual Architecture

The layered architecture pattern separates the application into logical layers, with each layer responsible for a specific set of tasks. The presentation layer, which is responsible for the user interface and user input, is implemented using the MVC pattern. The Model represents the business logic and data layer, while the Controller acts as an intermediary between the Model and the View. The View represents the user interface, which displays data to the user and receives user input.

Using this architecture pattern has several benefits for my application. First, it promotes separation of concerns, which makes the application easier to develop, test, and maintain. Second, it enables scalability, as each layer can be scaled independently. Third, it provides a clear and organized structure, making it easier to understand and modify.

In summary, by using a layered architecture pattern with the MVC pattern for the presentation layer, I can ensure that the application is well-organized, maintainable, and scalable.

Diagram

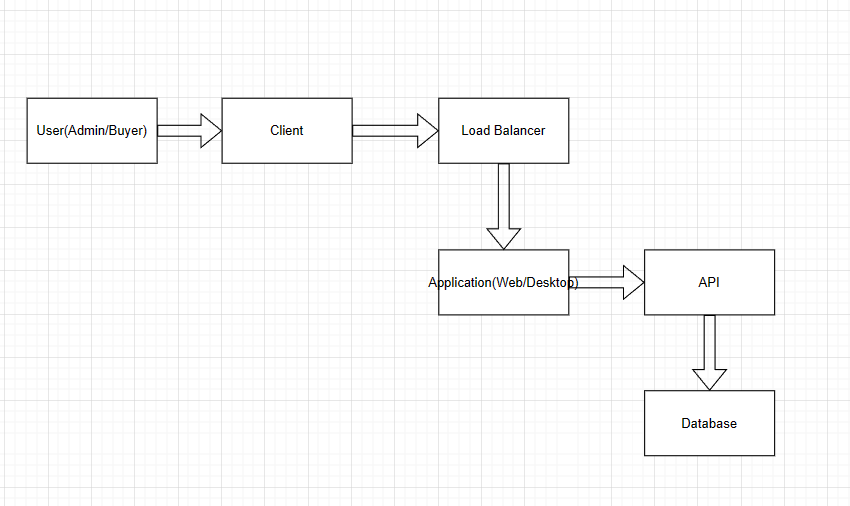
Description automatically generated

## Package Design

*[Create a package diagram]*

## Component and Deployment Diagrams

Deployment Diagram:



Component Diagram:

Diagram

Description automatically generated

# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

Scenario 1: User creates an account in the application

Sequence diagram:

Graphical user interface

Description automatically generated with medium confidence

Communication diagram:

Diagram

Description automatically generated

Scenario 2:Administrator validates a refund

Sequence diagram:

Chart

Description automatically generated

Communication diagram:

Diagram

Description automatically generated

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

## Class Design

# *Graphical user interface, application Description automatically generated*Data Model

# *Graphical user interface Description automatically generated*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

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

# Future improvements

*[Present future improvements for the system]*

# Bibliography