**ONLINE BOOKSTORE**

**PROJECT DOCUMENTATION**

Student: Toma Lungoci

Group: 3043

1. Introduction

The goal of the project is to design and implement a functional web application, more specifically, an online bookstore. It should feature the functionality of common e-commerce website, allowing for easy and efficient user experience. The application will support interaction with several types of users(actors) along with their access permissions and specific user interface. The clients can browse the products, search a specific wished item, add them to a wish-list or to the shopping cart, after which they can proceed with the order by choosing the desired mean of payment and shipping. The data of the system will be stored and managed in a database. There are many entities which populate the database, such as products, orders, shippers, categories and so on. The interaction between the users and the application will be implemented with the help of a web server, through which the client sends the requests, answered by the server. Finally, the software system will be constructed in a layered architecture, to support the object-oriented programming principles.

1. Technology Stack

The web application runs on a web server and can be accessed through the help of a web browser. The system is modeled after the “Client-Server” structure, where the client sends requests to the server while the server answers with a response. This communication (HTTP communication protocol) makes possible the interaction of the user with the system. The development of the application will be assisted by some common architectural and structural patterns, presented in the diagrams that will follow. To implement the functionality of the application I will use a Java framework. The application will be implemented with the help of the SpringBoot Framework, which provides the core features used for building web applications on top of the Java platform. To ensure the communication between the web server and the client’s requests, I will use a SpringBoot backend application with REST, written in Java. For the frontend, I will use the Angular framework, which encapsulates TypeScript, CSS, HTML to provide support to design the UI of the application. For maintaining all the information required by the bookstore, I will use a database in connection with the application. The database will contain essential information about the clients, products, orders among many others. More specifically, the database will be on a MySQL, while the interaction with it will be provided by the SQL query language.

1. Software Architecture

The application will be structured following the Layered Architectural Pattern in which each layer communicates with the layer directly below or above it (hierarchical structure). The system’s architecture is based on four layers, more specifically: Controller, Service, Repository and Model. Along these, there is the actual persistent database which stores the information.

The Model layer consists of the entities of the application. It represents the one to one mapping of the entities in the database, transformed into Java objects. For example, we have: Customer, Order, OrderItem, Product, ProductCategory, Address.

The DAO layer, or the Repository, consists of instantiations of the CRUD repository provided by the SpringBoot framework. It provides the interface for accessing the records in the database and for performing different operations.

The Service layer is the one using the Repository layer to perform the operations on the database. It uses the CRUD methods of the repository with additional application logic to provide proper functionality of the system.

The Controller is responsible for delegating the actions to the specific service classes. It performs the mapping of the endpoints with their actions and then calls the appropriate service method, without performing any logic.

Diagram

Description automatically generated

1. Requirements

The application will provide most of the common functionality provided by some popular competitors such as Amazon Books, Elefant, Libris. The users can either choose to be a guest or create an account within the bookshop. They can browse the website to find the desired books, or if they have a clear-cut option, they can use the search/filter options to narrow down the list of options. They can manage a wish-list and a shopping cart. Finally, a client can order the products, choosing what delivery option and company they desire. They can see the status of their order, and upon delivery and some reading time, they van leave feedback and a rating on the website, for the service and for the books as well. To ensure that everything is running smoothly the administrator of the website can also modify some products and manage the users.

The clients with account are the main type of actors, they can perform most of the essential actions possible on the website. The guest clients only inherit a small part of the functionality, that is browsing all the available books. The administrator is the most powerful actor, having access to all the actions of a client and more, such as managing some clients and their accounts and updating products. The shippers have limited access as well, their interaction with the application resembles a small application for managing and tracking the orders.

Actors interacting with the system:

* Guest client
* Client with account
* Administrator
* Shipper

Additionally, the website should be easy to use, intuitive, it should be efficient as to quickly provide the results to the queries performed by the user, it should be secure, to maintain the personal data of the customer in an encrypted way.

1. Diagrams

**Use Case Diagram**

Diagram, schematic

Description automatically generatedDiagram

Description automatically generated

Diagram

Description automatically generated Diagram

Description automatically generated

**Database Diagram**

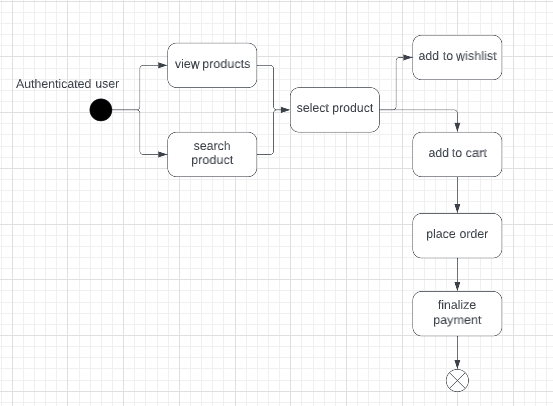
**Diagram

Description automatically generated**

**State Transition Diagram**

Diagram

Description automatically generatedDiagram

Description automatically generated****

**Sequence Diagram**

**Chart

Description automatically generated with low confidence**

**Class Diagram**

**Diagram

Description automatically generated**

**Package Diagram**

**Diagram, schematic

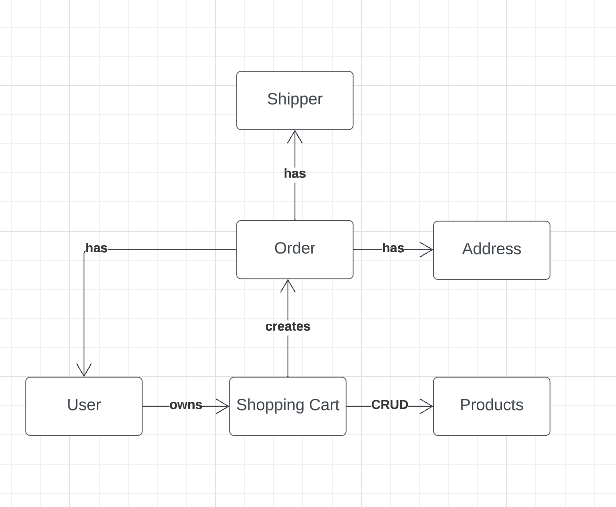
Description automatically generated**

**Deployment diagram**

**Diagram

Description automatically generated**

**Domain Model**

****

**Object Diagram**

**Diagram, schematic

Description automatically generated**

**Component Diagram**

**Diagram

Description automatically generated**