KaBook Store
Software Design Description

1. Revision History

Version	*Date*	*Author*	*Change Description*
1,0	15.04.2020	All members	Initial E-R constructor
1,1	10.05.2020	All members	Specific Pages
1.2	15.05.2020	All members	Authorization for Login
1.3	20.05.2020	All members	Avoid Redundancy (Redundant functionalities are removed)

2. INTRODUCTION

2.1 Purpose and Scope

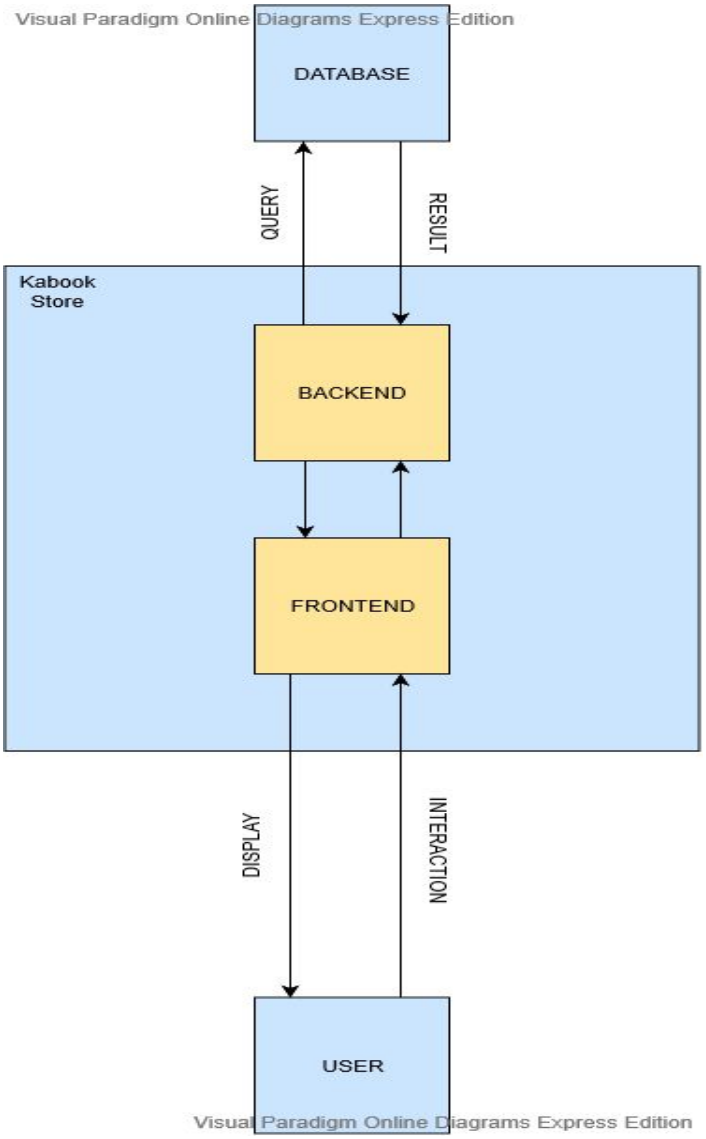
The purpose of this document is to give general overview about the design of the software. Details of the software are explained in this document such as how the software should work and how the interfaces should be. The topics covered in the document are; design constraints and decisions, design details including a class diagram, an E-R diagram and sequence diagrams of the use case and requirements traceability.

2.2 Document Overview

- Design Constraints and Decisions : Describe the general constraints implied by the design process and what are the impacts on the system architecture and modules design (time, tools, resources, etc.
- Design Details : In this section, the design model defined in the architectural design phase is elaborated.
- Requirements Traceability : Establish traceability relationship between requirements and software design to show that all requirements are met. Demonstrate relationship between classes and requirements using traceability matrix.
- Annexes : If some parts of the document need to be explained in more detail, create a separate document and annotate it in this section.

2.3 System Overview

Kabook Store is an web application that is stored on a remote server and delivered over the Internet. Basically, user interacts with the application and performs some functions through the Internet. This application most generally has two components. These are called as "Frontend" and "Backend". Frontend communicates with the backend after the user interacts with the frontend. Backend sends a query to database where the all data is stored and transmits the requested result to the frontend. Then, frontend presents the result to the user. System overview as a most general is drawn below:



2.4 Definitions, Acronyms, and Abbreviations

Term/Acronym	***Definition***
IEEE	The Institute of Electrical and Electronics Engineers
SRS	Software Requirement Specification
Java	Java is a general-purpose computer programming language that is concurrent, class based, object oriented, and specifically designed to have as few implementation dependencies as possible.

Hibernate	Hibernate ORM (or simply Hibernate) is an object-relational mapping tool for the Java programming language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate handles object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions.
HQL	Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) for writing SQL-like queries against Hibernate's data objects.
Angular	Angular is a TypeScript-based open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.
HTML	Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
CSS	Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML
VsCode	Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS.
Eclipse	Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment.
Bootstrap	Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

"Important Note: Definitions of some of the above mentioned abbreviations are taken from "wikipedia"."

2.5 References

Below you can see the references we used while developing our project:

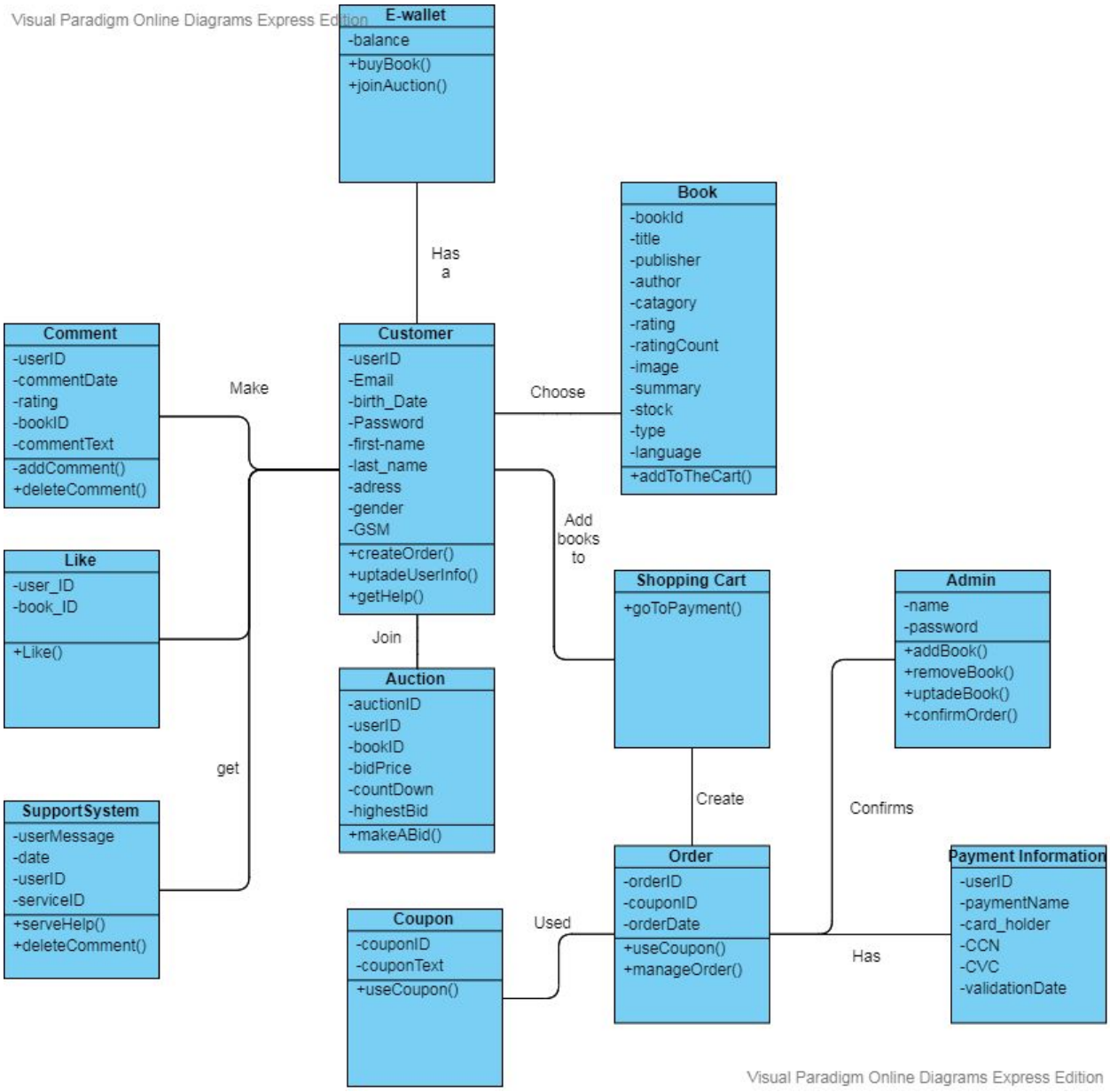
1. Angular Materials, https://material.angular.io/
2. Spring Boot, https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
3. Angular, https://angular.io/docs
4. 43+ Saatlik JAVA Kamp Kusu, Engin Demiroğ, https://www.udemy.com/share/101Q4RAksTcF9TRn4=/

3. Design Constraints and Decisions

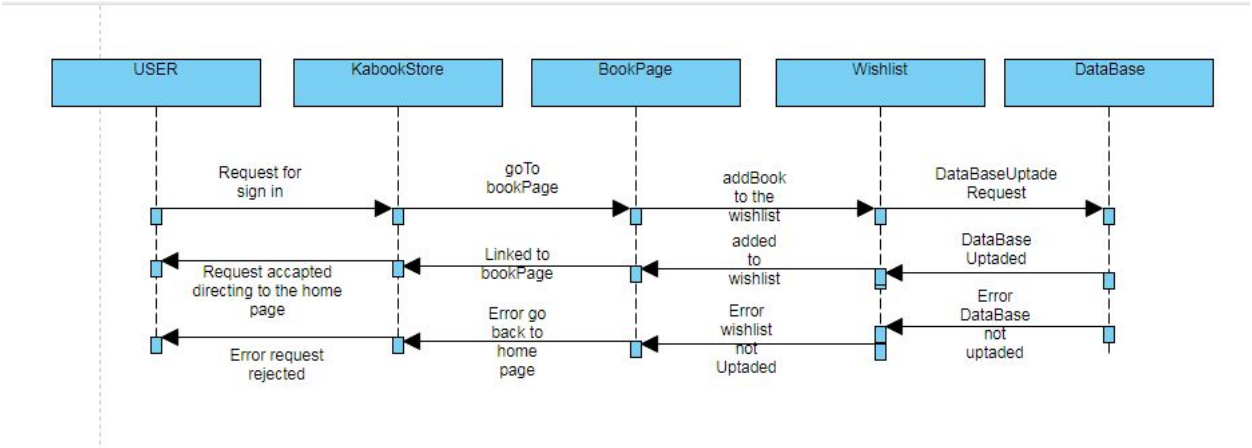
1. One of the biggest constraint was web browsers that developers use. As a result of the developers' use of different browsers, we could not take advantage of some basic html features. For this reason, we used the structure called Angular Materials that Angular Framework provided. In fact, this constraint has enabled us to avoid possible problems that our system will encounter in the future.
2. We did not intend to use a common database as a database. However, we decided to use a common database due to import and export problems while using MySQL workbench.
3. We decided on the development environment to be Linux. However, due to limited hardware resources in developers' environment, we decided to develop on Windows OSs that are already available
4. In the demo version of the application, we wanted CRUD operations to be done through popup pages. But the problem we had about the versions of Angular forced us to do these things on separate pages.

4. Design Details

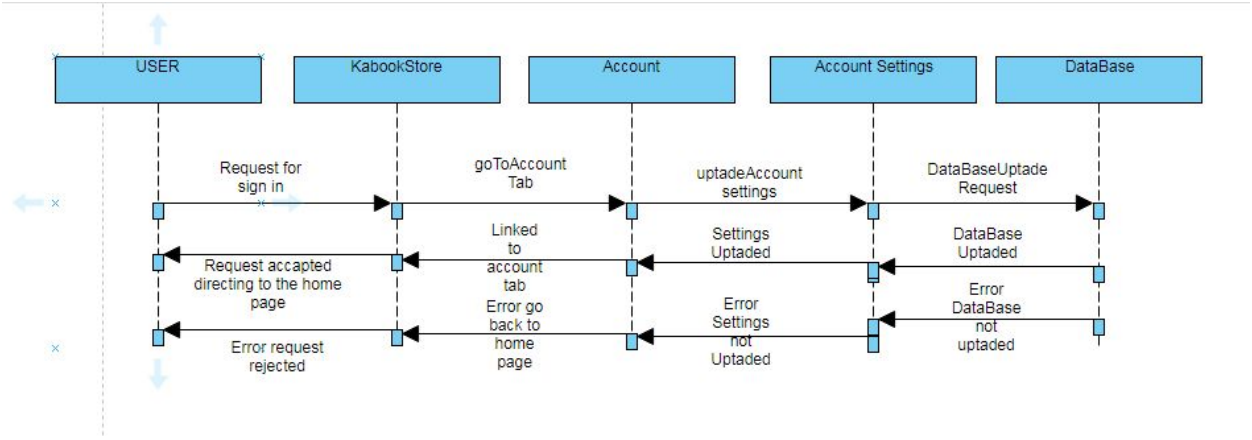
4.1 Software Components



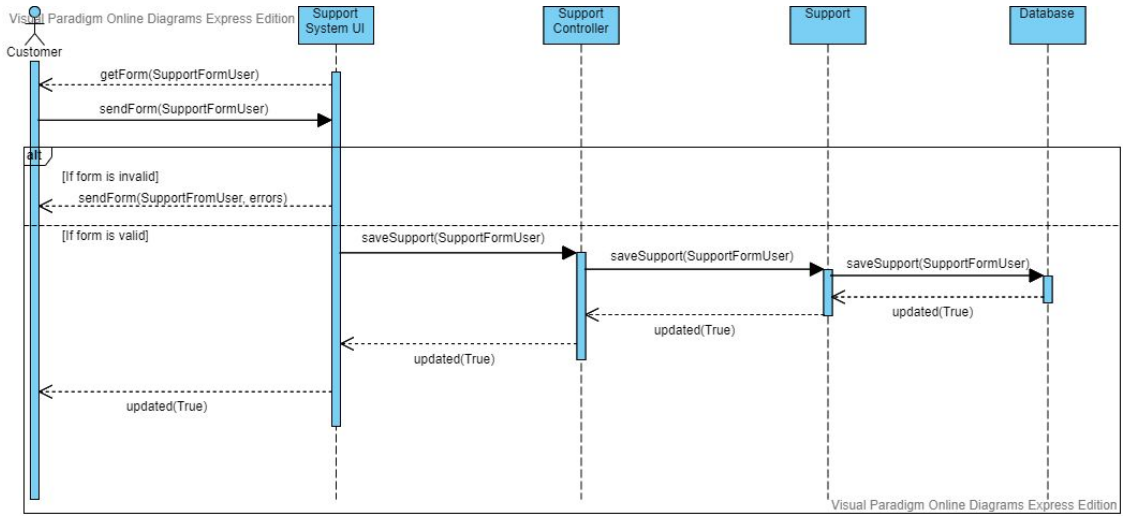
4.2 Software Behavior



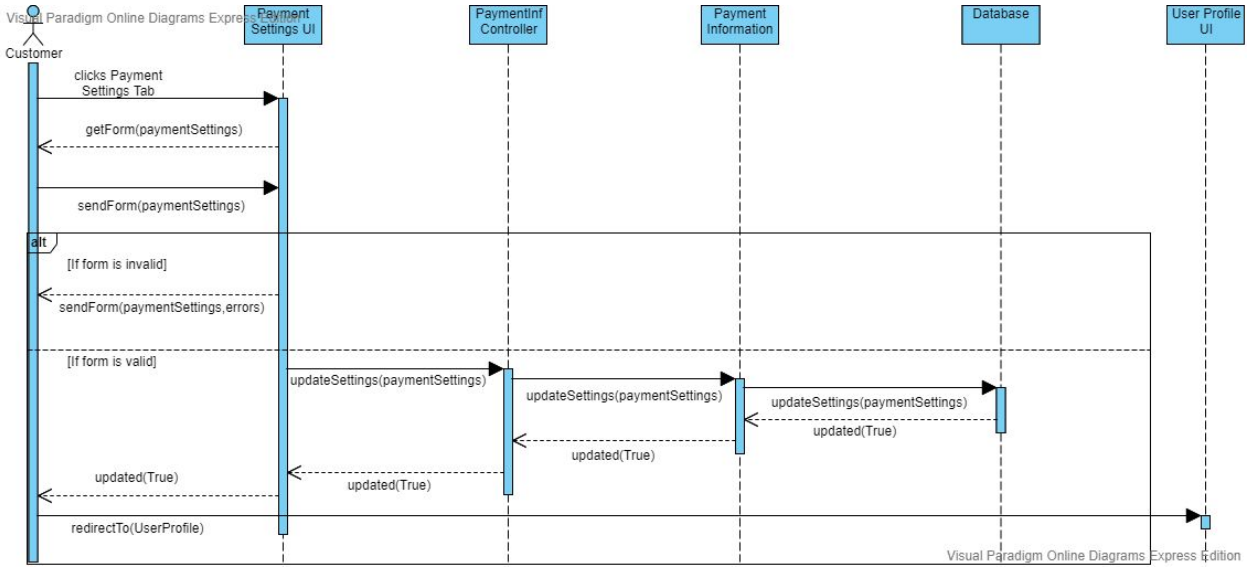
Wishlist Sequence Diagram



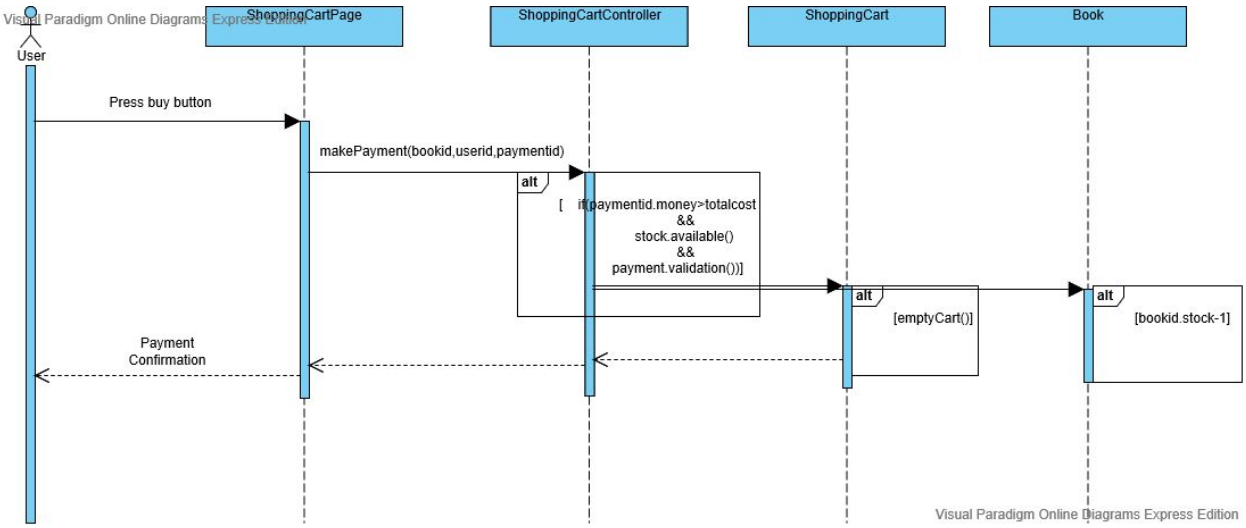
Profile Settings Sequence Diagram



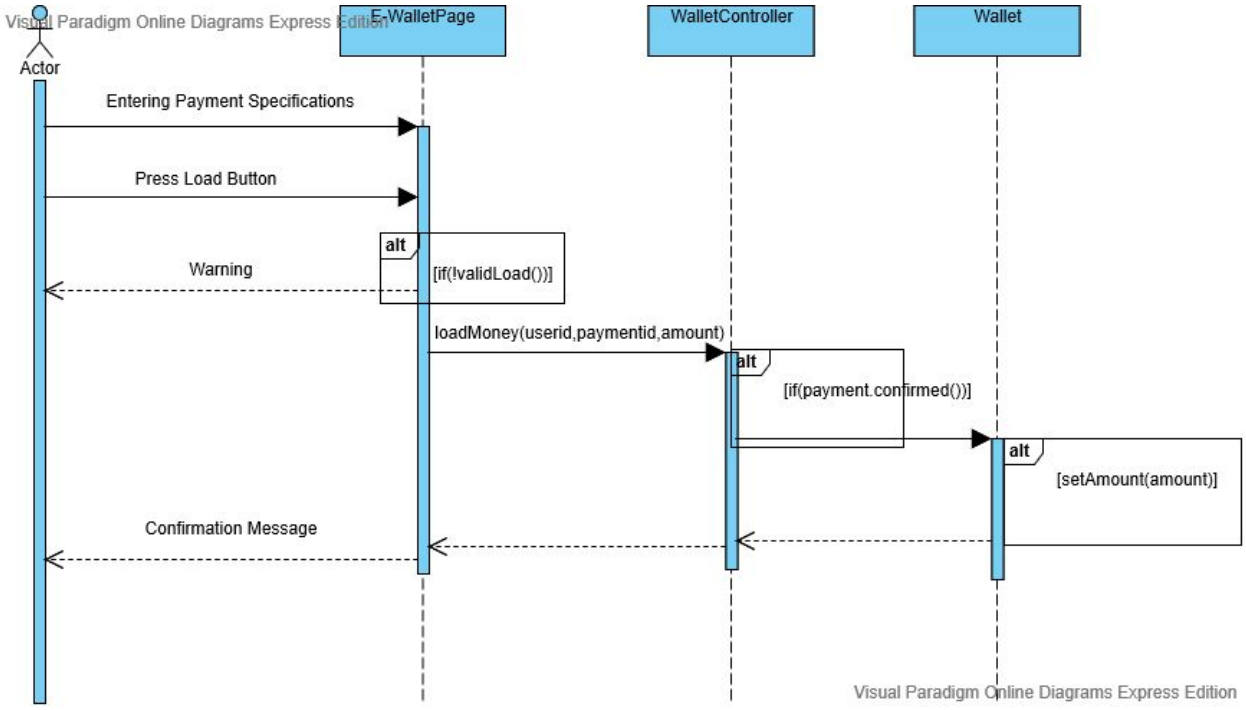
Support Service Sequence Diagram



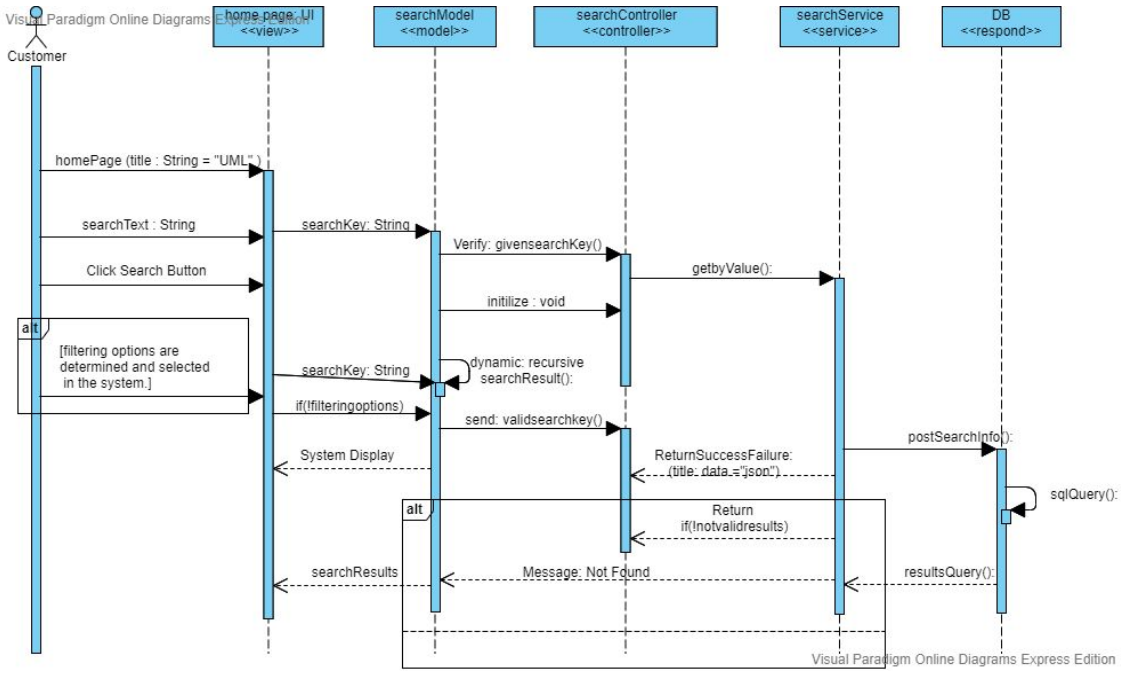
Payment Settings Sequence Diagram



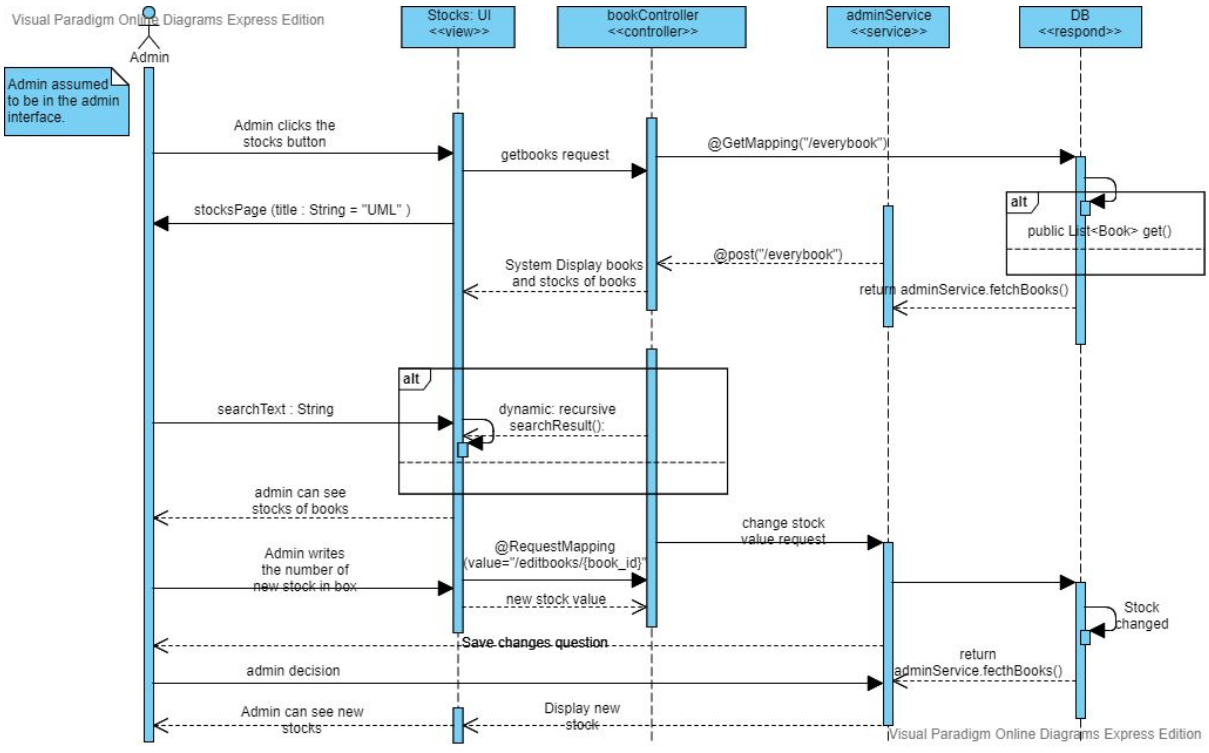
Shopping Cart Sequence Diagram



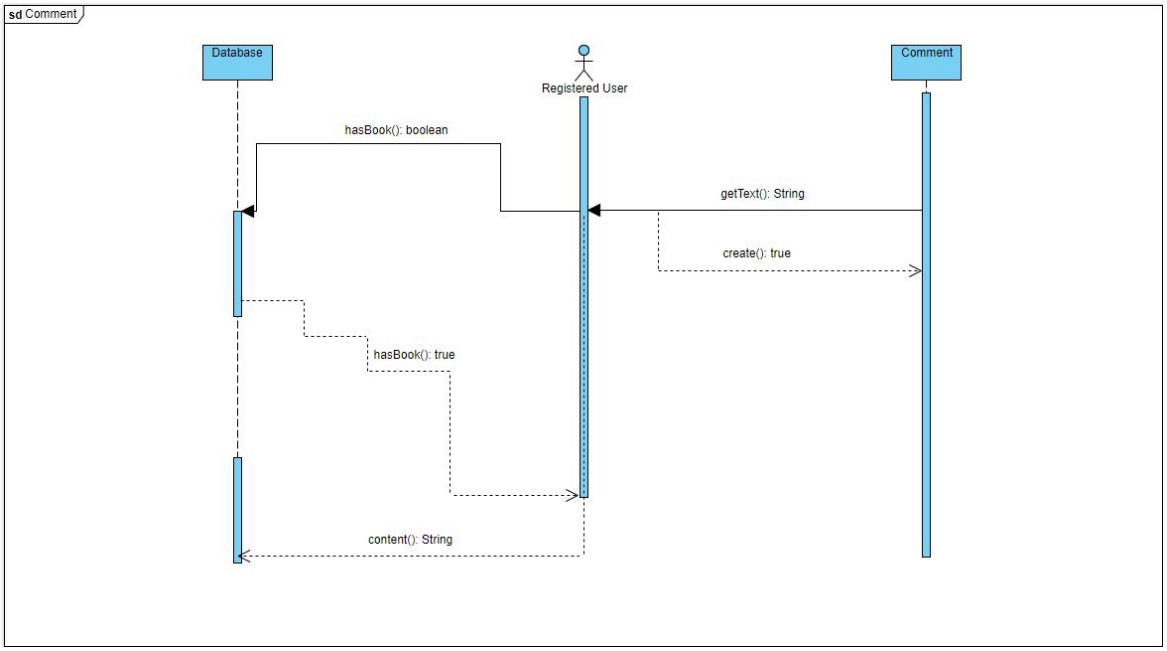
Payment Sequence Diagram



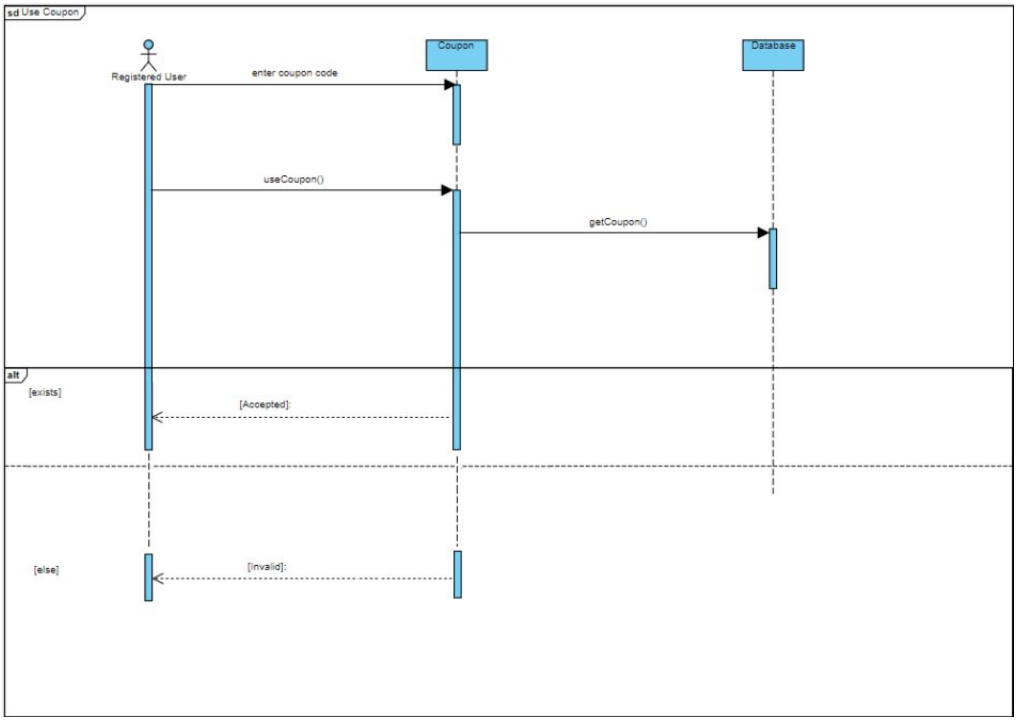
Search and Filter Sequence Diagram



Stock Control Sequence Diagram

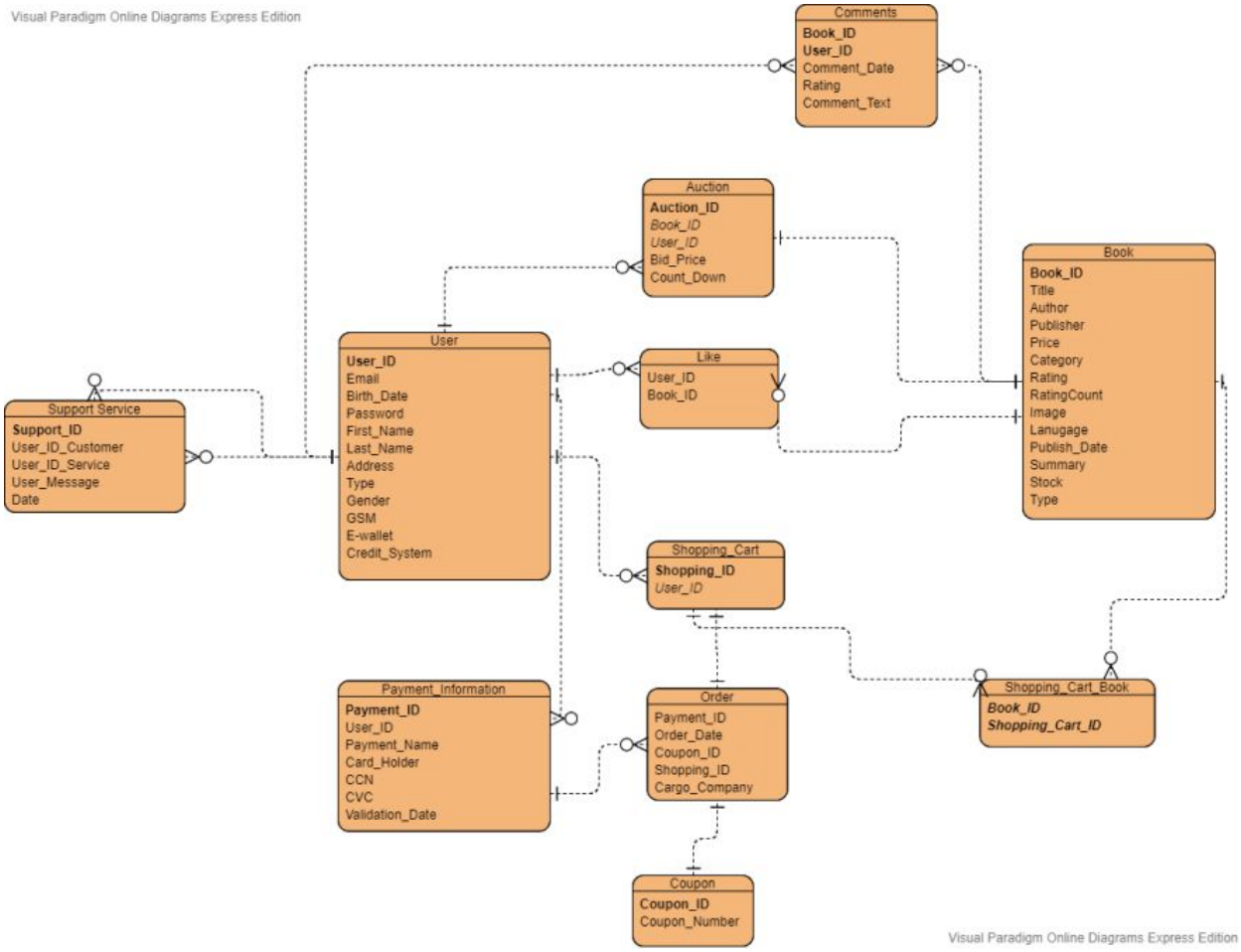


Comments Sequence Diagram



Coupon Sequence Diagram

4.3 Data Model (E-R Diagram)



4.4 User Interface Design

4.4.1 – Changes in UI

While designing the user interface, we tried to adhere to the simple interfaces we designed in the old times of the project, but unexpected situations occurred. Angular has been very helpful to us at some points, but we have not been able to take various steps in terms of package use and library use. While aiming to use POP-UP pages quite a lot, we found it more convenient and useful to use the user less than we expected and to direct the user to send different pages. One of the things that helped us the most when developing the project was that we used angular materials.

4.4.2 – Explanation of GUI (Graphical User Interfaces)

We are developing the user interface with AngularJS and we only have the main page in the interface, and we use the routing module in the middle of the page. We have a navigation bar at the top of our page and a sidebar on the left. Since Angular consists of components, all contents that change from the page are composed of components, and these components are accessed from the navigation bar and sidebar.

4.4.3 – Error prevention

Handling errors properly is essential in building a robust application in Angular. Error handlers provide an opportunity to present friendly information to the user and collect important data for development.In this article, we will compare several solutions for error handling in Angular apps. First, we will describe the traditional approaches using ErrorHandler and HttpClient. Then, we will show you a better solution using HttpInterceptor. We'll also show you how to use this interceptor to monitor and track errors centrally in Pop-UP pages.

4.4.4 – Design plans and types

We try not to design differently from the user interface we designed before. But smaller changes are taking place. User interface of our application will be easy to use and understandable. It uses English language since our application is designed for Course Project.

4.4.5 – User control, freedom and accessibility

We tried to keep the user interface as understandable and simple as we can. While unregistered users can only view the books, registered users can access, purchase, comment, rate and edit their profile after logging into the system. We created a separate interface for support and admins, and only authorized users can access it.

5. Requirements Traceability

REQUIREMENTS

1. Admin should be able to add books to the system.
2. The system shall provide password protection for admin account and customer accounts.
3. Customer shall add the desired book into the shopping cart.
4. System shall give a warning in any case that causes problem.
5. The system shall allow someone to enter review to books and rate the books.
6. The system shall provide different types of search methods such as searching by title, keyword, ISBN number, author etc.
7. User agreement must be provided in the system.
8. The system should provide unique account authorization for each user type(admin,customer,shipping company).
9. Customer shall add any book to wishlist.
10. System should provide a secure payment.
11. Customer should be able to use coupons in the payment phase for discount.
12. Users should be informed about discount/promotions.
13. Users can update their account informations via account panel.
14. System shall be a flexible for debugging and system update.
15. Users should be able to inform the system about their complaints/ requests to develop the system furthermore.

CLASSES IN CLASS DIAGRAM

1. SupportService
2. SupportController
3. User
4. UserUpdate
5. Login
6. Payment

7. Register
8. UserProfile
9. UserController
10. ShoppingCart
11. OrderController
12. AppController
13. KabookStoreApp
14. BookController
15. Review
16. Book

REQ CLASS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1															X
2															X
3	X	X	X	X				X	X		X	X	X		X
4		X		X									X		
5		X		X				X							
6			X	X						X	X	X			
7		X		X			X								
8			X	X	X	X			X			X	X		X
9	X	X	X	X	X	X		X	X		X	X	X		X
10	X		X			X					X				
11			X	X	X	X				X	X	X			
12	X	X	X	X	X	X	X	X	X				X		
13	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
14	X		X	X	X	X			X						
15				X	X	X									
16	X		X	X	X	X			X						