Boom Book Store
Architecture Notebook

1. Purpose

This document describes the philosophy, decisions, constraints, justifications, significant elements, and any other overarching aspects of the system that shape the design and implementation. It reflects the important architectural decisions that made about project. This document contains information about different kind of views and abstractions with related diagrams.

2. Architectural goals and philosophy

Despite the fact that this system will not be developed to subrogate a system or adapt to a legacy system , it provides extensibility and reusability on modules to do so if demanded. Another important note about architecture philosophy is Open/Closed Principle (which is a SOLID design principle) implemented in the project. To provide further information about architecture, architectural goals are explained below.

- Since it is designed as an easy to use web-based application , there should be no hardware dependency.
- The only need to use the system from user's perpective is to use a modern browser.
- Data persistence will be provided via addressing a relational database which is being managed by MySQL and it should respond the clients any moment while keeping data integrity and consistency.
- The architecture is testable (which allows to check each element individually and integrated) and stable. (which provides endurance to possible changes or extensions in future).
- The system's response time to user during functional requirements (login, register , add a book to cart etc.) should not be more than few seconds. But the actual performance of the system can be measured after the deployment.
- The system provides enough authentication and authorization which means only admin should be able to use all features of the system.
- The sofware design should have high cohesion and low coupling between modules.
- The architecture should be easy to understand and implement.

As a summary, the architecture should provide **Design Qualities** which are conceptional integrity , maintainability, and **Run-time Qualities** which are reliability , scalability and interoperability , performance and security and **Non-run-time qualities** that portability , modifiability.

3. Assumptions and dependencies

In order to utilize the system functionalities , we assume that:

- Even if the system is not dependent to any hardware interface , the users have an operating system.
- Users have an environment with Internet connection.
- Users have a browser which is able to send HTTP / HTTPS request.

- The database is capable to process and store the data that will be send by users.
- The database has quick transaction processing.
- The Tomcat Server embedded in Spring Boot runs smoothly

4. Architecturally significant requirements

As our consideration on requirements, the architecturally significant requirements are determined as :

- The system's response time to the user shouldn't be more than few seconds.
- The system should encrypt the user passwords in case external database attacks.
- The system should work on today's browsers especially on Google Chrome
- The system should use secure HTTP or HTTPS requests because of transactions
- The system must keep track of users' orders and delivery situations.
- The system should validate the credit card of the user during payment process

5. Decisions, constraints, and justifications

DECISION/CONSTRAIN	JUSTIFICATION
Using a layered architecture with important assets in inner layers (for example :database connection)	It increases the security of the application
Using MVC design pattern	To provide high cohesion and low coupling
Using replaceable components with less possibly dependency	To ensure maintainability of the project
Reside important operations as most local as possible with minimum communication	To increase the performance
Having more than one alternative user interface components	To provide availability of the project in case some components' failure

6. Architectural Mechanisms

We use three stage architectural mechanism for common solutions to common problems that can be used during development to minimize complexity.it is an important method especially for non-functional requirements

1.Analysis Mechanism: The analysis mechanism is one of the initial stages of our project. We have identified it in the first stages and it is designed to understand the requirements well and to have team members master the process. It is important for determining the requirements and processing them effectively and reliably.

2.Design Mechanism: The design mechanism is the stage that we decided together with the reasons for the technologies to be used to improve the architecture. For example, since we want to keep the data permanently, we use RDBMS(mySQL).

3.Implementation Mechanism: In this mechanism, we actually started to implement the architecture, we designed the interface using the html and bootstrap for the view part, and wrote code in the spring for the controller part to realize the demo**.**

7. Key abstractions

KEY ABSTRACTION	DESCRIPTION
Customer	Its is the user of the website who has the functionality of registering, logging in , buying book etc.
Book	It’s the abstraction of the object who will be being sold
Payment_Service	It’s the third party library which will be ensure payment
Shopping_Basket	It’s the abstraction of gathering products in real world
Courier_Company	Its’s the abstraction of real world cargo companies which will deliver the books to the customers
Admin	The manager abstraction of the system who has extensive control over the system.

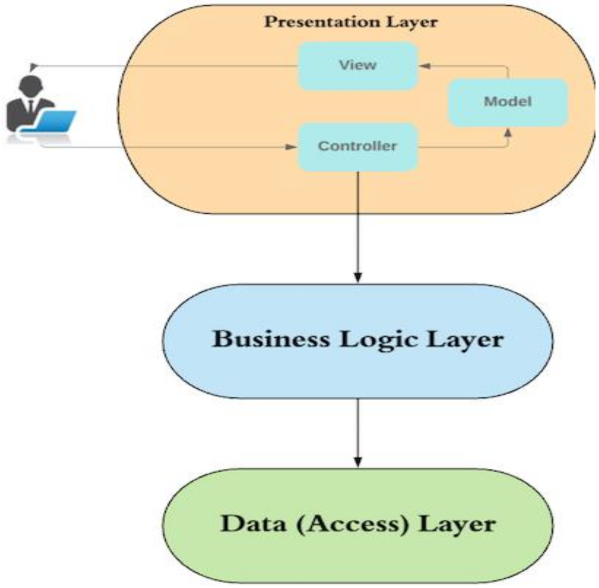
8. Layers or architectural framework

We used 3-layer architecture so that the system was expected to be more reliable, fast and easy to manage. There are three distinct layers in this architectur: Data Layer, Business Logic Layer, Presentation Layer

1.Presentation Layer: This is the user interface of the application that presents the application’s features and data to the user.

2.Business Logic Layer: This layer contains the business logic that drives the application’s core functionalities. Like making decisions, calculations, evaluations, and processing the data passing between the other two layers.

3.Data Layer: This layer is responsible for interacting with databases to save and restore application data.

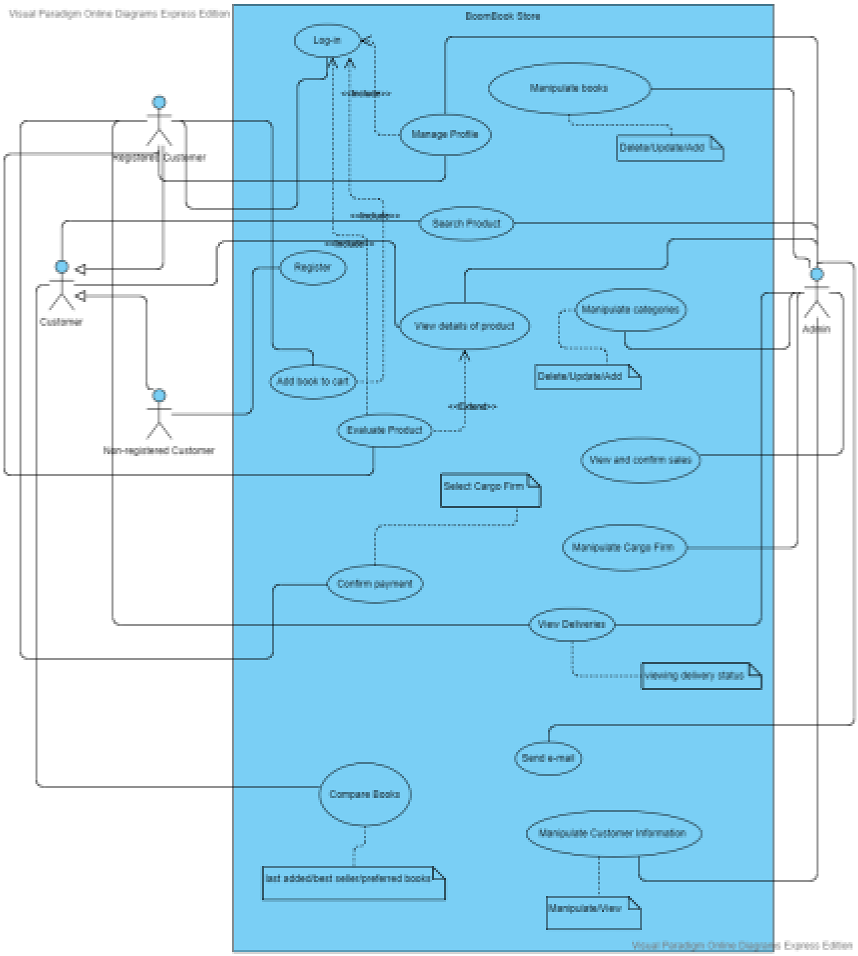


9. Architectural views

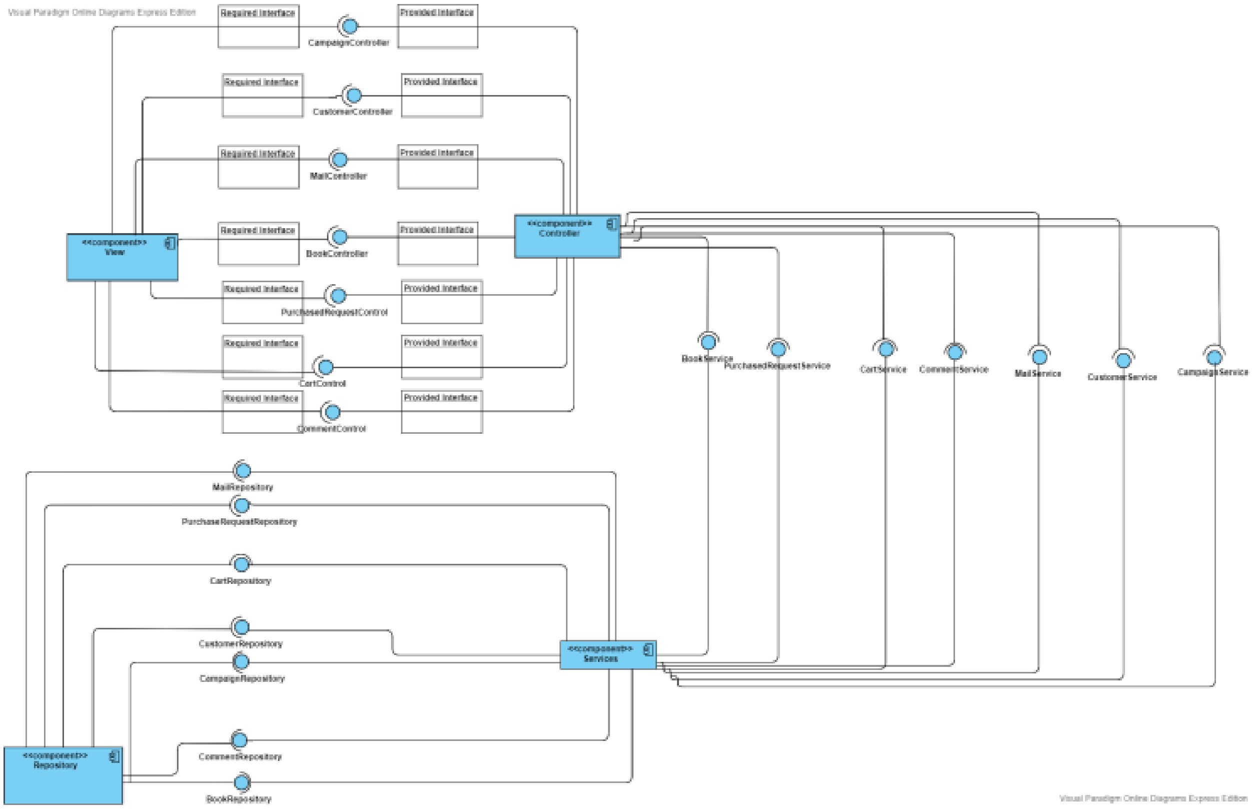
Recommended views

- **Logical View:** Since the size of the class diagram is too high, we sent it as an attachment outside the report.

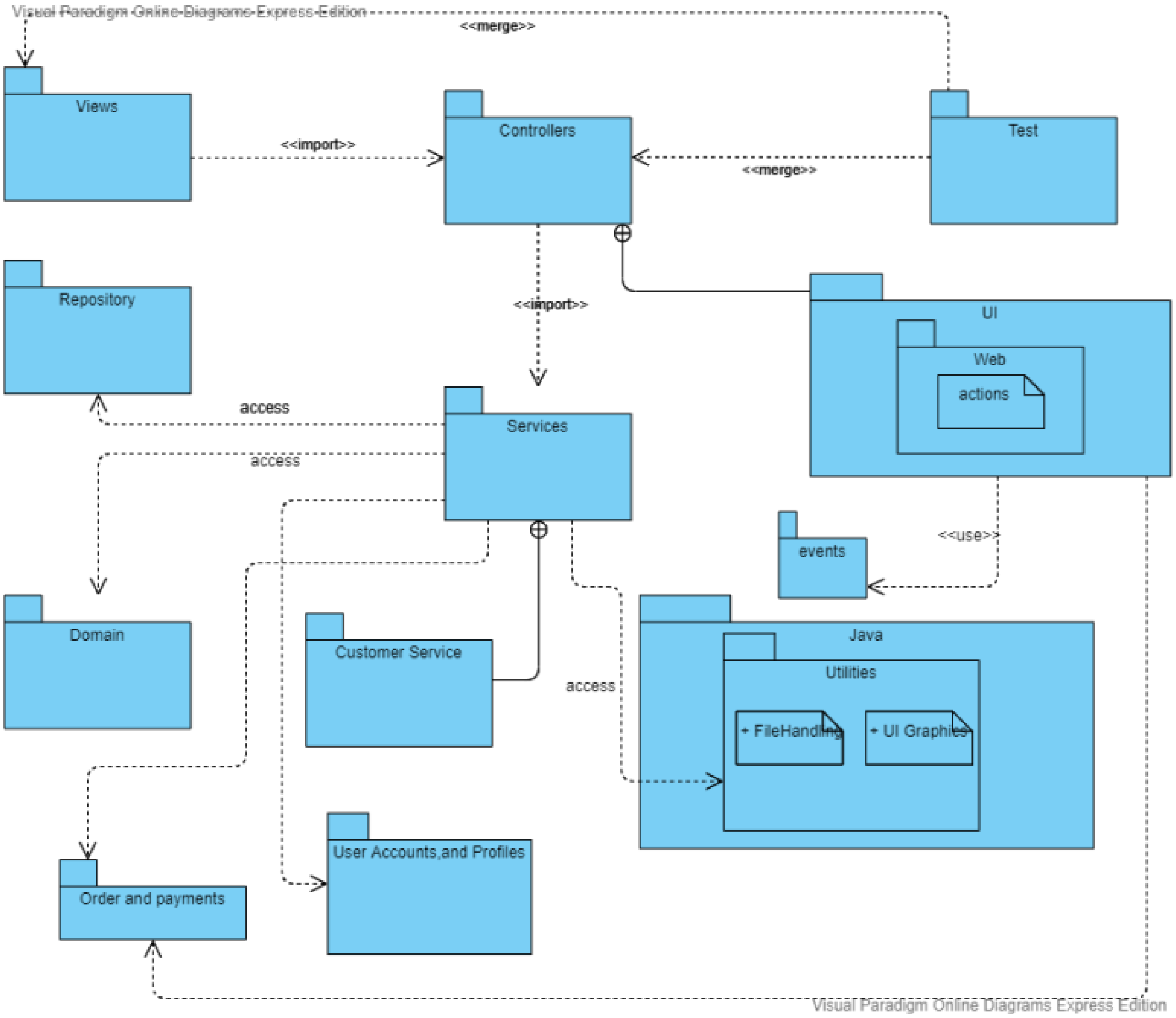
- **Use case:**



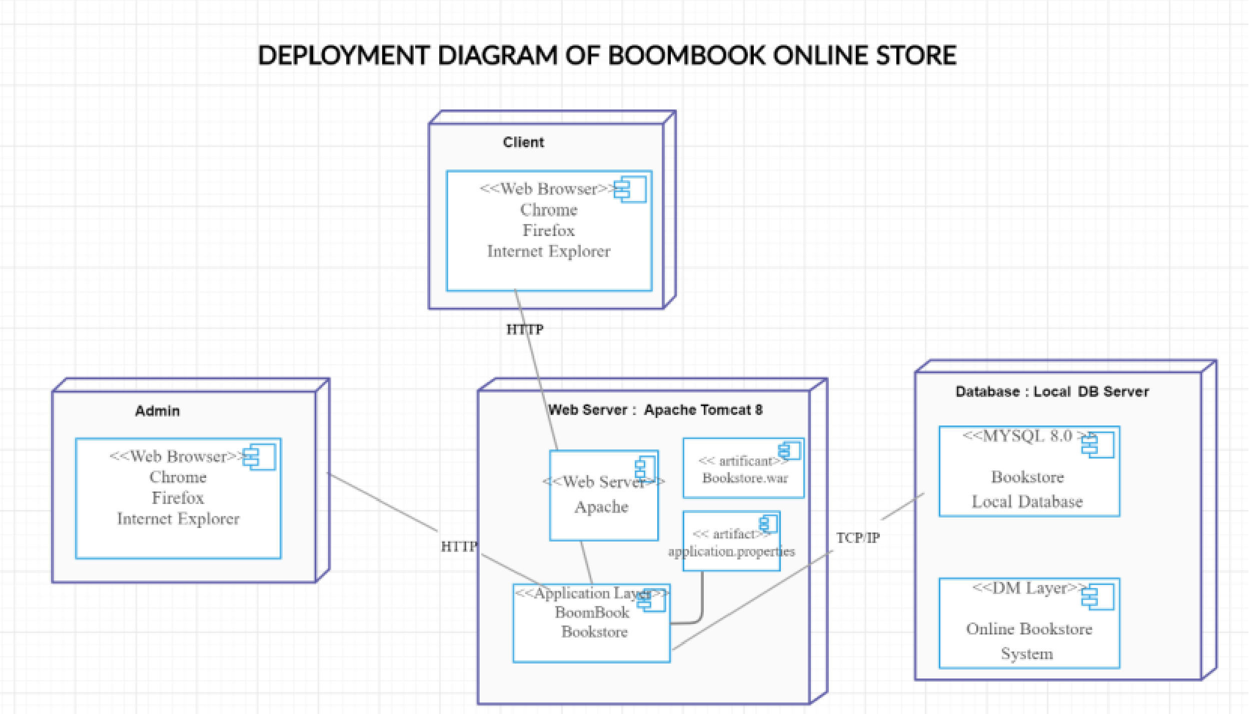
- **Implementation View:** Component Diagram


Visual Paradigm Online Diagrams Express Edition
Required Interface
CampaignController
Provided Interface
Required Interface
CustomerController
Provided Interface
Required Interface
MailController
Provided Interface
<<component>>
View
Required Interface
BookController
Provided Interface
<<component>>
Controller
Required Interface
PurchasedRequestControl
Provided Interface
Required Interface
CartControl
Provided Interface
Required Interface
CommentControl
Provided Interface
BookService
PurchasedRequestService
CartService
CommentService
MailService
CustomerService
CampaignService
MailRepository
PurchaseRequestRepository
CartRepository
CustomerRepository
CampaignRepository
<<component>>
Services
<<component>>
Repository
CommentRepository
BookRepository
Visual Paradigm Online Diagrams Express Edition

packacge diagram:



- **Physical View**
  Deployment Diagram


DEPLOYMENT DIAGRAM OF BOOMBOOK ONLINE STORE
Client
<<Web Browser>>
Chrome
Firefox
Internet Explorer
HTTP
Admin
<<Web Browser>>
Chrome
Firefox
Internet Explorer
HTTP
Web Server : Apache Tomcat 8
<<Web Server>>
Apache
<< artificant>>
Bookstore.war
<< artifact>>
application.properties
<<Application Layer>>
BoomBook
Bookstore
TCP/IP
Database : Local DB Server
<<MYSQL 8.0 >>
Bookstore
Local Database
<<DM Layer>>
Online Bookstore
System