**Shopping Cart System**

**Low Level Design (LLD)**

Date: 13/07/2022



Current Document Version: [*1.0*]

DOCUMENT APPROVAL

**Approvers of this document**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Department** | **Role** | **Signature** | **Date** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Document Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Version #** | **Author** | **Date** | **Description** |
| 1.0 | Dnyanada Dhore | 13/07/2022 | Shopping Cart System |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.0 Document Purpose](#_Toc94636300)

[2.0 Intended Audience](#_Toc94636301)

[3.0 Project Background, Objective(s)](#_Toc94636302)

[3.1 Project Background](#_Toc94636303)

[3.2 Project Objective](#_Toc94636304)

[4.0 Design Pattern](#_Toc94636305)

[5.0 Requirements](#_Toc94636305)

[6.0 Solution Diagram](#_Toc94636306)

[7.0 Solution Steps](#_Toc94636307)

[8.0 Classes/function name](#_Toc94636308)

[9.0 Data Tables](#_Toc94636310)

[10.0 System Diagram](#_Toc94636311)

[11.0 ER Diagram](#_Toc94636311)

[12.0 Http Status Code](#_Toc94636311)

[13.0 Unit Testing](#_Toc94636311)

[14.0 DFD Diagram](#_Toc94636311)

[15.0 Use Case Diagram](#_Toc94636311)

# 

# Document Purpose

This document describes the solution architecture for Shopping Cart System

# Intended Audience

This document is intended as a reference for the following roles and stakeholders who are interested in the Shopping Cart system technical architecture.

|  |  |
| --- | --- |
| Role | Nature of Engagement in WB Classics Portal Technical Architecture |
| Product Owners/SME | Key stakeholder to ensure that the architecture is aligned with business goals. |
| Business Analysts | Business analysts are one of the stakeholders who are informed with the key architectural decisions. |
| Enterprise Architects | To enforce Shopping Cart System Platform Architecture is aligned to business goals and architecture, architectural guidelines. |
| Solution Architects | To ensure solution design and architecture is aligned to business requirements, architectural guidelines. |
| Developers | Use Technical Architecture Document as the guiding document for detail design and implantation approach to align with Shopping Cart System. |
| End-User | User can see the products, can order the products. |

# Project Background, Objective(s)

## Project Background

This Shopping Cart System is online store where different products will be available. From where user can buy the product and they get the home delivery of the products.

## Project Objective

In this system user can register themselves with there personal information , they can buy the products .After buying they can pay the amount by online mode or cash on deliver also available. Once the order is placed user gets there products within 3 to 4 days. And the admin can delete add and update the product information.

# Design Pattern

|  |  |  |
| --- | --- | --- |
| # | Name | Description |
| 1  2  3 | Angular  Web Api  Database | To create and design the front end  Using HTTP requests, we will use the respective action to trigger various operations  To store and retrieve the information |

# Requirements

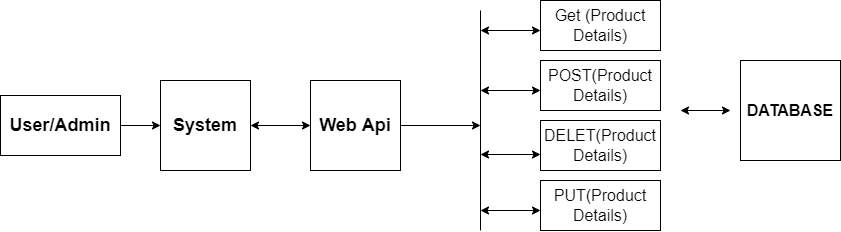
**Hardware Requirements:-** 64 GB Windows System

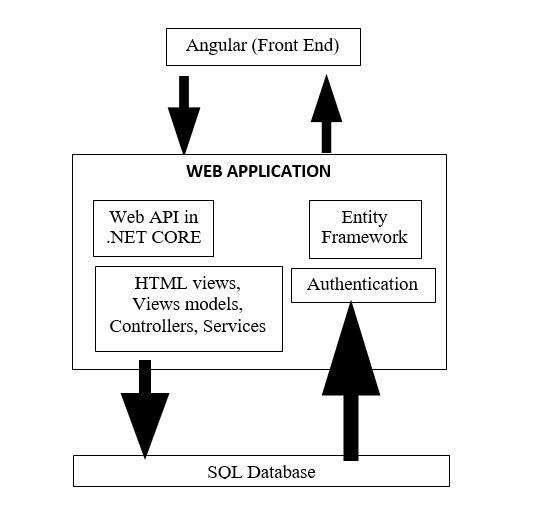
**Software Requirements:-** 2019 Visual Studio , Microsoft SQL Server Management Studio,

Visual Studio Code

**Specification:-**HTML, CSS, Bootstrap, JavaScript, TypeScript, Angular ASP.NET, jQuery , MySQL

# Solution Diagram





# 7.0 Solution Steps

**User Registration and login**

1. User will enter the required details such as firstname, lastname, email, phone number, password, address and click register button, browser directs the request to user registration API
2. call reaches the API gateway
3. API gateway does the routing and forwards the request. So, the user registration gets successful.
4. Once the user registered him. Later the user can login to the system.
5. Login and Signup are necessary to place the order.
6. For the registration there will be validation. So, if user want to register then validation have to be successful.
7. Without registration user can add the product to the cart but order will be placed only if user login to the system.

**Admin Registration and login**

1. Admin will enter the required details as like user.
2. This data will be stored in the database.
3. After admin logins to the system admin will be redirected to the page where user can add, delete, update the product

**Product Addition, Deletion and Update**

1. Admin can add the product by post service.
2. Product list can be available to both user and admin by get service.
3. Deletion of method is possible by delete service.
4. If admin wants to edit the product details, then it is possible through put service.
5. The product will be added to the cart as soon as user clicks on the add to cart button. User can not add, delete or edit the product details.

**Cart Details**

1. The cart will display the total price of the product those were added by user
2. User will be able to remove item from the cart

**Payment Details**

I this section, the user needs to add the proper address with the fields like area, postal code and the payment details such as if the want online payment then there should be card details , if the payment is cash on delivery then the have to select that option.

# 8.0 Classes and Functions

Following are the important Concepts of the project

* User
* Product
* Signup
* Login
* Order
* Cart
* Transaction

In the .Net Core,

In the **Model** folder there will be class files as per the above modules. In these files we will declare all the variables of respective modules.

In the **Repository** folder there will be interfaces and classes as above which we will be useful for performing CRUD operations.

In the **Services** folder again, there will be classes as per the above modules which will implement all the required http services.

The **Controller** handles and responds to user input and interac**tion**. For example, the controller handles URL segments and query-string values, and passes these values to the model.

|  |  |  |
| --- | --- | --- |
| # | Class | Description |
| 1 | User.cs , Product.cs, Cart.cs, Order.cs, Transaction.cs | Model for holding the data schema for user, product, cart, order, transaction respectively. |
| 2 | Iuser.cs, Iproduct.cs, Icart.cs, Iorder.cs, Itransaction.cs | The interface in Data Access Layer for user, product, cart, order, transaction respectively. |
| 3 | UserRepo.cs, ProductRepo.cs, CartRepo.cs, OrderRepo.cs, TransactionRepo.cs | Implementation of interface for user, product, cart, order, transaction respectively. |
| 4 | UserCont.cs, ProductCont.cs, CartCont.cs, OrderCont.cs, TransactionCont.cs | Controller handles the incoming HTTP requests and send the response back to the caller. |
| 5 | UserSer.cs, ProductSer.cs, CartSer.cs, OrderSer.cs, TransactionSer.cs | It is Business Access Layer holding the Business Logic and mediates the communication between the controller and repository. |

# 9.0 Data model/Table

User Details Table

|  |  |  |
| --- | --- | --- |
| PK | UserID | INT |
|  | FirstName | VARCHAR(25) |
|  | LastName | VARCHAR(25) |
|  | Email | VARCHAR(255) |
|  | Phone | VARCHAR(10) |
|  | Password | VARCHAR (255) |

Admin Table

|  |  |  |
| --- | --- | --- |
|  | email | VARCHAR(255) |
|  | password | VARCHAR (255) |

Product Table

|  |  |  |
| --- | --- | --- |
| PK | ProductId | INT |
|  | ProductCategory | VARCHAR(25) |
|  | ProductName | VARCHAR(25) |
|  | Price | VARCHAR(25) |
|  | Description | VARCHAR(25) |
|  | Image | VARCHAR(255) |

Cart Table

|  |  |  |
| --- | --- | --- |
| PK | CartId | INT |
| FK | UserID | INT |
| FK | ProductId | INT |
| FK | ProductName | VARCHAR(25) |
|  | Quantity | INT |
|  | TPrice | VARCHAR(255) |

Order Table

|  |  |  |
| --- | --- | --- |
| PK | OrderId | INT |
| FK | CartId | INT |
| FK | UserId | INT |
| FK | ProductId | INT |
| FK | ProductName | VARCHAR(25) |
|  | Quantity | INT |
| FK | TPrice | VARCHAR(255) |

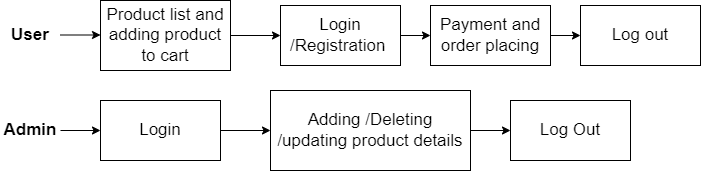
Transaction Table

|  |  |  |
| --- | --- | --- |
| PK | TransactionId | INT |
| FK | Tprice | VARCHAR(25) |
|  | Transaction\_Status | VARCHAR(25) |
| FK | OrderId | INT |
| FK | UserId | INT |
|  | TransactionType | VARCHAR(25) |

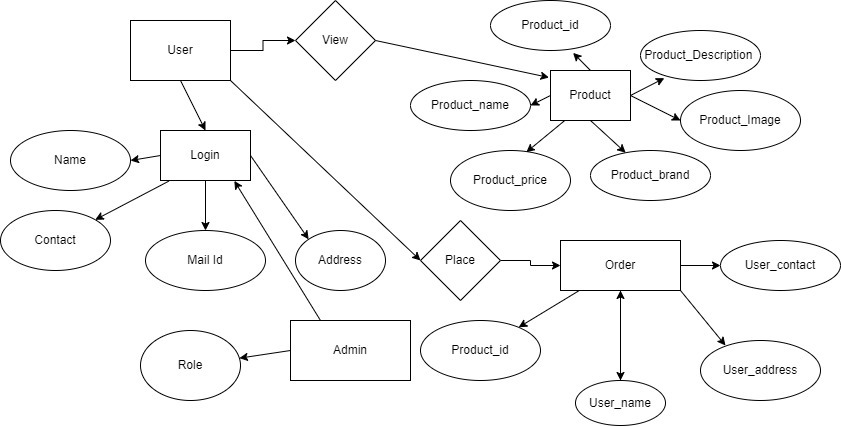
Address Table

|  |  |  |
| --- | --- | --- |
| PK | AddressId | INT |
| FK | UserID | INT |
|  | Address | VARCHAR(100) |
|  | City | VARCHAR(25) |
|  | State | VARCHAR(25) |
|  | Pincode | VARCHAR(6) |

# 10.0 System Diagram



# 11.0 ER Diagram



# 12.0 Http Status Code

201-user Register

200-Request succeeded

400-Inputs are invalid

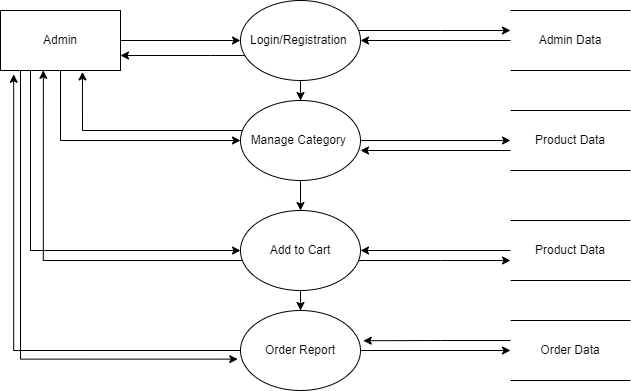
404-user not found

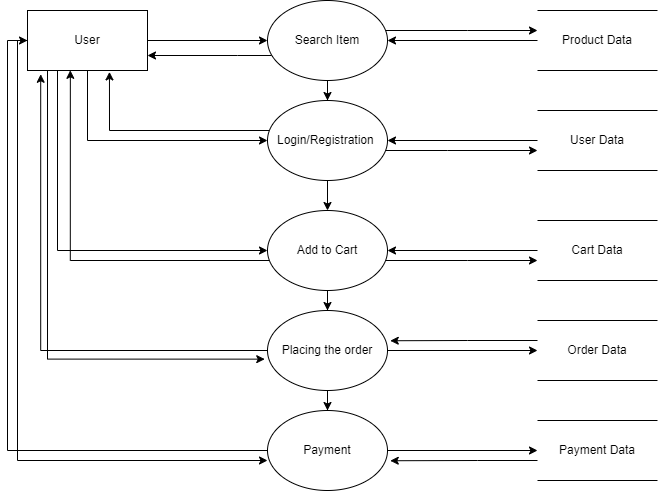
502-Bad gateway

# 13.0 Unit Testing

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case Id | TEST CASE SCENARIO | TEST CASE | PRE CONDITION | TEST STEPS | TEST DATA | EXCEPTED RESULT | ACTUAL RESULT |
| TC 1 | Registration | Enter Valid data to get register | User needs to enter all the valid details | Enter First name, last name, email and every necessary information | Valid details | Successful registration | Successful registration |
| TC 2 | Adding Data | Enter Valid data to add the product | Admin needs to enter all the valid product details | Enter product name, Image, Price and every necessary information | Valid Details | Successfully added | Successfully added |
| TC 3 | Updating Data | Enter Valid data to Update the product | Admin needs to enter all the valid product details | Enter product name, Image, Price and every necessary information to update | Valid Details | Successfully  updated | Successfully Updated |
| Tc 4 | Placing Order | Enter valid payment details | User needs to enter all the valid details | Enter address And payment details | Valid Details | Successfully Placed | Successfully Placed |

# 14.0 DFD Diagram





# 15.0 Use Case Diagram

