REPORT GENERATION SYSTEM - eCommerce Technical Architecture and Solution

Author: Srimanikandan Ravikumar

Version: 0.1

Date: 10-10-2023

Table of Contents

1. Introduction

Report Generation System - eCommerce is a project that deals with the reports of the Online Shopping company **SportsZone** which is providing good quality products like accessories, equipment, clothing, nutrition etc,. related to sports.

The main motto of the project is to show the reports with data that are generated via **SportsZone Web application**.

This project includes two use cases namely the customer and the admin. Customer can browse, view, buy products which are all available in the inventory. Customer will be able to buy products via cart option. Admin can update the order status and customer details. Admin can view the various reports i.e Customer reports, Product reports, Order reports.

Reports are generated with real time data which will be stored on to database by integrating with the customer via web application.

The purpose of this document is to add the necessary detail to the current business requirement to represent a suitable architecture and design for coding. This document is also intended to provide a high-level design prior to coding and can be used as a reference manual for how the modules interact at a high level.

1.1 Architecture Components Overview

This section of the document describes the technical components in **Report Generation System - eCommerce**. The **SportsZone web application** presents all the design aspects of the application architecture and the details of the business components. Apart from the proposed solution, this document also includes deployment options and the non-functional requirements identified for this project. The architecture components considered have been listed as below.

No	Component Name	Description
1	Application Architecture	This section explains the basic layout of the architectural design proposed for this application.
2	Technical Stack Layout	This section explains the technology stack proposed for the various layers of the application.
3	Functional Layout	This section explains the functional layout for the various components of the application and details on how they interact with each other.
4	Process Layout	This section of the document explains the development process during the various stages of the application development.
5	Non-Functional Requirements	This section of the document lists the identified non-functional requirements which are essential for the seamless working of the application.

1.2 Design Principles

These principles will help us to create a system architecture that adheres to the proven principles, minimizes the cost, helps in incorporating any changes in the requirements and promotes re-usability and scaling the application.

- Separation of concerns
- Single Responsibility principle
- Principle of Least Knowledge
- Do not Repeat Yourself (DRY) Principle
- Minimize upfront design

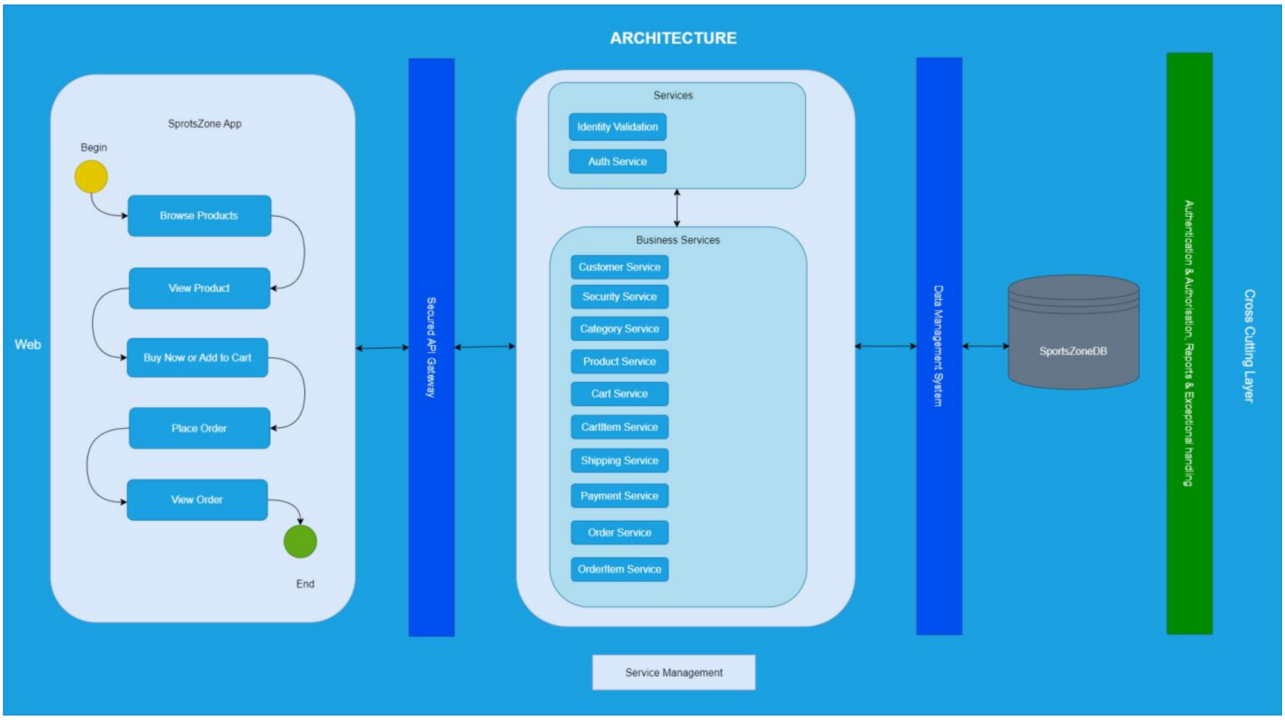
The ability to reuse the components is one of the key principles followed in this approach. In this design, we have identified the components which can act as a Platform for this application and provides an ability to extend it, for different devices considering the future requirements to extend the application.

2. SportsZone Technical Design

This section briefs on the solution architecture which forms the core of the design. The proposed high-level architecture of the **SportsZone web application** is as shown and explained below.

2.1 Architecture

This section of the document gives the main objectives of the **SportsZone web application** to develop a scalable and highly configurable application. The components have been designed to provide an application platform for enabling scalability to build extendable features.



2.2 Architecture Description

This representation provides the logical view of various components which are involved in the **SportsZone web application.**

SportsZone web application UI enables the user to order products via online and workflow management. The solution is designed to identify alternative solution to shopping by purchase anything from home with just one click.

The business services like Customer service,Security service, Category service, Product service, Cart service, Cart-item service, Shipping service, Payment service, Order service, Order-item service hold the core functionalities of the SportsZone application. The other API services like Identity Validation, Auth Service are used for validating patient details.

Identity Validation service is an another add-on feature in API which is responsible for validating the data given by the customer.

Auth Service is an authentication layer of the API which ensures that the customer is valid or not and it also provides session management through limited timestamp and also used to persists the user login.

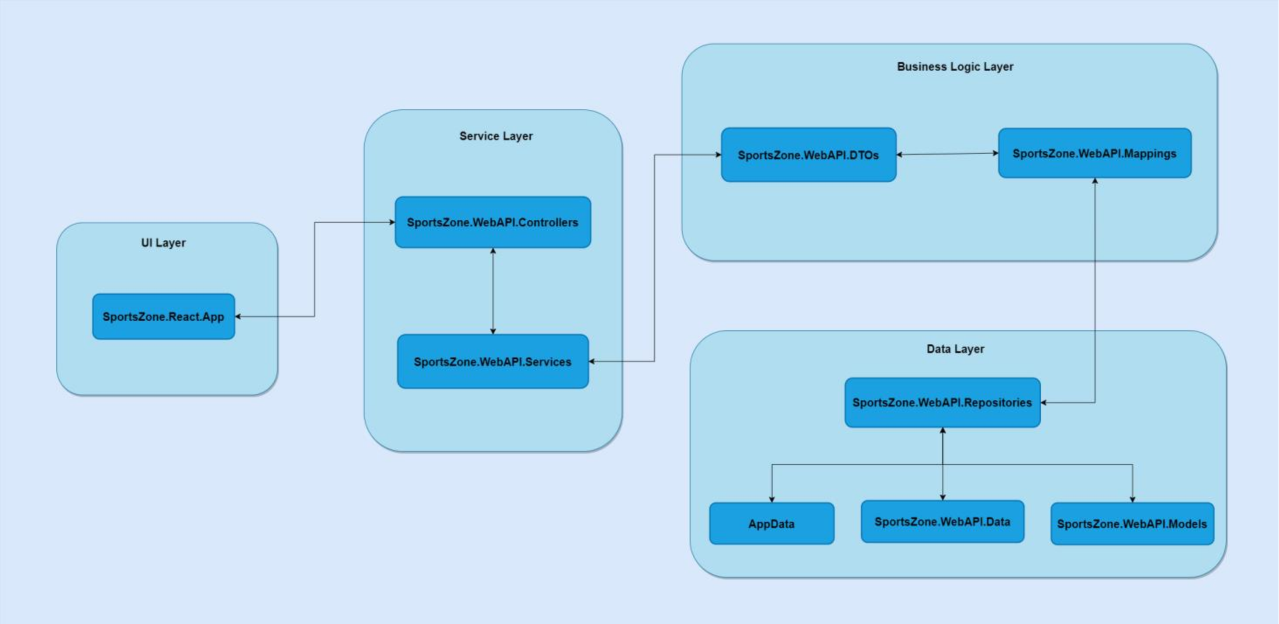
The Secured API gateway will authenticate the valid users to access the service from the User Interface. This service is responsible to handle each request from the user interface and provide the necessary response for them. This layer will communicate with the business service layer to perform certain operations/functionality.

The Data management system connects the Service Layer to Data store.

The cross-cutting layer consist of Security, customer management, Reports & Exception handling.

2.3 Component Level Design

This section provides a detailed blueprint of individual software components, specifying their functions and interactions within a system..



2.3.1 Data Stores

No	Datastore Name	Description
1	SportsZoneDB	This database captures the details of all transactions used while the shopping is made by the customer from account creation to the order placing.

2.3.2 Presentation layer

This layer represents the man machine interface. This layer is responsible for presentation, user interaction, client-side validations, and interaction with the process layer. Responsive web design (RWD) approach has been adapted for the web design aimed at crafting sites to provide an optimal viewing and interaction experience for easy reading and user navigation with a minimum of re-sizing, panning, and scrolling. This design would also aim to have a provision to extend support to other devices such as tablet.

2.3.3 Service Layer

This layer provides access to service, and it contains the implementations of service interaction. The Client will interact with the services of the **SportsZone web application** to populate the data from the Core API / database. All data interactions are handled by REST API calls. It contains the necessary base use cases for the service. It includes the Authentication, Authorization, Exception handling, Security, Handlers, Dependency Injection., etc. It contains the request and response entities of client interface. The request schema from the client interface should be equivalent to corresponding request entity in this layer. The services should handle the request from the client interface and provide the response for it. The service traverse through the Service layer which is used to manage the database operation and SportsZone Service. The service authentication must be evaluated in this layer. This enables us to access the services/endpoints in the system.

2.3.4 Business Service Layer

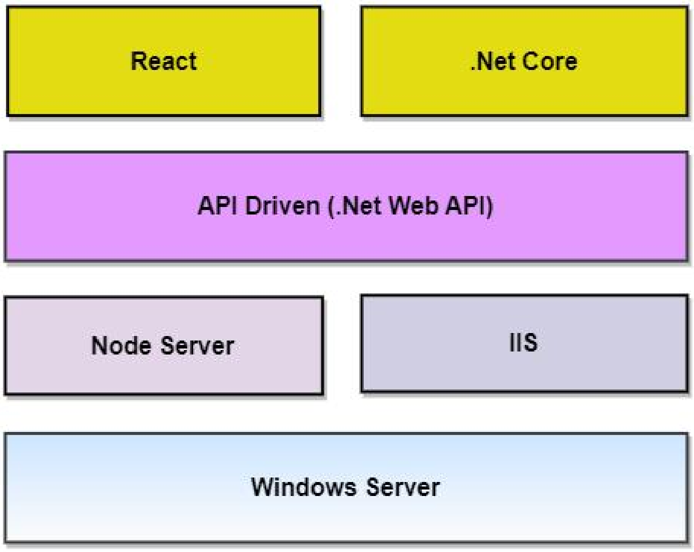
This layer is responsible to implement the business logic and it would take care of external layer communication like Database operation and SportsZone.WebAPI Service. It would map the view entity and business entity for manipulating the data between different layers.

2.3.5 Data Layer

This layer has entities that are used to persist the data into database and compute the logic in business service layer. The database connection/context should be established in this layer. Also, it contains field mapping and the files/ images to be displayed.

3. Technical Stack Layout

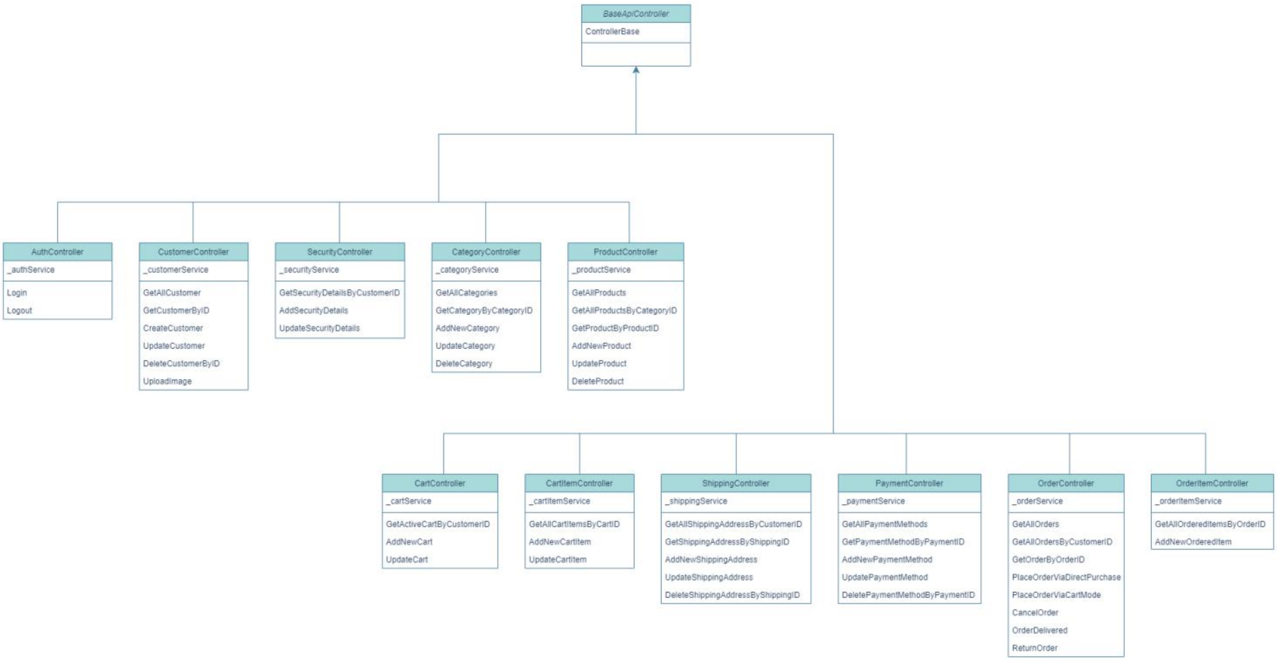
This section comprises the list of technologies which has been proposed for the **SportsZone web application**. In this project, we are leveraging cutting-edge technologies to deliver a robust and modern web application. We are utilizing React v18.2.0 for our front-end, ensuring a responsive and dynamic user interface. Our back-end is powered by .NET Core API targeting .NET 5.0, providing high-performance and scalable server-side capabilities. For data storage and management, we rely on SQL Server, with a minimum version requirement of SQL Server 2019 (as supported by SSMS 2019), ensuring data integrity, security, and efficient data handling. This tech stack combination empowers us to deliver a seamless and efficient application experience to our users. .NET Core 3.1 is a free and open-source, managed computer software framework for Windows, Linux, and mac-OS operating systems. It is a cross-platform successor to .NET Framework.



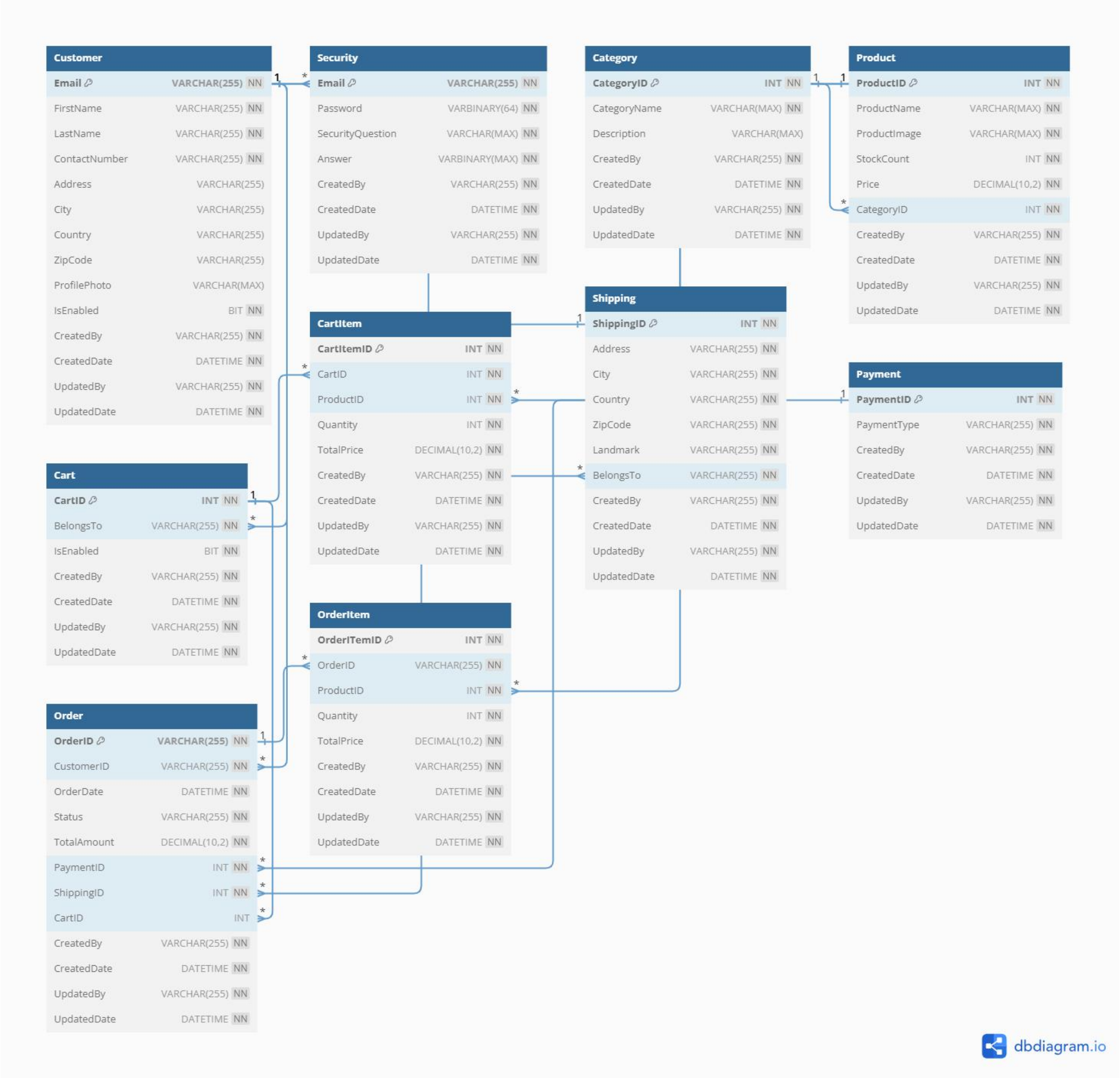
4. Functional Layout

This section describes the interaction between the various functional units of the **SportsZone web application**. This layout depicts use activity, entity relationship and class diagram of the application. This gives a detailed perspective of each unit in the system.

4.1. Class Diagram



4.2. Entity Relationship Diagram



Customer

Email	**VARCHAR(255)**	NN
FirstName	VARCHAR(255)	NN
LastName	VARCHAR(255)	NN
ContactNumber	VARCHAR(255)	NN
Address	VARCHAR(255)	
City	VARCHAR(255)	
Country	VARCHAR(255)	
ZipCode	VARCHAR(255)	
ProfilePhoto	VARCHAR(MAX)	
IsEnabled	BIT	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Security

Email	**VARCHAR(255)**	NN
Password	VARBINARY(64)	NN
SecurityQuestion	VARCHAR(MAX)	NN
Answer	VARBINARY(MAX)	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Category

CategoryID	**INT**	NN
CategoryName	VARCHAR(MAX)	NN
Description	VARCHAR(MAX)	
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Product

ProductID	**INT**	NN
ProductName	VARCHAR(MAX)	NN
ProductImage	VARCHAR(MAX)	NN
StockCount	INT	NN
Price	DECIMAL(10,2)	NN
CategoryID	INT	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

CartItem

CartItemID	**INT**	NN
CartID	INT	NN
ProductID	INT	NN
Quantity	INT	NN
TotalPrice	DECIMAL(10,2)	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Shipping

ShippingID	**INT**	NN
Address	VARCHAR(255)	NN
City	VARCHAR(255)	NN
Country	VARCHAR(255)	NN
ZipCode	VARCHAR(255)	NN
Landmark	VARCHAR(255)	NN
BelongsTo	VARCHAR(255)	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Payment

PaymentID	**INT**	NN
PaymentType	VARCHAR(255)	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Cart

CartID	**INT**	NN
BelongsTo	VARCHAR(255)	NN
IsEnabled	BIT	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

OrderItem

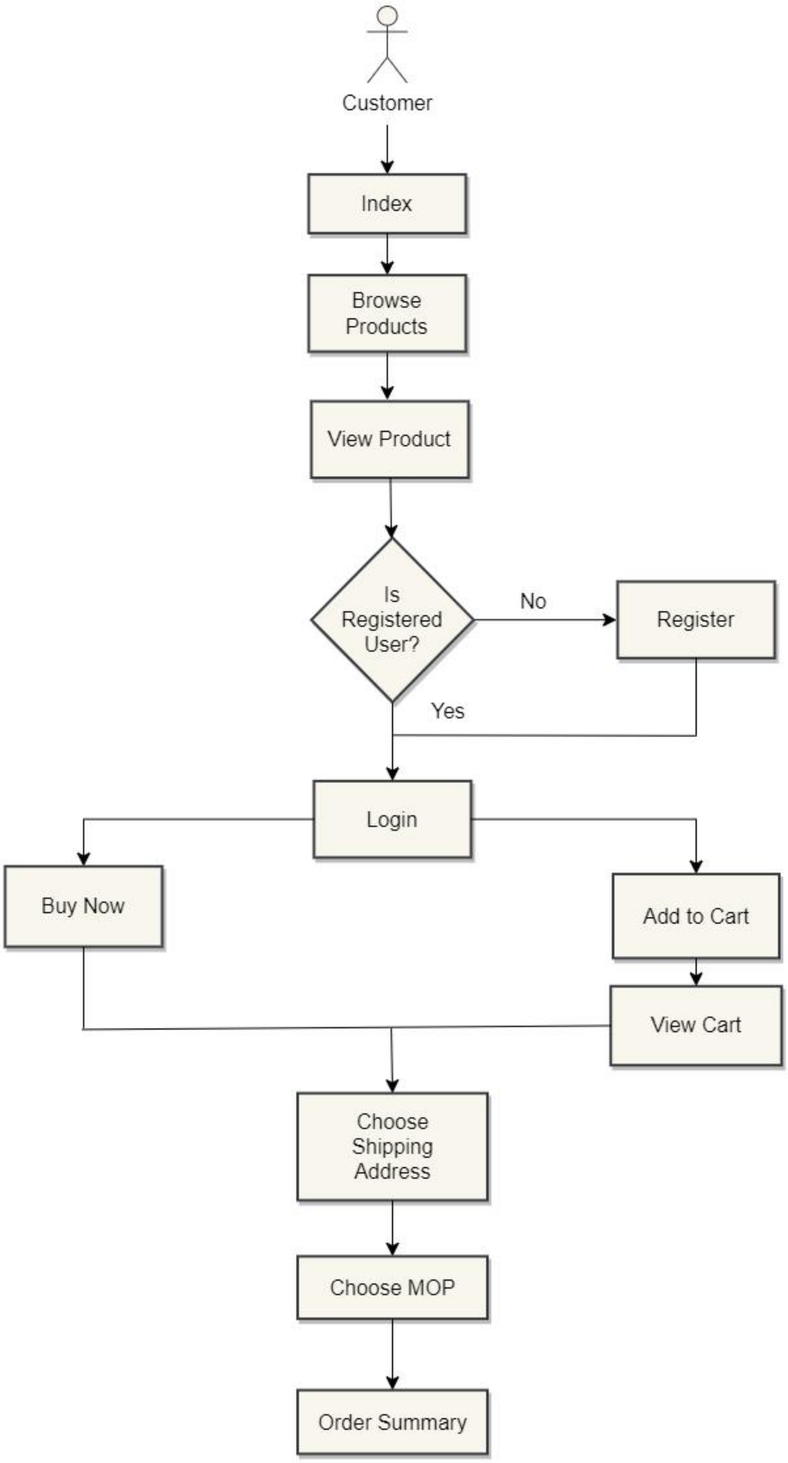
OrderITemID	**INT**	NN
OrderID	VARCHAR(255)	NN
ProductID	INT	NN
Quantity	INT	NN
TotalPrice	DECIMAL(10,2)	NN
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

Order

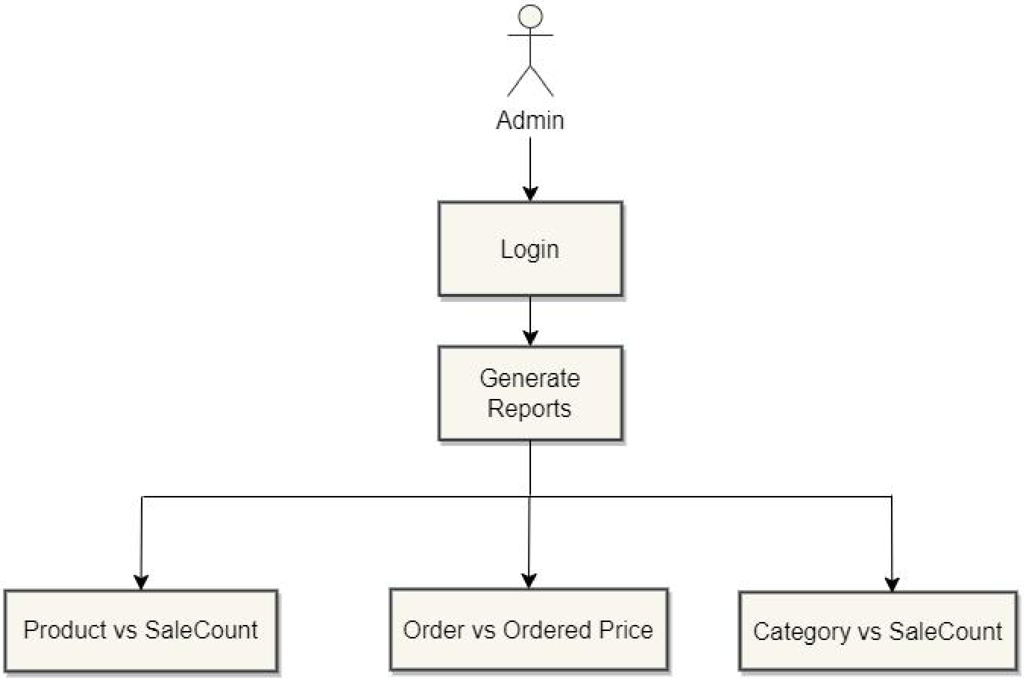
OrderID	**VARCHAR(255)**	NN
CustomerID	VARCHAR(255)	NN
OrderDate	DATETIME	NN
Status	VARCHAR(255)	NN
TotalAmount	DECIMAL(10,2)	NN
PaymentID	INT	NN
ShippingID	INT	NN
CartID	INT	
CreatedBy	VARCHAR(255)	NN
CreatedDate	DATETIME	NN
UpdatedBy	VARCHAR(255)	NN
UpdatedDate	DATETIME	NN

4.3. Activity Diagram

Order Placing - Customer



Generate & View Reports - Admin



5. Non-Functional Requirements

This section will describe the non-functional attributes. These attributes captured here are the possible non-functional requirements identified so far as part of the requirements analysis. This section would be expanded based on the needs of the application.

5.1 Security

The security can be further enhanced through RESTful service calls over HTTPS. The application framework will be used for authentication and authorization of system.

5.2 Performance

Application is designed to deliver exceptional performance by leveraging asynchronous method calls and HTTPS security measures. This combination ensures that our API operates efficiently even under heavy loads while maintaining data security and confidentiality for users. With these features, our API sets a high standard for reliability and performance in the realm of RESTful web services.

6. Glossary

Term	Definition
API	The Application Programming Interface is a service-based interface which contains all the methods used for SportsZone web Application.
DB	Database