Detailed Software Technical Design for Online Shopping Website (FLYBUY)

Table of Contents

1. Introduction

Trends is an e-commerce website for online clothing shopping. E-commerce websites are now the go to shopping for any kind of products in the busy world. More businesses are adapting online shopping nowadays. The customer can view all the products on the website and can make new orders after registering to the website. This saves time for the customers and makes it easy to purchase products from anywhere.

1.1 Purpose

The purpose of the system is to make shopping easy and time saving for customers. While they are in any place they can make orders and purchase the products they like. Customers can view the product details and find different categories of products from categories pages for each section in the website. Customers can select different delivery options like standard and express to get the products on time or earlier if they want.

1.2 Project Scope

- Starting the system implementation by developing the API which handles all the users' requests and developing relevant responses for them with exceptions handling.
- Logging all the request and response flow on the endpoints that comes from the customer.
- Client side development starts whit UI implementation and then with implementing services within components and with the API.
- Unit testing API endpoints if they are responding as expected

1.3 Technologies Used

- Client – Angular
- API - .NET core web API
- Database – MSSQL Server
- Logging - Seri log
- Token service – JWT
- Hashing & Encryption - BCrypt

2 Requirement Specification

2.1 Functional Requirements

- Register Customer
  - User should be able to register to the website
- Login
  - Registered customers should be able to login to the system
- View Products / Categories
  - All users who visits the webpage should be able to view products & categories
- Add products to the cart
  - All users who visits the webpage should be able to add products to the cart
- Remove / modify products in the cart
  - Cart should be able to modified & items should be able to remove from the cart
- Checkout
  - Logged in customers should able to make the checkout if they have products added into the cart
- View Bill
  - After making orders customers should be able to see the bill for the order
- Email bill to the customer
  - On a successful order an order confirmation email should be sent
- View payment history
  - Customers should be able to see all the orders they made

2.2 Non-functional Requirements

- Easy to use
  - Customers should be able to easily lookup things and navigate to pages without confusions. UI should be user friendly and familiar to the customer experience.
- Performance
  - Website should be with high performance and user should not be waiting too long for responses.
- Scalability
  - For the increase of user requests the system should be able to handle it well.
- Maintainability
  - The system must be maintained after release and if any problems occur to the system it must be handled
- Security
  - The security is really important as it is an online shopping webpage. All the requests and response must be secured. User details must be securely stored and only authorized users can perform relevant actions.

3 Core Technical Services Design

3.1 Validation

Error Trigger	Action	Description
User forgets to enter a required field	Display the message “Field name is required”	Display required field message
User enters invalid password	Display the message “Password should contain 1 uppercase, 1 lowercase”	Display invalid password message
User request order bill for invalid order id	Display message “Order not found, Invalid order id”	Display message
User tries to access other users order history	Display message “Unauthorized access”	Return 401
User attempts checkout without log in	Route user to login page	Route to login page
Request for product that doesn’t exist	Display the message “Product not found”	Product not found message

3.2 Error & Exception Handling

In the system at API level we handle the errors & exception. For all exception handling global error handling built in to the .NET core is used. The system handles all the expected exception and also if unexpected exception is thrown the system could handle it without terminating the server.

3.3 Logging

For logging in the .NET core web API Seri log is used. Seri log is a structural logging library for Microsoft .Net and has become the preferred logging library for .Net applications. With the help of Seri log all the users’ activities on the backend is logged and is stored in a table in the database. All the errors are also logged which helps to fix errors and produce error free system.

3.4 Security

Security is a main concern on an online shopping web application. In our system we used multiple security methods.

3.4.1 Authorization

For authorizing the user JWT token is used. JWT tokens allows to authorize the user request to access the relevant routes. Refresh tokens are also used to refresh the JWT access token When they expire at the same time to make a very good user experience to the system. So the users don’t need to login again and again to the webpage the JWT access tokens are refreshed in the background without making poor user experience. At the frontend all the data which are stored in browser are encrypted as well so the data of the user is secured from attacks.

4 Detailed Design

4.1 Tier Architecture

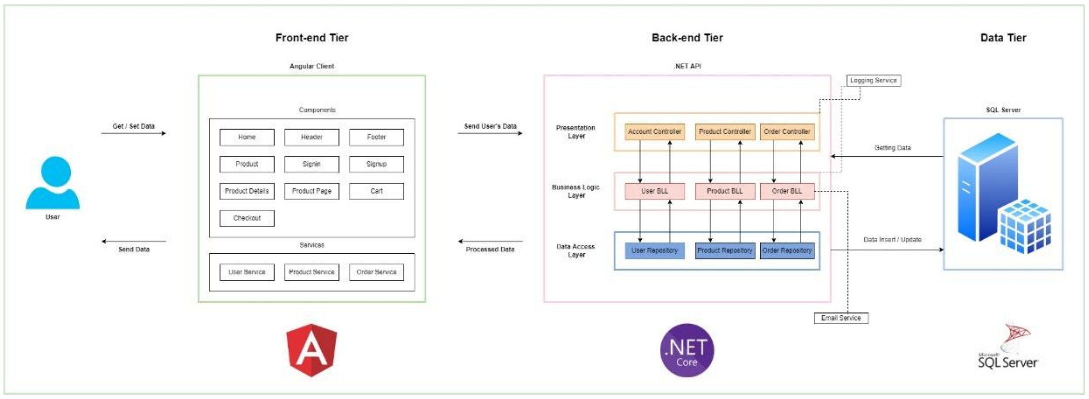

Figure 1

According to above diagram there are 3 main parts.

1. Client Tire
2. Middle Tier
3. Data Tier

4.1.1 Client Tier

The client end, in other word frontend of this project is developed using Angular 13. As the programming language, typescript is used here. This framework use 'mvc' architecture. To select 'Angular' as the frontend development framework, following reasons were considered.

- Highly maintainable.
- Cross platform
- High Efficiency
- Performance and speed
- Supporting 3rd party libraries

4.1.2 Middle Tier

To implement the middle tier, it has used asp.net core as the framework. Due to following reasons, it has been selected this platform for the project.

- Cross – platform
- Open source
- Built-in dependency injection
- Lightweight
- High performances
- Ability to host on IIS, Apache, Docker

As the software architecture pattern, it has been chosen layered architecture. It provides following advantages.

- Re-use of low-level functionalities
- Standardize on implementation
- Helps encapsulation
- It helps well in the early stages of products.

Since the project is not very complex nor large the above architecture was selected over the other architectures.

4.2 Use case Diagram

Registration Process of a Guest User

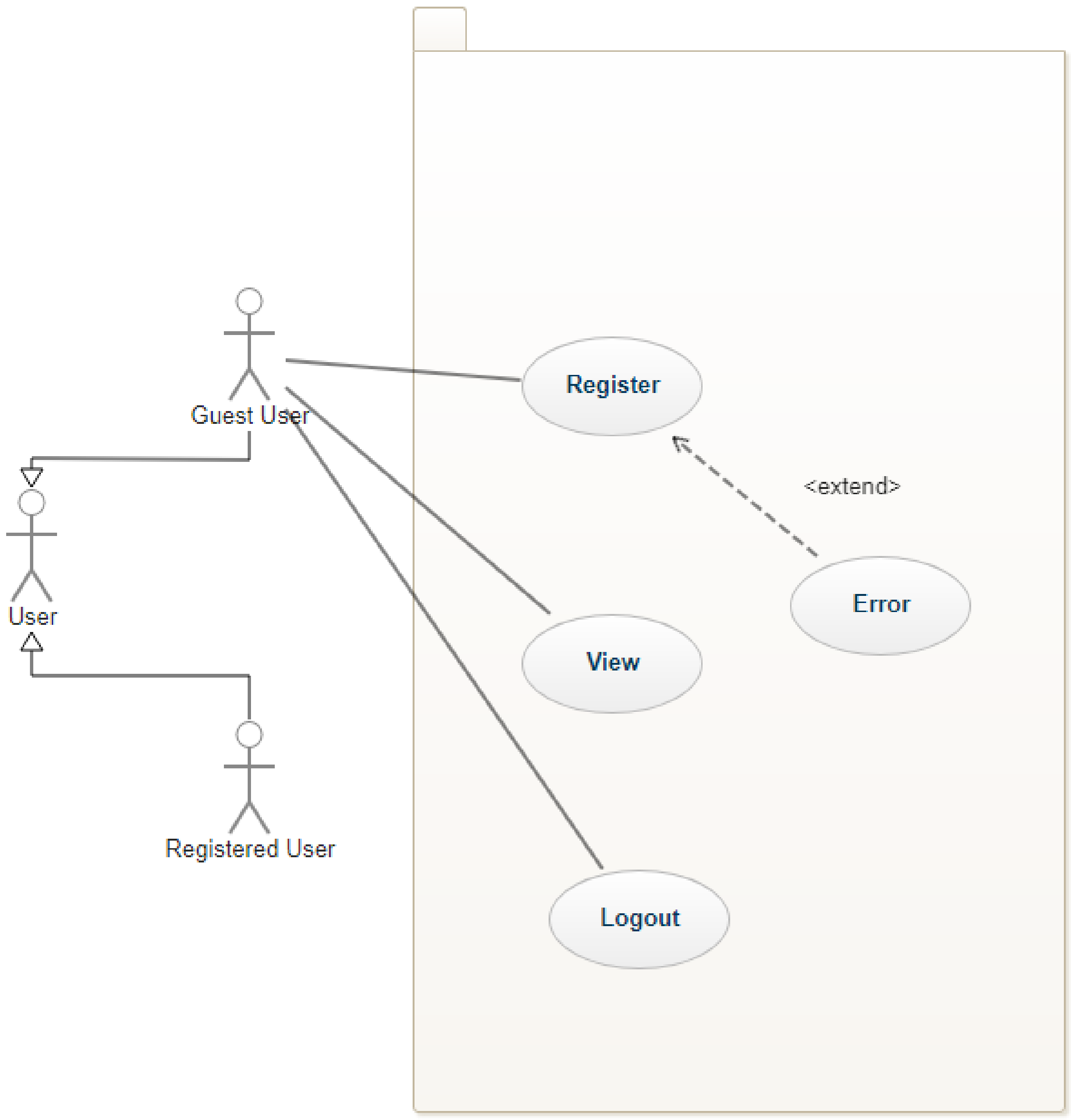

Figure 2

Login Process of a Registered User

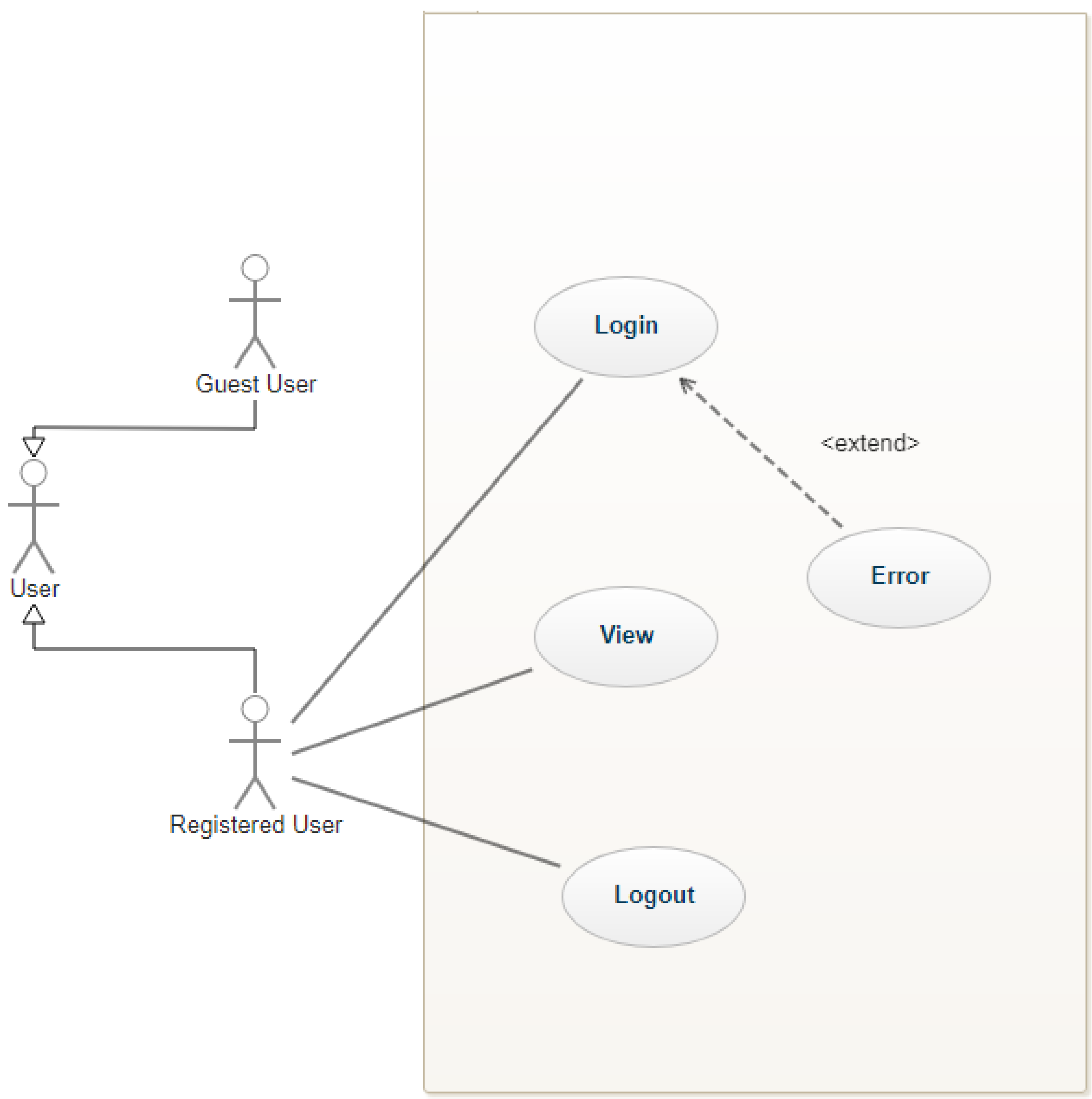

Figure 3

Shopping Process

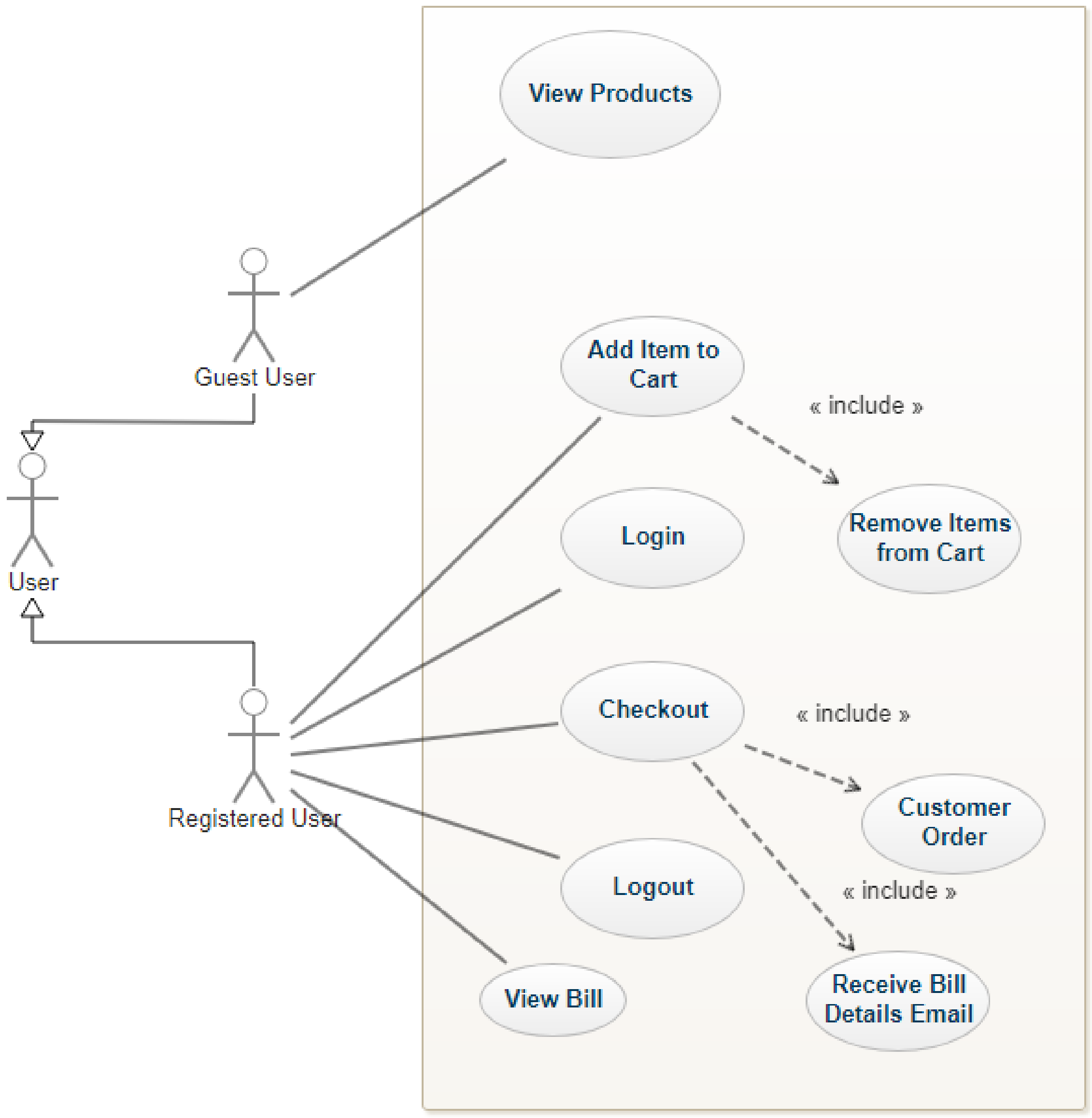

Figure 4

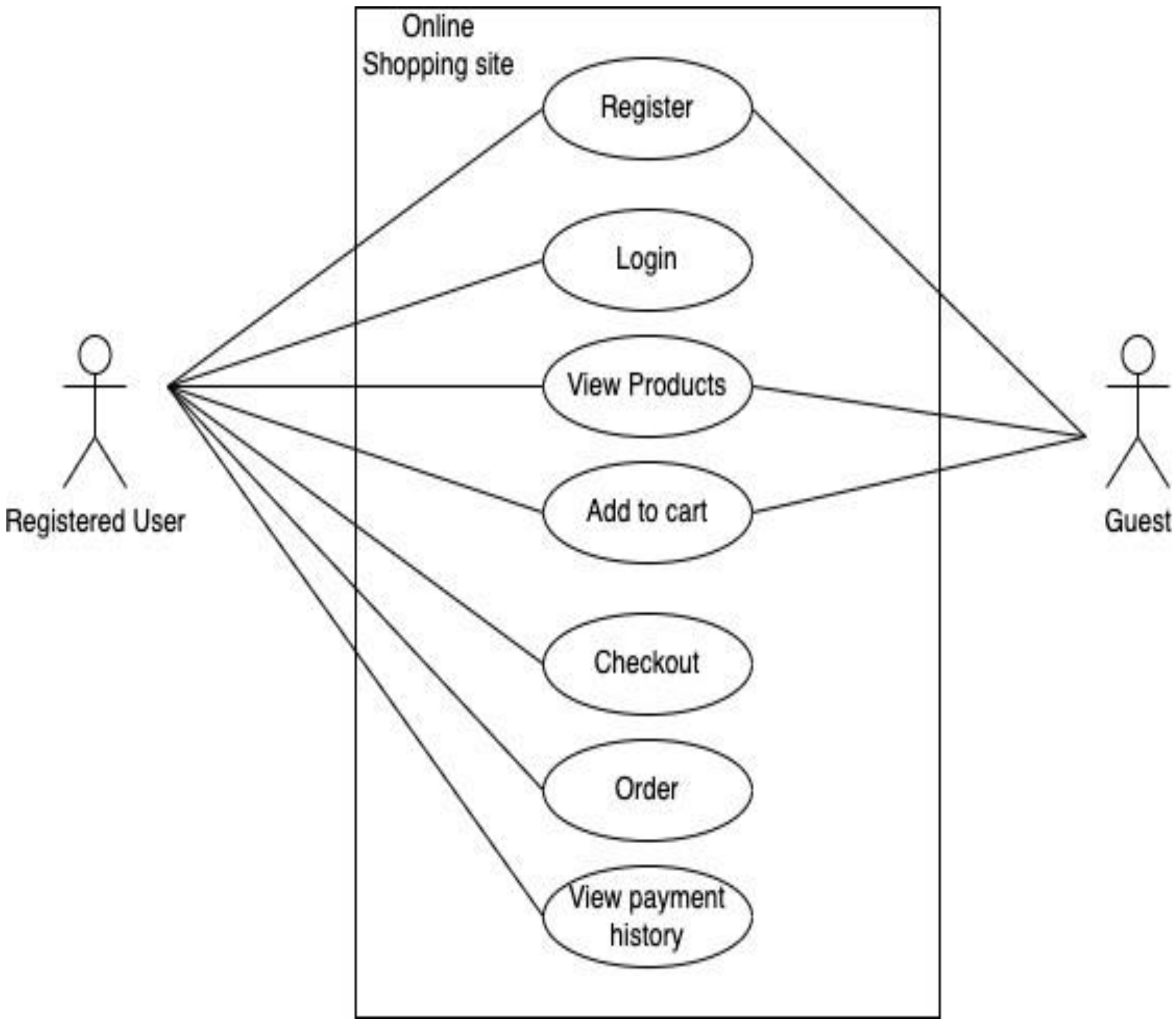
Online Shopping site
Register
Login
View Products
Add to cart
Checkout
Order
View payment history
Registered User
Guest

Figure 5

4.3 Entity Relationship Diagram

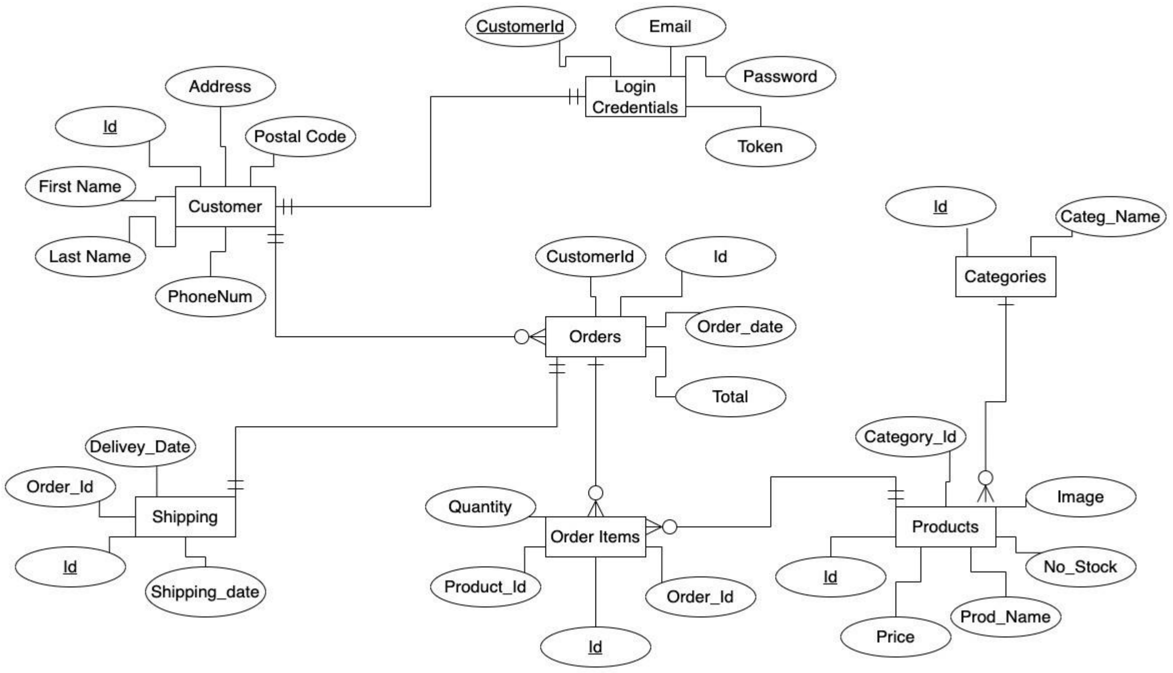

Figure 6