Title: J-SHOES Store

The software requirement document for a website that sells shoes.

Authors: Emmanuel Mojiboye

Date: October 14th, 2024

Version: 1.0.0

Table of Contents

Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

1.4 Technologies' Used

1.5 References

Overall Description

2.1 Product Perspective

2.2 Product Features

2.3 Stages for verifying User Account

2.4 Classes, Methods, and Attributes

2.5 Modeling

2.5 Design and Implementation Constraints

Functional Requirements

3.1 User Management

3.1 Admin Management

3.3 Order Management

3.4 Payment Management

3.5 Financial Management

Non-Functional Requirements

4.1 Performance

4.2 Security

4.3 Usability

4.4 Scalability

System Architecture

5.1 Architecture and its reasons: Why Monolithic Architecture was used

5.2 System Components

5.2 Data Storage

5.3 Integration Points

User Interface Design

6.1 Navigation

6.2 Forms and Screens

6.3 UI/UX Design Images

Test Plan

7.1 Unit Testing.

7.2 Integration Testing.

7.3 System Testing.

7.4 User Acceptance Testing.

8. Deployment

9. Maintenance and Support

10. Appendices’

1. Introduction

1.1 Purpose

The Shoe Store Website aims to provide an easy-to-use platform for J-SHOES Inc. to go about performing its daily business activities. This Website will offer a flexible and smooth medium for J-SHOES Inc. to interact with its customers.

1.2 Scope

The Application will contain a range of features ranging from how Users sign up for an account to different roles the User can perform. There will be different account types in the application for defining different roles like a User account, a Merchant account, and an Admin account. The application will be built with the Agile methodology where we won't just spend years in developing the entire solution but we would break the application into different parts, ship it to Customers and get feedbacks from them.

1.3 Definitions, Acronyms and Abbreviations.

- **Website/Web Application:** This is going to be the application that J-SHOES Inc. Customers can access remotely and carry out various business activities with the Company online. Such business activity includes; Tracking shipments, purchasing products, etc. As we proceed, we will be using this term interchangeably to describe the same thing which is an app for J-SHOES Inc. customers to use.
- **User:** This is an account owner on the website. Everyone who creates an account on the web application immediately becomes a User. But along the line, newer roles that a User might be promoted to a Merchant, or
- **User Account:** This is the set of data that defines a user on a computer system. It includes unique username that identifies it amongst other users and combined with a **password,** allows the user to log in
- **User Roles:** This helps to explain the activities a **User** can perform on the **website**.
- **Authentication:** This is the process of verifying the identity of a person or device. A common example of authentication is entering a **Username** and **Password to login.**
- **Username:** This is a unique identifier for user's account on a computer system. A username combined with a password is one of the most common **authentication** methods used to login to computers, services, and websites.

- **Password:** This is a string of characters that performs part of user **authentication** on a computer system, along with a **Username.**
- **Login:** The combination of a username and password is known as a **login.**
- **Login Details/Information:** It contains fields for username and password. The login details let's the Website know who you are.
- **Customer:** A special type of User that buys shoes on the J-SHOES Store website.
- **Product:** This describes what the Website sells, and in our own case its shoes.
- **Shopping Cart:** This is an online cart user will be using to add multiple shoes on the website.
- **Checkout:** A checkout at a restaurant is a counter or area in a store where goods are checked out, i.e where goods are bought.
- **API (Application Programming Interface):** This is a way for 2 applications to talk to each other.
- **(UI) User Interface:** This describes how the Customers will interact with the Web Application.
- **(UX) User Experience:** This describes the Customers experience with the web application.
- **HTTP (Hypertext Transfer Protocol):** This is an Application Protocol for distributed, collaborative, hypermedia information systems. Http is a language used to communicate on the world wide web. **Clients and Servers** exchange data using HTTP.
- **HTTP Server:** An HTTP Server is a software that processes incoming requests using the HTTP (Hypertext Transfer Protocol) and delivers responses. The HTTP server serves web pages, APIs, or any other content over the internet or a local network. When a client (such as a web browser) sends an HTTP request, the HTTP server interprets it and returns an appropriate HTTP response, often with data like HTML, JSON, or image files. **The needs of the Customer is to build an HTTP Server for a shoe Ecommerce website.**
- **The HTTP Server will:** Serve users both static and dynamic contents, Provide a RESTful API for handling client requests, Perform real-time data handling (using WebSockets).
- **Client:** An application that sends a request.
- **Server:** Another application that listens for requests and sends the required responses.
- **JSON (Javascript Object Notation):** Is used to tell our servers the specific format we want our data in.
- **Business Logic:** This defines the core services our application carries out. It is the backend of our application.
- **REST (Representational State Transfer):** Another way in which APIs are written.
- **REST APIs:** REST is a classification of API based on how they are built. Rest defines a set of functions like GET, PUT, DELETE, PATCH, POST, etc. that Clients can use to access Server data.
- **Latency:** Latency is the delay in transmitting or processing data, this is also the delay before the data transfer starts. To be more specific while referring to our application, we can call it **Network latency,** Network latency takes place during the communication over a network (including the internet).
- **Throughput:** Throughput is how much data can be transferred from one location to another in a given amount of time. In the context of our application, one of this location can be from a database or another API server our application relies on.
- **TDD (Test Driven Development):** This is an approach to approach to building softwares. In TDD, the application is broken down into multiple specs, tests will be written for each spec and the RGR workflow would be used in implementing each spec. This TDD will be used for both Unit Testing and Integration Testing.
- **Spec:** This stands for specification. And a spec defines a specific behavior that your software is supposed to perform.
- **RGR Workflow:** RGR stands for Red Green Refactor; Here, you write a test first, that test must fail before moving on to the Green stage. The Green stage is where you implement the

minimum amount of solution to make your test pass, after the test passes, you get a Green. And finally, you check if you can refactor the solution you've written i.e refactor the code.

1.4 Technologies:

1. Front End: Html5, CSS3, Javascript, jQuery.
2. Backend: C#, ASP. NET Core 8 for building the HTTP Server.
3. Database: MySQL Relational for Structured data Storage.
4. ORM: Entity Framework Core
5. Authentication and Authorization: JWT(Json-Web-Tokens), ASP.NET Identity
6. Payment Gateway: Pay Stack API
7. CDN (Content Delivery Network): Cloudflare
8. Unit Testing: MsTests
9. API Documentation: SwaggerUI
10. Development Tools: Visual Studio Code, Git for source control, GitHub
11. Deployment: GitHub Actions, GitHub Pages.
12. Real-time Data Handling: Web Sockets.

1.5 References

- How I use Merise to create ready to scale databases **; A Jordan Temim Article on Medium.**
- Requirements Document for the Attendance System; **By Atif Naseem.**
- Database Design & Modeling / Entity-Relationship**; By Yann Thierry-Mieg, EFREI 2006-2007.**
- Software Design Tutorial; **From Tech With Tim on Youtube**
- Advice, Tips, Mentoring and Teachings; **From Kimberly Jacques.**

Overall Description

2.1 Product Perspective

J-Shoes Store is a Web Application where Customers can come to and buy a shoe of their choice. The Application will be offering multiple shoes that anyone can buy. Although, only a User would be allowed to successfully order a shoe. From the perspective of the application, a User is someone who has successfully completed the account creation process, and a core part of this process is User successfully linking their Bank Account details to the application.

2.2 Product Features

- **Users must be able to create an account:**

A User who has already created an account by registering for an account. And a User will register for an account by entering his first name, last name, a password twice (one field for password and another field for confirm password), a user name, an email, and a Phone number. User also needs to enter bank account info like the card numbers and credit or debit card CVV details. Although this bank details information is not compulsory for a User registration. This feature will only be compulsory when a User actually tries to checkout a Shoe.

A User cannot have a second account with the same email address. Although one phone number can be used for multiple accounts, but only one email address per account. Collecting the User's phone number details is only a way to collect additional information about the User. This phone number is another way to carry out a KYC. KYC stands for Know Your Customer, and it is a way for a Business to collect valid credentials about the users of their app. Businesses requests such details to avoid fraudulent users from using their applications.

So, the email address will be a way to identify unique Users whenever they use our application.

- **User must be able to login into an account:**

  Logging in into an account shows that a User must have created an account in the first place. Whenever a User logs in into an account, User supplies his username/email and a password. Yes, User should be able to log in into their account whether with an email or a username and their password.

- **User must be able to buy a product:** A product is what the website sells and in our own case, it's a shoe. Although everyone on the internet should be able to see the shoes that are being sold on the website but only Users who own an account on our website can purchase a product. Hence, after a user a User clicks the buy button to buy a product our website they will be forced to create an account on our website.
- **User have an address:** A User's address is needed so the shoe can be delivered or shipped to the User's doorstep. Although, when creating an account, this address is not mandatory. But for a User to be able to fully verify their account, they must supply the details about their address.
- **User must be able to checkout a product:** Checking out a product occurs after a User must have seen the product and would like to purchase the product from the website. The checkout process describes how Users would be able to add a product to a shopping cart, pay for that Product and then the website starts the shipment product for a User.
- **Users must be able to purchase different shoe sizes:** A Shoe size is necessary because it helps User define what kind of Shoe will fit into their legs. Our website would be offering multiple shoes because a shoe size for a 12 year old might be much more different than a shoe for a 32 year old. Ideally the website should be able to show a user the size of that shoe. Some of the ranges of a shoe size that user would see on the website includes: Sizes 20-39 for Male, Sizes 40-49 for Male. And for females: shoe size: 20-29 for Female, shoe size 30-49 for Female.
- **Users can view a detailed shoe information:** User should be able to see a detailed information about the shoe they want to purchase, this information includes images, descriptions. This information should also state some of the reviews of that shoe. This detailed information is only meant for valid Users and not only for everyone on the internet, and by User I mean a person who has actually created an account on our website.
- **Users must be able to filter shoes based on preferences:** The website allows Users to be able to filter all shoes based on their choices, and some of those choices can be Price, color, brand,

size. So a User can filter our website based on the price they can afford. There will be filter options for Shoes less than $20, Filter options for Shoes less than $50 and filter options for shoes greater than $100. User should also be able to input a price range they can afford, these feature requires User to be able to input information into 4 fields named: brand, color, price range, and size. Although the price range is divided into 2 extra fields named minimum price and maximum price. An example of this is: A User saying that they want a shoe with brand **nike,** They want it to be within the price range of **$80 - $210,** and they would like the color blue, and lastly they would like the size 44 for a Male.

- **Website must have multiple shoes for various occasions/activities:** The Website should be able to offer Customers with Shoes for Sports, for Vacations, for the Office, shoes for native wears, Shoes for casual outings.
- **Website must have a Shopping Cart:** The Shopping cart is an online cart that will be on the website. With this feature, anybody on the internet, not only our User can use it to add different Shoes that they would like to purchase from the website.
- **Website must offer customers multiple brands of shoe:** In our website, there would be many brands of shoe that User can purchase. So, our website would be partnering with multiple shoe brands like Nike, Adidas, Jordan, Louis Vuitton, and much more. This will ensure our Users can select any Shoe brand that they like and would love to purchase.
- **Website must show best seller shoes:** In our website, there would be a section that is constantly updated by a list of shoes that have been sold the most. This will be a list of the top 6 shoes our website sold in last 3 months.
- **Website implements a Payment gateway solution for online transactions:** This website will allow Users to make Payments for a Shoe in an easy way. A Payment Gateway is a way for our website to actually collect money for Shoes a User wants to purchase.

2.3 Stages for verifying a User Account:

To become a fully verified User on our web application, each individual must pass the following stages of verification:

i. **First stage:** The Individual details needed for this stage is a firstName, lastName, phoneNumber, dateOfBirth.
ii. **Second Stage:** Individual supplies all the attributes under the address class, attributes such as: country, state, city, streetAddress, postalCode.
iii. **Third Stage:** Individual supplies all the attributes under the BankDetails class, and these details include cardNumber, cVV, cardType(e.g: Verve, Visa or Master card), expirationDate.

After an Individual completes all of the stages above, then such individual can be called a fully verified **User.**

2.4 Classes, Methods, and Attributes:

User

- firstName: string
- lastName: string

- phoneNumber: string
- dateOfBirth: date
- isVerified(): boolean

Address

- country: string
- state: string
- city: string
- streetAddress: string
- postalCode: string

BankDetails

- cardNumber: string
- cvv: string
- cardType: enum
- expirationDate: date

Order

- status: string
- totalPrice: string
- date: string

Shoe

- shoeName: string
- brand: enum
- color: enum
- size: enum
- price: string
- priceRange: enum

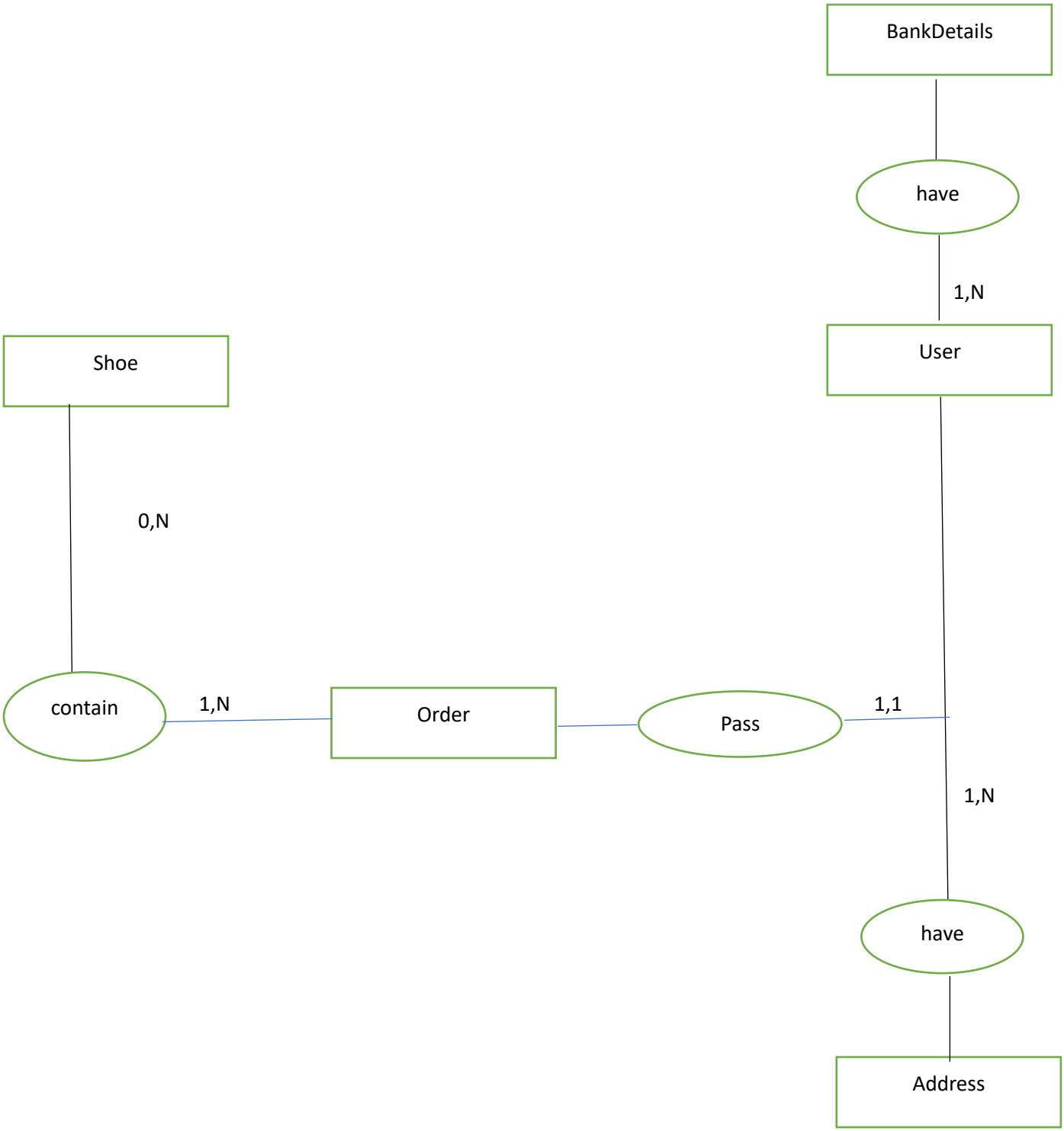
2.5 Conception/Modeling:

I will be Using the Merise Modeling to help in creating my database and describing the entities that exists in the v1 of the J-Shoes Store Application.

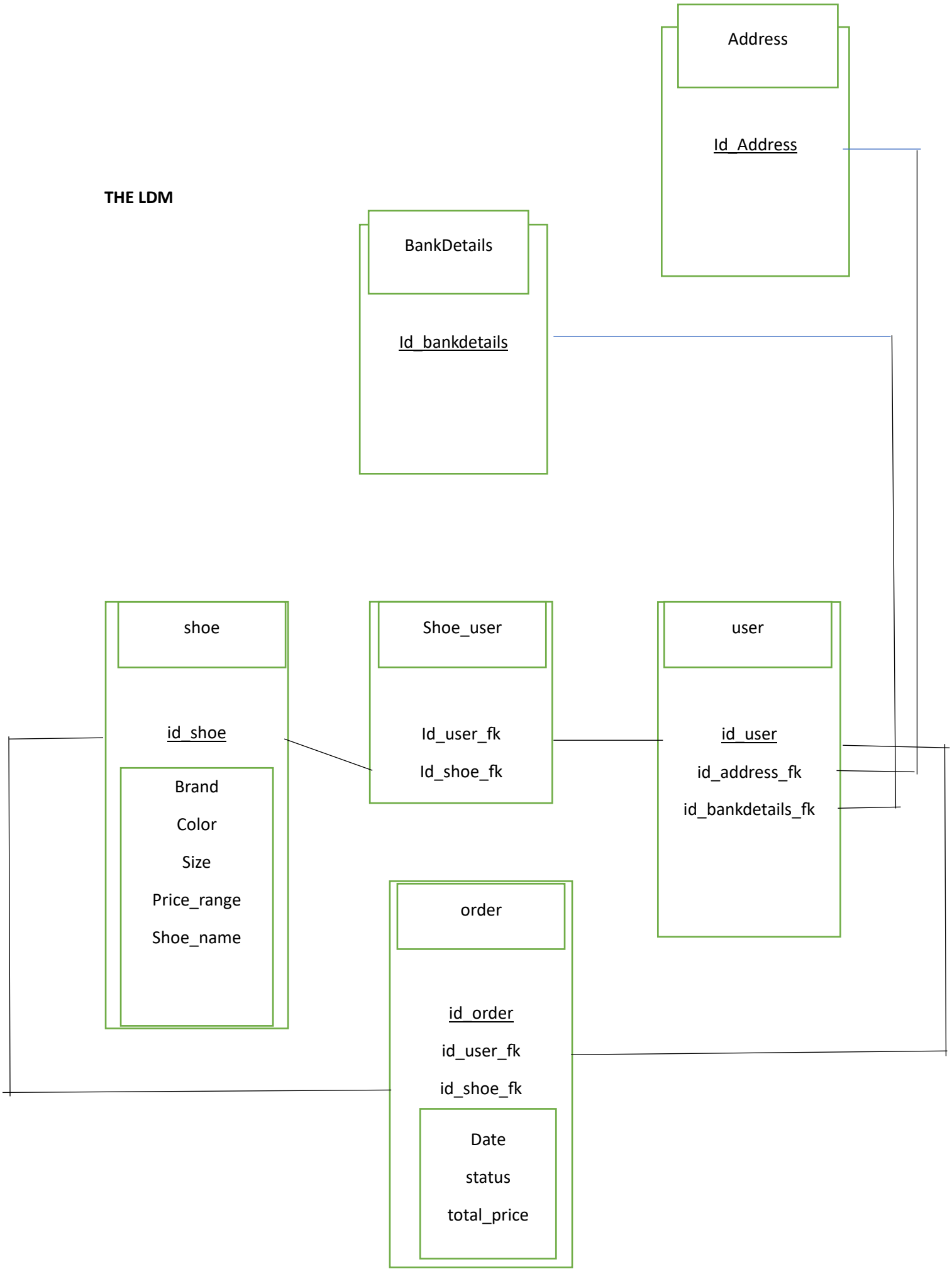
The Business Needs for modeling the V1 of application:

1. A user can buy one or more shoes.
2. A User can have one or more addresses
3. A User can have one or more bank details.
4. An order is passed by a single user.
5. An order can contain at least one or more shoe.

THE CDM


BankDetails
have
1,N
User
Shoe
0,N
contain
1,N
Order
Pass
1,1
1,N
have
Address

THE LDM


Address
Id_Address
BankDetails
Id_bankdetails
shoe
id_shoe
Brand
Color
Size
Price_range
Shoe_name
Shoe_user
Id_user_fk
Id_shoe_fk
user
id_user
id_address_fk
id_bankdetails_fk
order
id_order
id_user_fk
id_shoe_fk
Date
status
total_price

3. Functional Requirements

3.1 User Management

- User can register and log in
- User roles only include: User, and Admin
- User profiles store personal information, addresses, bank details, and roles within the web application.
- Users can read and order shoes from the web application.
- A Default User will be a profile for all other user's irrespective of the newer roles they may attain. This default User will serve as the basis for all other user profiles.
- User Dashboards would be implemented for all Users.

3.2 Admin Management

- Admins can add/edit/delete users and assign roles.
- Admins can create, edit, and delete shoes on the website.
- Every single admin is a User.
- Admin profiles store all User profiles information and some extra information such as; roles, list of all Users, etc.

3.3 Order Management

- All Users can see the orders they placed and track such orders accordingly.
- Orders will be tracked based on order id's. And a typical order id will help in knowing order details such as date the order was carried out, the total price, and the status (pending or fulfilled) of the order.

3.4 Payment Management

- Application will Integrate with payment gateways for online payment.
- Application will track all shoe payments that have been made.

3.5 Financial Management

- Application will record all shoe sales that have been made.

- Application will Generate receipts for Users after buying a shoe.
- Application will generate reports such as Payment details, shipment tracking, and these reports can be exported in various formats (PDF, TXT).

4. Non-Functional Requirements

4.1 Performance

- Low latency(time-interval) in application, response time for User actions should be less than 5 seconds.
- A large amount of data must be supplied to a User in a given amount of time, and thus a higher throughput in the application. The higher the throughput the better for my application, this is because it means more amounts of data would be supplied to my Users in a given amount of time.

4.2 Security

- User authentication and authorization.
- Implementing Json-Web-Tokens (JWT) for an higher level of security for authenticating and authorizing Users.
- Regular security audits and updates.

4.3 Usability

- Intuitive and user-friendly UI.
- Accessible on various devices (responsible design).

4.3 Scalability

- The system should handle an increasing number of users and data.
- The system should be able to handle a growing database.

5. System Architecture

I will be using a Layered(3-tier) Monolithic Architecture for this Project. Layered Architecture is a software design that separates an application into 3 distinct layers each responsible for specific tasks. This structure helps to organize code, improve maintainability and make it easier to scale or test individual parts.

Inside this layer,

- All database interactions like querying, saving, updating, and deleting will be handled by the **Data-access layer (Repositories)**,
- All core logic like Validation, deciding how to process requests, and calculations in the application will be handled by the **Service layer (Business Logic)**,
- All user inputs such as HTTP requests from a web browser or API client will be handled by the **Presentation Layer (Controllers).** And thus enforcing separation of concern.

5.1 Why a Monolithic Architecture was used?

J-SHOES Store Inc. only owns a small to medium sized retail store, simplicity should be a key factor and the Monolithic approach should be sufficient for the business needs. Also, having a Monolithic architecture means all server logic would be in one codebase, and since Client has less than **50,000** Customers it's a good idea to stick with this Monolithic approach.

5.2 System Components

- **Frontend:** Html, Css, Javascript, jQuery.
- **Backend:** C#, Asp.Net Core 8.
- **Database:** MySQL.

For more information about the System components, check out the Technologies section (**1.4**) above.

5.3 Data Storage

- **Data** will be stored in a relational database system.

5.4 Integration Points

- The System will integrate with external systems, including financial software and email services through APIs.

6. User Interface Design

6.1 Navigation

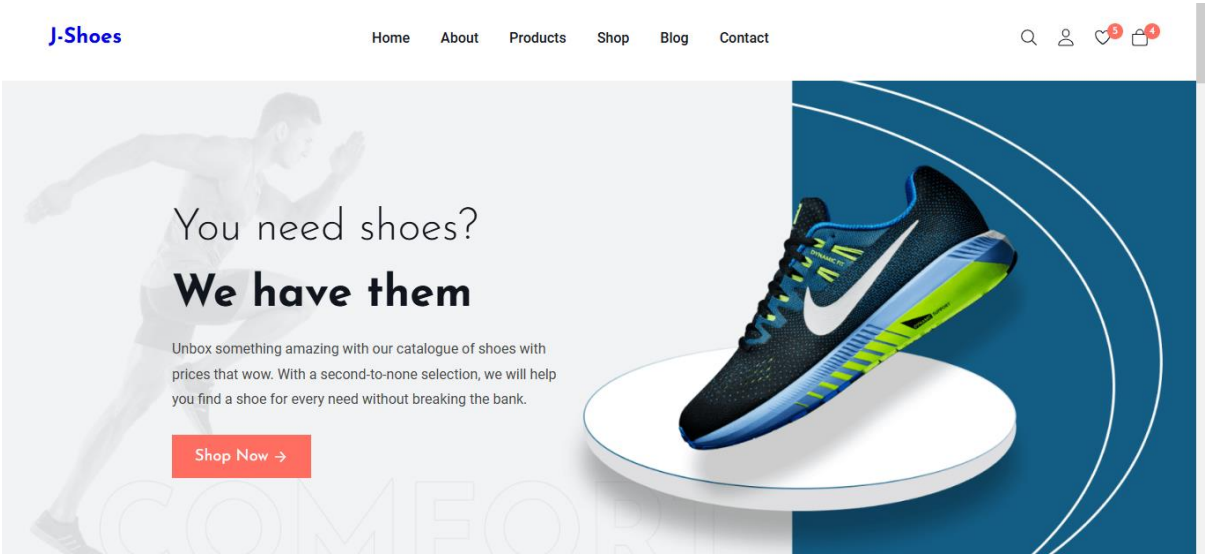
- User-friendly navigation menus and links.
- Dashboard for quick access to key features.

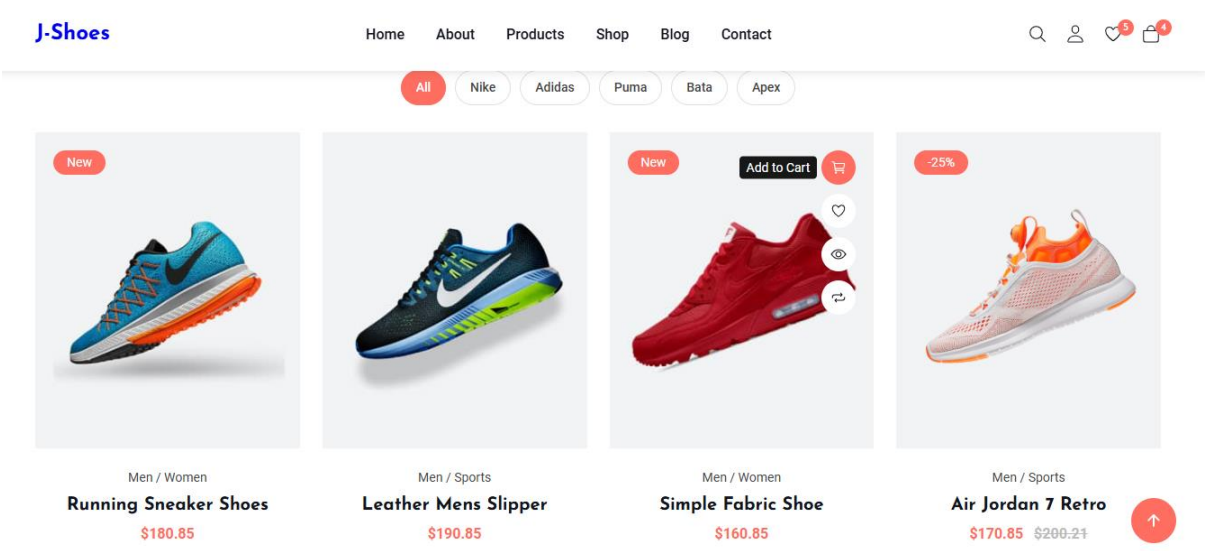
6.2 Forms and Screens

- Consistent and visually appealing forms and screens.
- Mobile-responsive design.

6.3 UI/UX Design Images

Below are some of the Images of what the Frontend of the Application should look like:





J-Shoes

Home About Products Shop Blog Contact

New Add to Cart

Men / Women

Nike Air Max 270 SE

$120.85

Men / Women

Adidas Sneakers Shoes

$100.85

Men / Sports

Nike Basketball shoes

$120.85

Men / Women

Simple Fabric Shoe

$100.85

J-Shoes

Home About Products Shop Blog Contact

Bestsellers Products

All Nike Adidas Puma Bata Apex

New

Men / Women

Men / Sports

New

Men / Women

-25%

Men / Sports

J-Shoes

Home About Products Shop Blog Contact

Nike Special

New

Men / Women

Running Sneaker Shoes

Men / Sports

Leather Mens Slipper

New

Men / Women

Simple Fabric Shoe

J-Shoes

Home About Products Shop Blog Contact

MEN COLLECTIONS

Explore All →

WOMEN COLLECTIONS

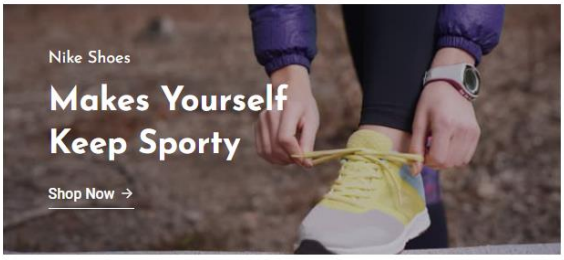
Explore All →

SPORTS COLLECTIONS

Explore All →

J-Shoes

Home About Products Shop Blog Contact



J-Shoes

Home About Products Shop Blog Contact

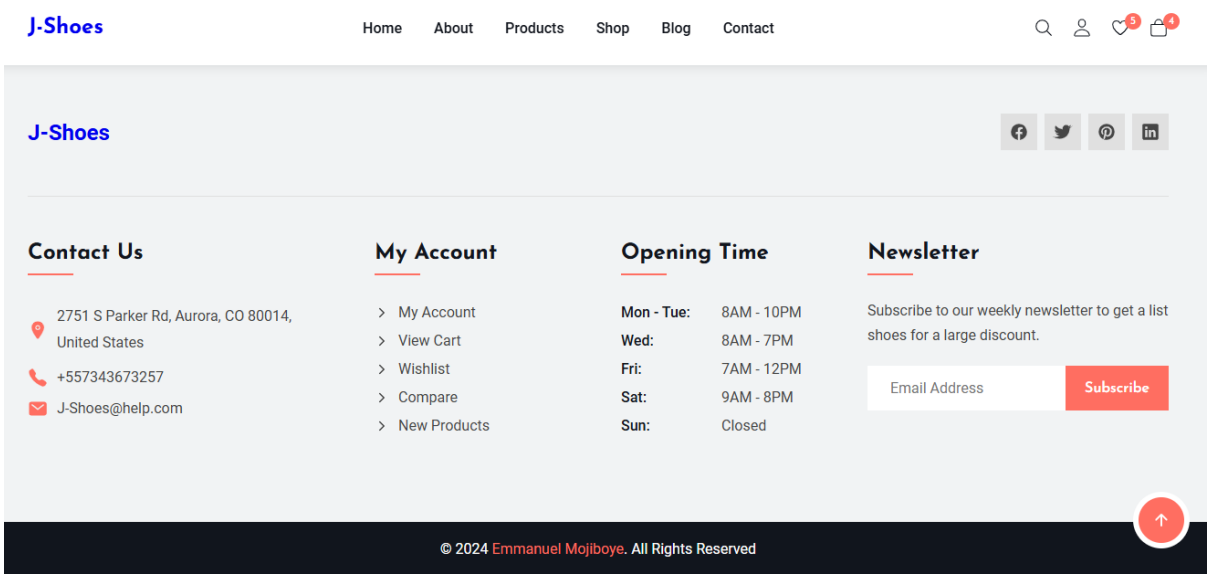
FREE SHIPING
All orders over $150

QUICK PAYMENT
100% secure payment

FREE RETURNS
Money back in 30 days

24/7 SUPPORT
Get Quick Support

J-Shoes



7. Test Plan

7.1 Unit Testing

- Testing of Individual components, and functions, and modules.
- Writing and Testing individual specs. A spec helps to define a specific behavior of an application.
- Using the TDD approach, and following the RGR (**Red-Green-Refactor**) workflow.

7.2 Integration Testing

- Ensuring different modules work together as expected.

7.3 System Testing

- Overall system functionality and performance testing.

7.4 User Acceptance Testing

- Testing with real users to ensure usability and functionality.

8. Deployment

- The application will be deployed on a secure server with regular backups and updates.
- Utilizing GitHub Actions for a smooth and effective Continuous Integration/Continuous Development (CI/CD).
- Utilizing GitHub pages for deploying multiple frontend of application. This will help in making decisions on User Interface/User Experience.

9. Maintenance and Support

- Maintenance is a very important stage in the Software Development Lifecycle, without maintenance and regular updates, a software will struggle and may not live for long.
- Ongoing maintenance, updates, and technical support will be provided as needed.

10. Appendices'

- How I use Merise to create ready to scale databases **; A Jordan Temim Article on Medium.**
- Requirements Document for the Attendance System; **By Atif Naseem.**
- Database Design & Modeling / Entity-Relationship**; By Yann Thierry-Mieg, EFREI 2006-2007.**

The J-Shoes Software Requirement Document serves as a comprehensive reference for the development and implementation of a web application where User's are capable of buying shoes and performing other transaction related activities. The document outlines the system's functional and non-functional requirements, ensuring a clear path for successful development and deployment.