Abstract

A web-based application can be understood as a program, that allows users to connect with a remote server, using a web browser interface. Unlike traditional websites (client-server) whose application logic gets rendered on users workstation, a web-based application has its application logic rendered on web server (Bourne & Brown, 2013). 'Mingo Shoes' is an interactive, web-based, and dynamic website that provides users an experience of online shopping. Developed using .NET technologies, the website is small, well designed and has specific roles for both users as well as system admins. MVC architectural pattern is used throughout the process where program logics are divided into Model-View-Controller. Similarly, Entity Framework is used for creating databases and perform data operations. The website offers variance of functionalities including, authentication, authorization, profile management, order management, product categorization, purchase history and many more.

Table of Contents

1. Introduction

'Mingo Shoes' is a mobile retail website that serves customers as an online e-commerce platform, allowing them to view, select and purchase shoes of their choice. Considering the craze of online shopping in today's world, the website focus on making purchase easier and more convenient for users. From available products, users can explore, select and order shoes based on their unique preferences. With appropriate use of software engineering technologies and .NET technologies, the website aims to solve user problem to some extent.

1.1. Project Background

The project is to build a web-based application, suitable for mobile or handheld devices. The website is built dynamic and interactive ensuring that users can complete certain actions on it. The technology used during this website development is ASP.NET. The development process tools include, .NET6, VS Studio 2022, SQL Server Management Studio (SSMS), .NET packages and other frameworks depending on system attributes. The data from our website is stored in server-side SQL Server database making sure of privacy and other malicious parties. Considering the MVC architectural pattern that we are using, we need to create model which will resemble database table. Then we establish database connection to Entity Framework and use code first approach for out system development.

The project spins around User Interface (typefaces, colors, iconography, etc), Human Computer Interaction (the way user and device work together), User Experience (focused on users), ergonomics and other crucial factors, making it user-friendly and convenient for users. With all these considerations, the web-based application project is expected to offer users an experience of online shopping, focusing specifically on footwear.

1.2. Project Aim

The aim of this project is to implement a dynamic retail website that operates on standard browser and has a full-fledged computerized system to automate current manual system.

1.3. Project Objectives

- To use suitable programming language to accomplish task, according to requirement
- To build a futuristic website that will be easy to enhance and maintain
- To accumulate and analyze data relevant to footwear industry
- To design a website that follows design principles and guidelines
- To build a website that is dynamic, interactive and user-friendly
- To provide necessary guidance to users, ensuring convenience

1.4. Project Scope

The system will permit users to purchase shoes, sandals and other footwear accessories according to user preferences. One can simply log in to the web-based app, select an item of his/her choice and order it right after. Moreover, users can save items to wishlist, manage profile and rate transactions as well.

1.5. Targeted Audiences

'Mingo Shoes' is a unisex footwear store that offers all sort of footwear and accessories. However, the platform is slightly more focused on audiences, between the age group of 13 to 39. The main reason behind this is younger generation's desire to wear stylish, quality and selective products. Despite this, the website offers products to audiences of every gender and age group.

1.6. Project Schedule

To demonstrate detailed schedule of this project, a gantt chart is created. In simple words, a gantt chart is graphical illustration about how a project is divided and scheduled within specific time period. In our case, the project initiated with a research in first and second week of August, followed by other tasks including planning, designing and development. Orange bars in the gantt chart refers to task duration, whereas first red flag and second red flag denotes the completion of front-end section and back-end section respectively.

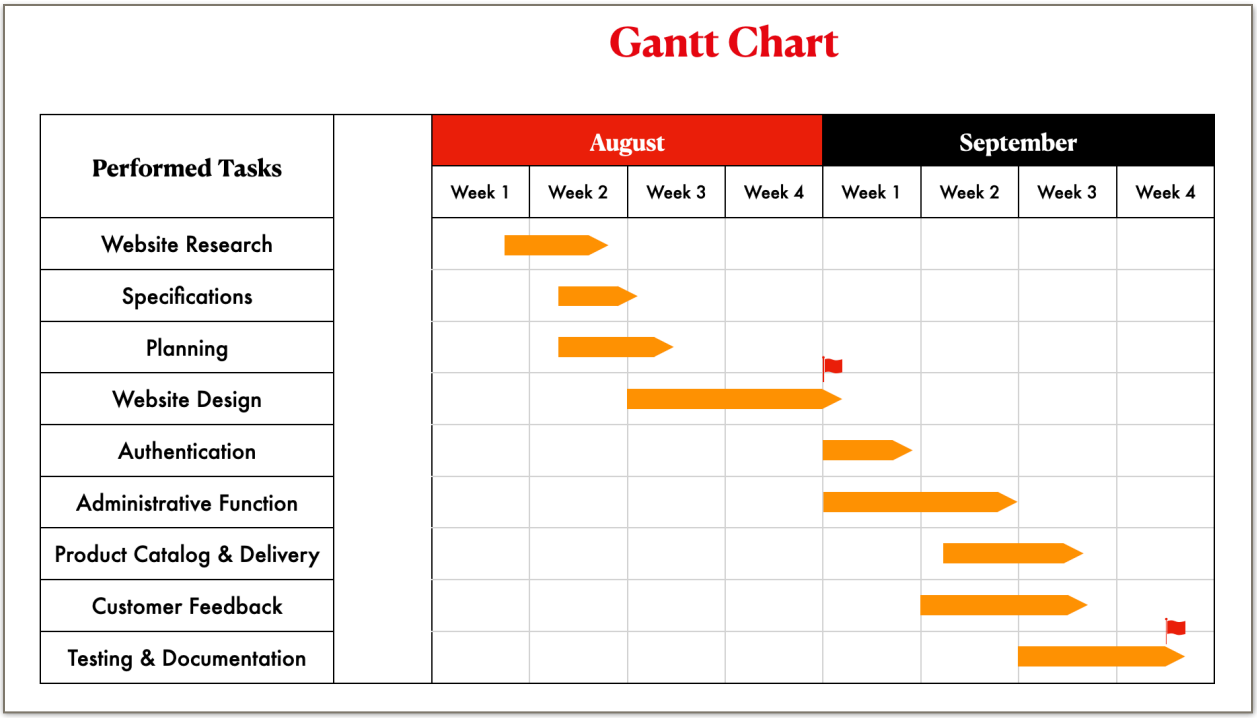


Figure 01: Project Schedule Timeline in Gantt chart

1.7. Major Functions

Mingo Shoes offers similar functionalities to that of any normal e-commerce platforms. A user can log in to the website, explore available products, save them and purchase those items at any time. Getting into details, some of other functionalities include :

- **CRUD :** Create, Read, Update and Delete operations
- **Category** : Add a New Category, Edit, Delete and Notify Users
- **Product :** users can save and purchase product while admin can do CRUD operations
- **Cart :** customers can add items to cart before finalizing the purchase
- **Checkout :** customers can checkout after completing their purchase
- **Search :** customers and admins can search available products on the website
- **Orders :** admins can manage, authorize and reject an order

2. Requirement Specification

2.1. Audience Modelling

First and foremost, audience modeling refers to an approach that finds unique and high-value consumers which can be used in both tactical or strategic marketing purposes. Considering size and complexity of websites and other applications, there is a high chance most of them will face issues related to maintenance, delays, unreliability and development backlog. Therefore, a detailed regulation of discipline needs to be implemented so that a trustable structured approach is followed while designing, developing and maintaining websites (Wang et al., 2005).

A suitable approach is Web Semantic Design Method (WSDM), which is an audience guided method, specifically for websites and web-based applications. Once a mission statement is expressed for the website, the next thing is audience modeling and to focus on target audience. Analyzing different types of users for our website (eg. teenagers, college students, young adults, etc.), WSDM allows us to classify them in audience-class. Diving into details, the audience modeling and its classes is further divided into two types : audience classifications and audience characterization. Audience classification identifies similar users and group them together while audience characterization determines pertinent characteristics for all audience classes. Using this approach, users can be studied in detail and used for marketing and enhancing the service of website as well (Troyer et al., 2008).

2.2. Audience Classification

For our project website, we will have two types of system users. They are :

i. System Admin :

The system admin plays an important role in handling products, transactions and other website contents. System admin has an influential role of controlling users, products, payments and deliveries. Moreover, admin can also analyze sales reports, authorize payments and remove users who seem to be unethical. Unlike customers, a system admin is directed to an admin panel

after entering their credentials. They can create a product, edit descriptions, update prices and delete them as well. Customers actions are monitored by admins and authorized accordingly.

ii. Customers :

Customers are users who visits website, log in with credentials, view and order desired products. They are not provided with same functionalities as system admin, but features which will be essential while purchasing a product. A customer can however, manage his profile, make necessary transactions and provide feedbacks to products and transactions. Unlike admin, they are not permitted to create, update and delete specific products from the website. Actions of all customers are dependent on admin. For instance, a customer can select a product and order that as per his wish. However, the delivery authorization will be provided by the admin. Once the transaction is accepted by admin, the order is considered well equipped for being delivered.

2.3. External Requirements

Stakeholders and consumers will have to understand minimal requirements and conditions to operate websites. The basic requirement for operating our web-based application includes :

i. Operating System (OS) Requirements :

- **PCs and Laptops :** Windows 7 and later, MacOS 10.10 and later
- **Handheld Devices :** Android 7 and later, iOS 11 and later

ii. System Hardware Requirements :

- A minimal of 2GB RAM
- A minimal of 16GB free storage
- A CPU with minimal of 2GHz frequency
- A minimal of 4 Mbps stable internet connection
- A monitor with minimal of 1024*768 resolution

iii. Browser Requirements :

- Mozilla Firefox
- Google Chrome
- Safari
- Microsoft Edge

3. Design

Design is an integral part of system development process. In general, system design refer to the approach of integrating system architectures, modules and engineering regulations to allow understanding of affluent functioning systems. System design can be understood as a bridge that that links problem domain and system through a manageable path (Bidgoli, 2003). When Software Requirement Specification is provided, it is then time to convert those necessities which decides how a system functions to achieve a goal. More to that, a design includes how a system actually looks and feels while using it (Sutcliffe, 2016). It includes diagrams and patterns such as wireframes, colors, user flows, hierarchy and design systems.

3.1. Design Tools and Frameworks

Throughout the design process, a lot of design frameworks and tools have been used with the intention of achieving clear and user-focused design for our system. To begin, progressive enhancement approach of design is implemented to build a responsive mobile application. This approach focus on designing interfaces starting from smallest screen to the largest screen. For this, languages such as HTML, JS along with frameworks such as bootstrap and CSS5 are used.

Figure 02: Languages and frameworks used in web designing

3.2. System Flow Chart Diagram

i. System Flow Chart

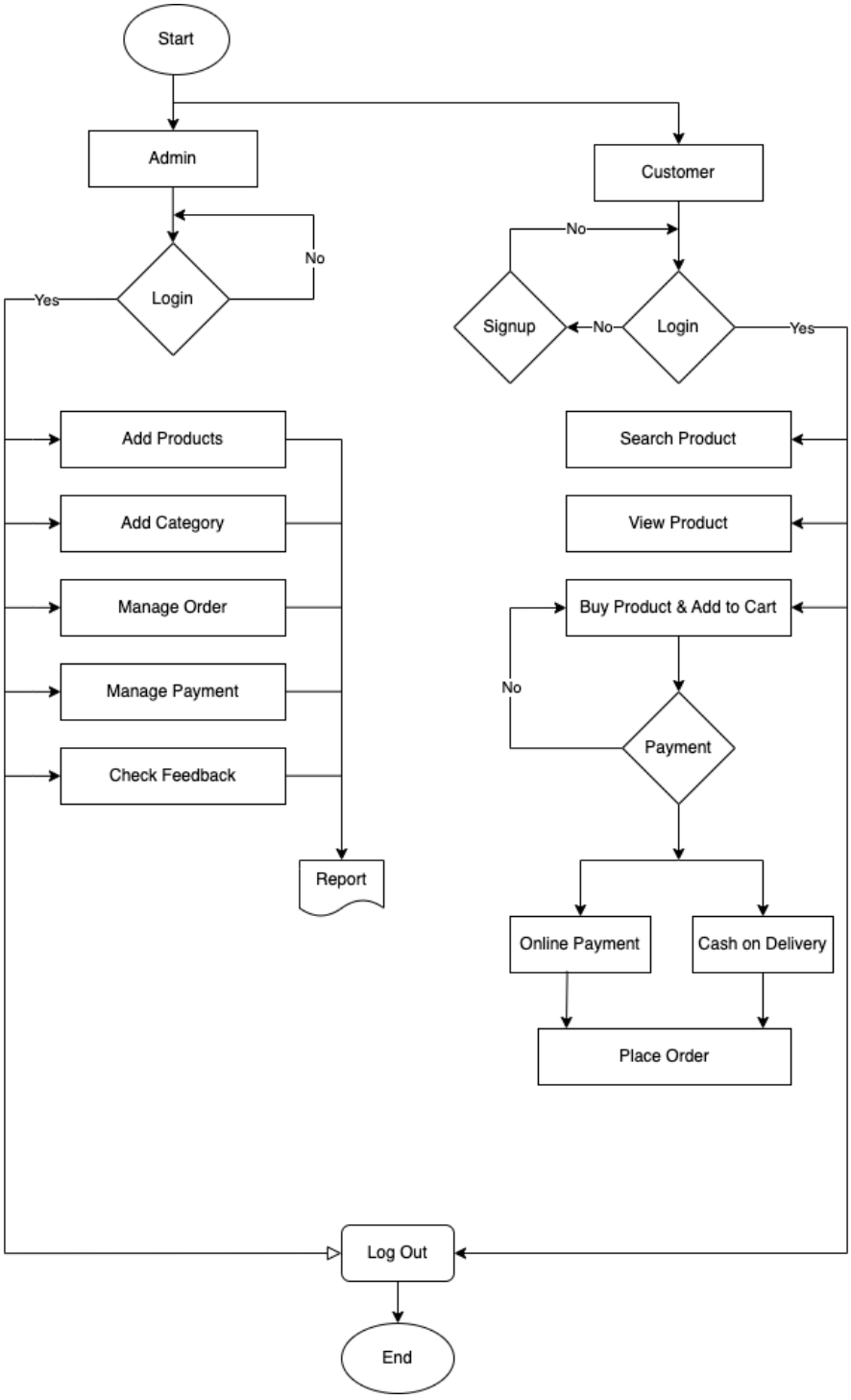


Figure 03: System Flow Chart of Mingo Web Application

ii. Data Flow Diagram

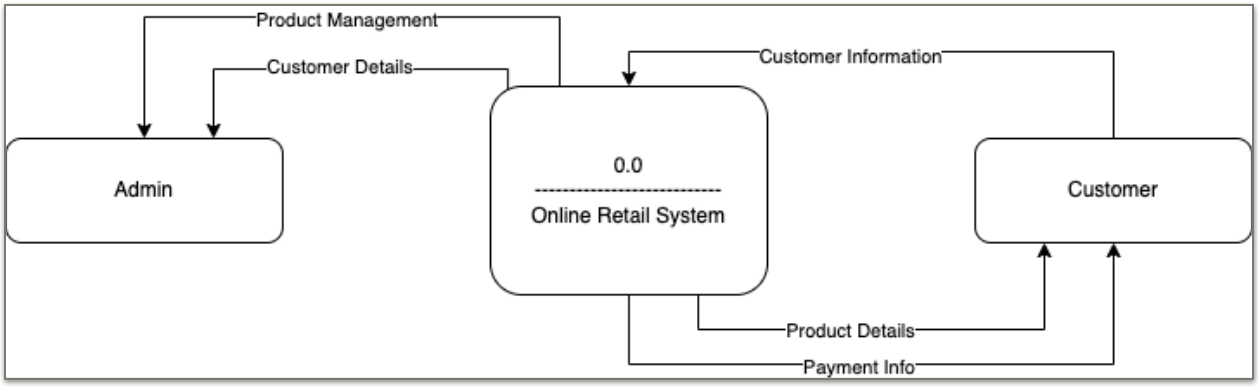
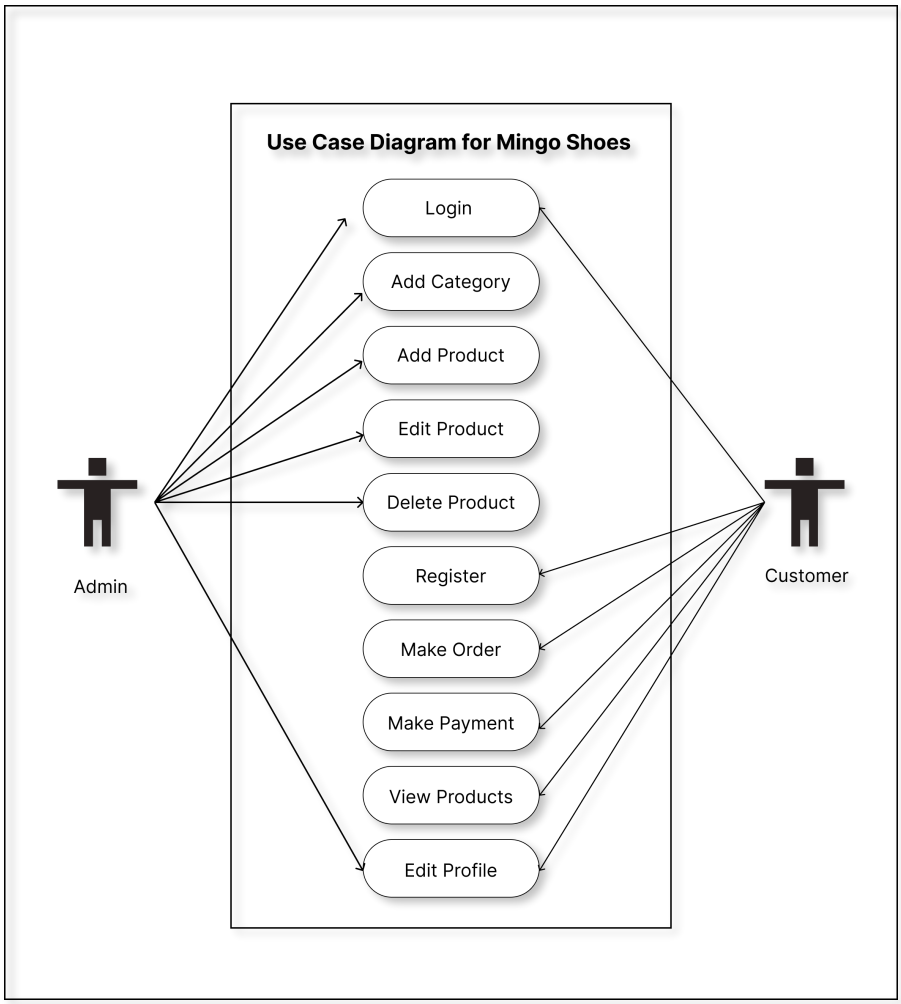


Figure 04: Level 0 Data Flow Diagram

iii. Use Case Diagram



3.2. Database Structures

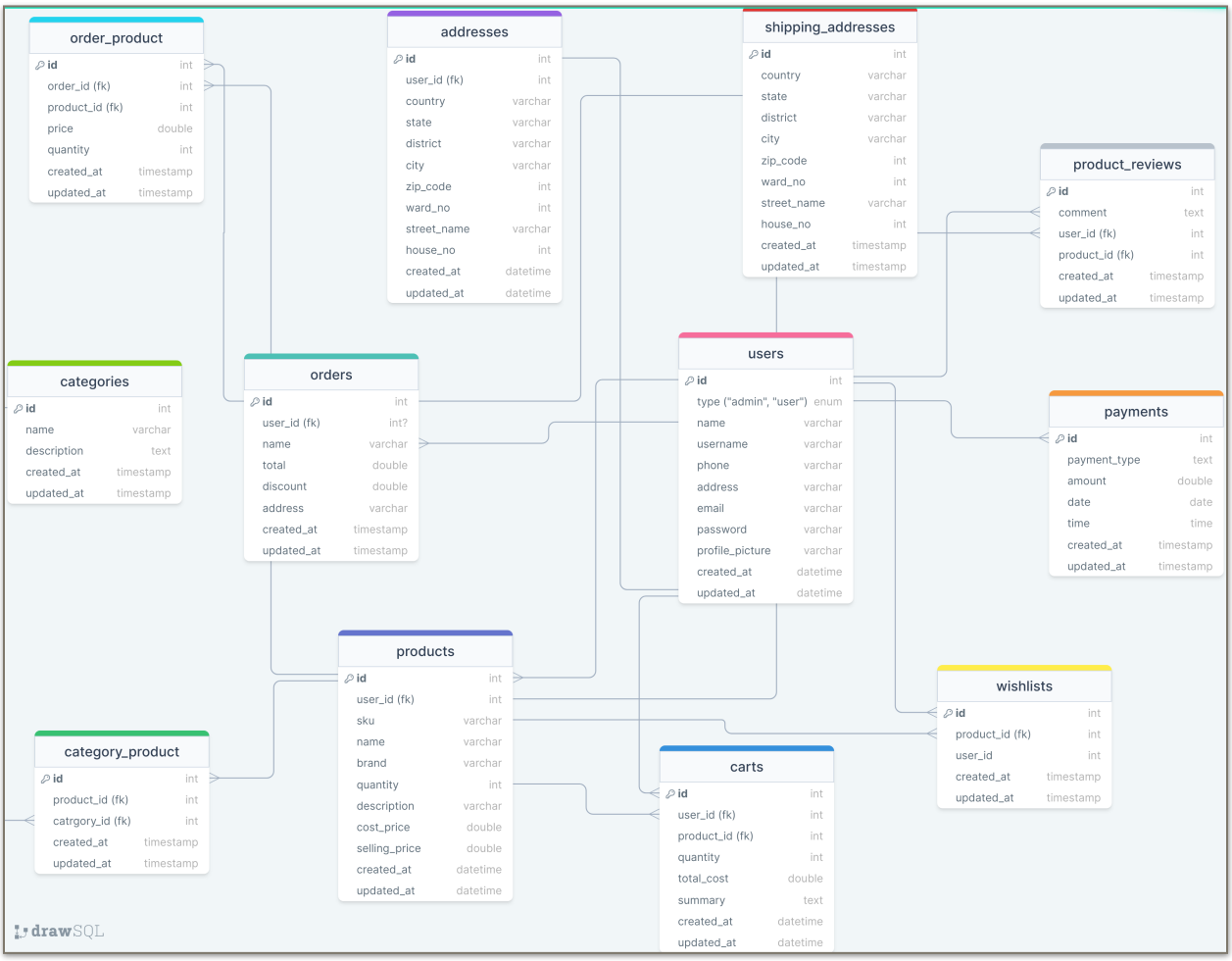


Figure 06: Database Diagram for Mingo Web Application

3.3. Wireframes

Following systematic UI pattern, wireframes of out web-based application are created so that user experience remains utmost and necessary changes can be made accordingly. Analyzing its usability, wireframes are modified during development phase as well. The wireframes are not the final products but needs more refinement. Diagrams of our system wireframes are shown below :

i. Wireframe for Landing Page

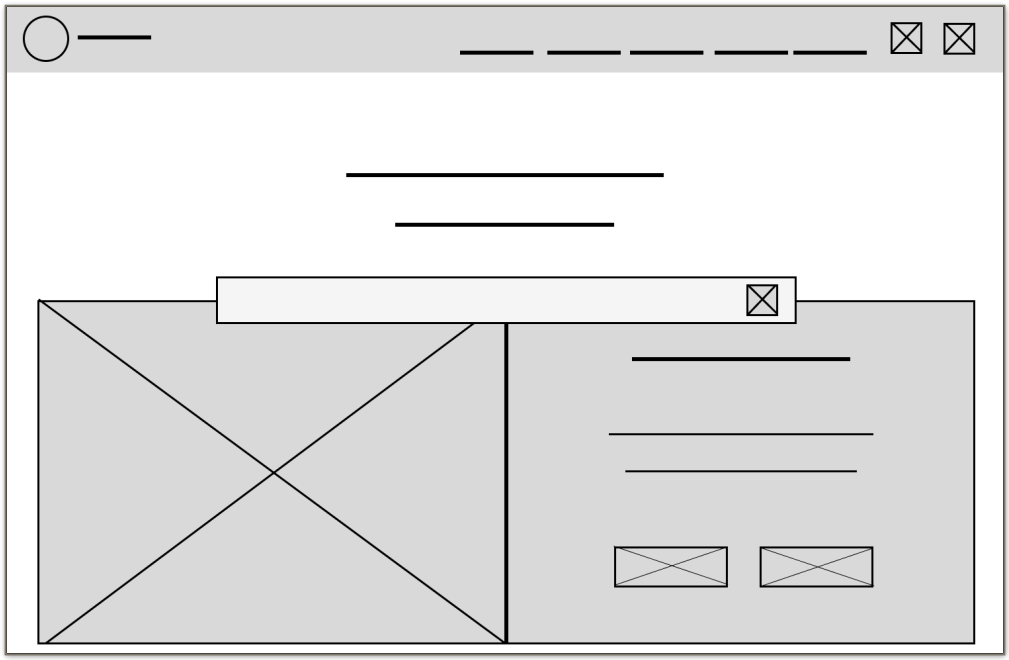


Figure 07: Wireframe for Landing Page

ii. Wireframe for Product Category

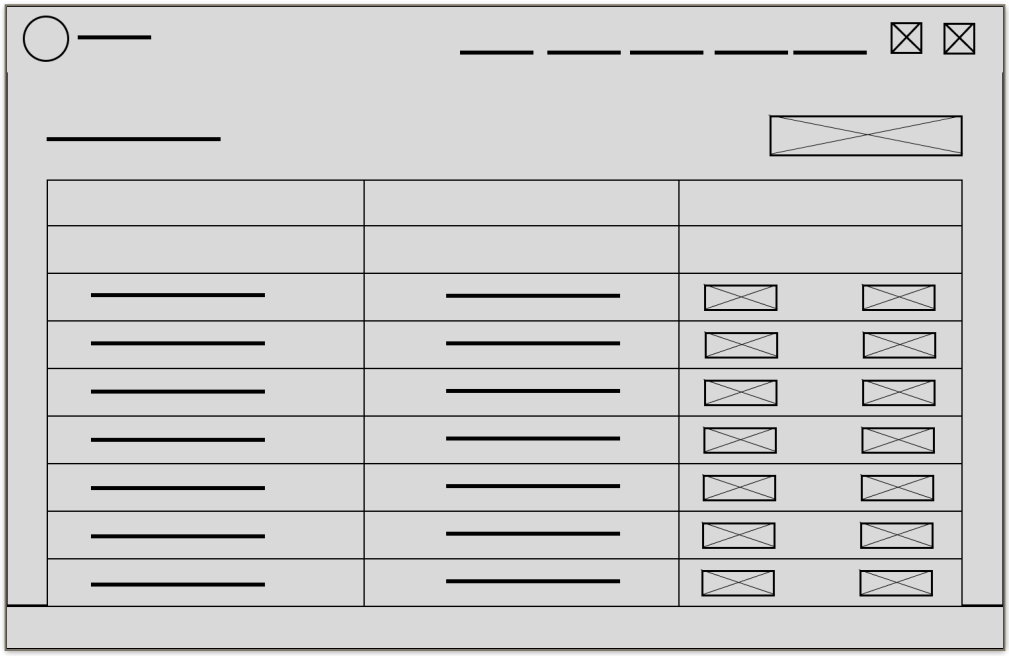


Figure 08: Wireframe for Product Category

iii. Wireframe for Login Screen

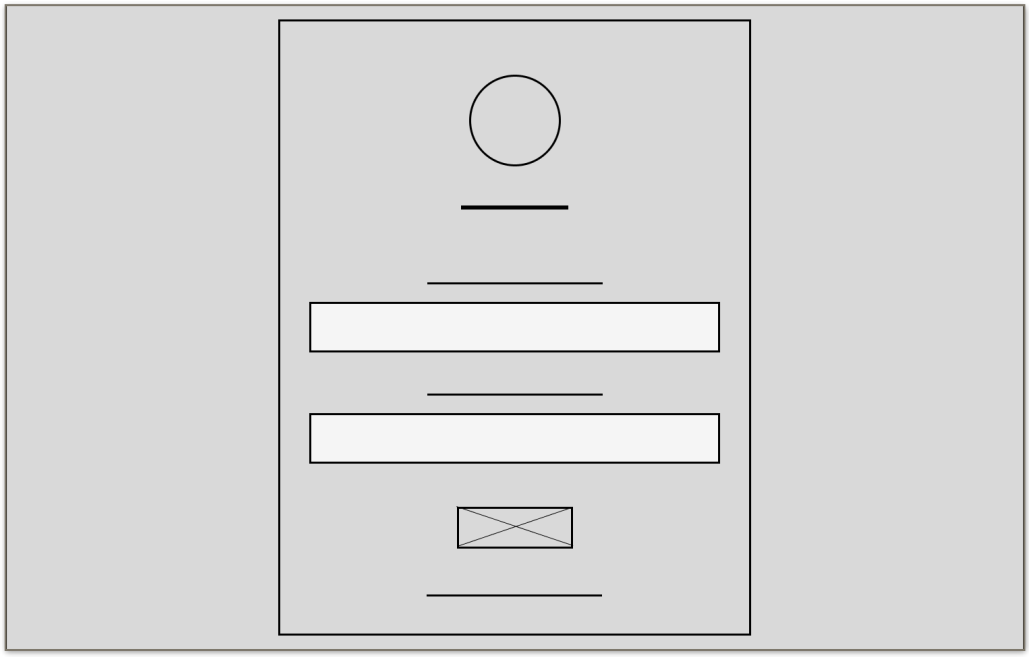


Figure 09: Wireframe for Product Category

iv. Wireframe for Signup Screen

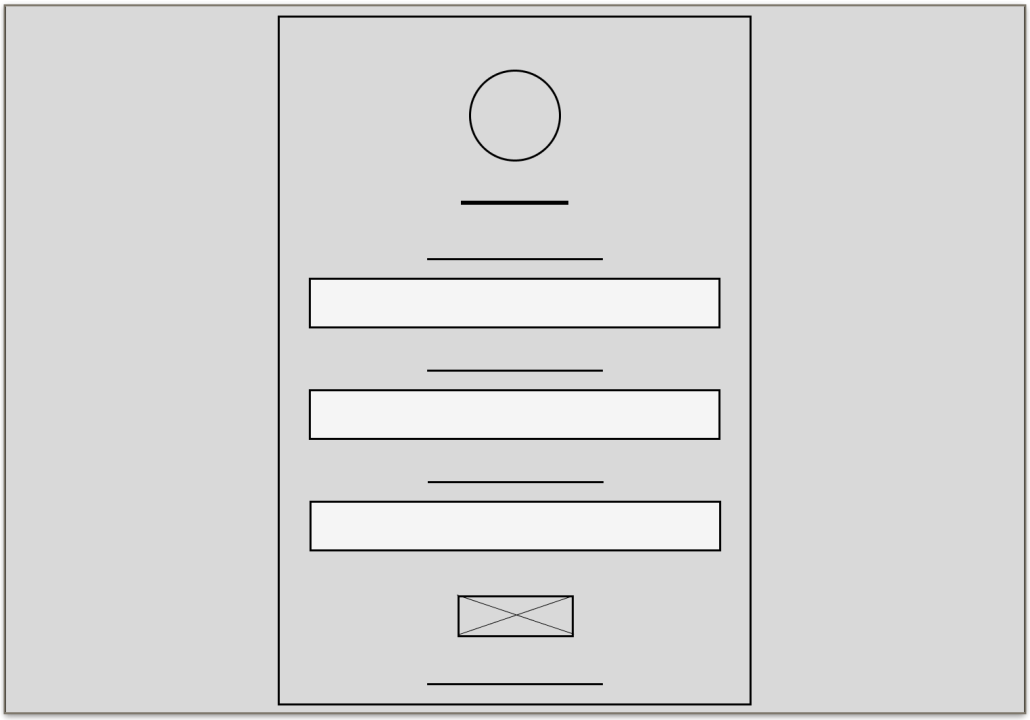


Figure 10: Wireframe for Product Category

v. Wireframes for Products and Brands

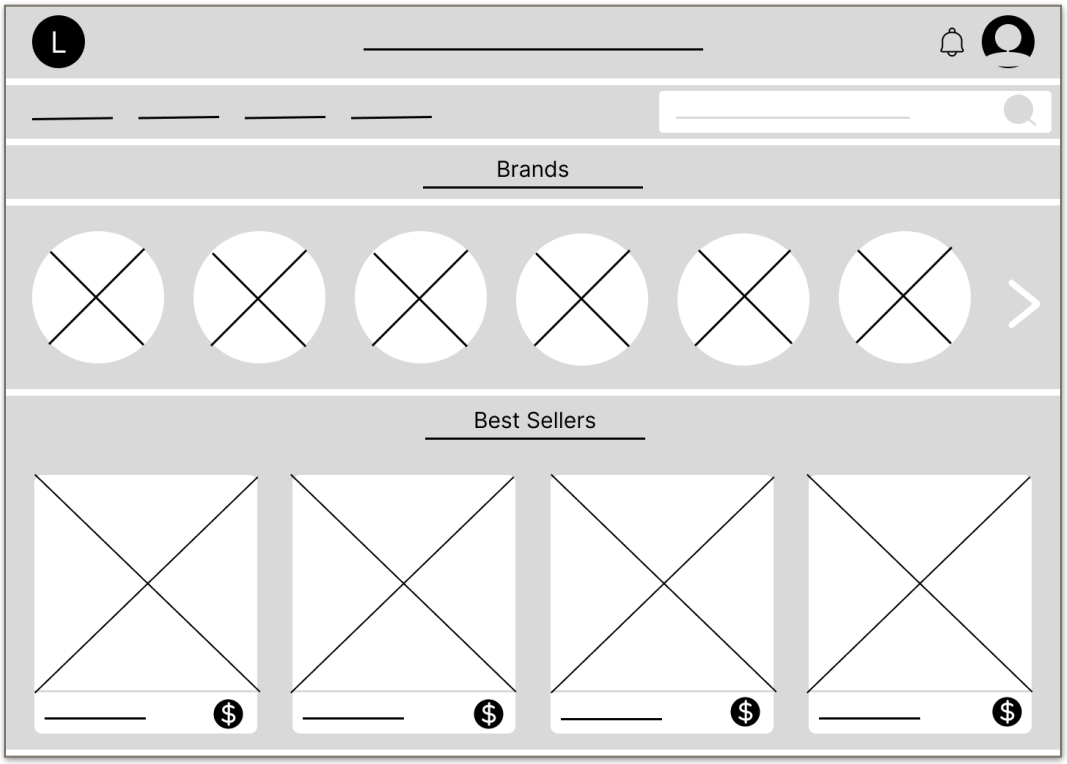


Figure 11: Wireframe for Products Display and Brands

vi. Wireframe for Cart

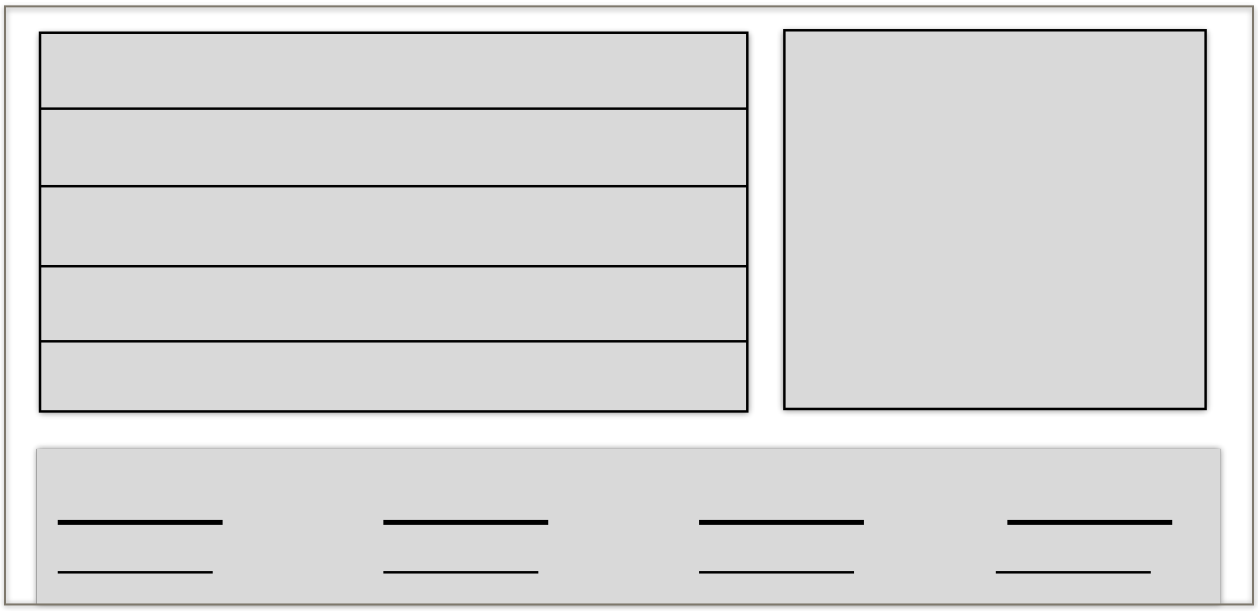


Figure 12: Wireframe for Cart Page

3.4. Interface Design and Webpage Descriptions

i. Product Category Page

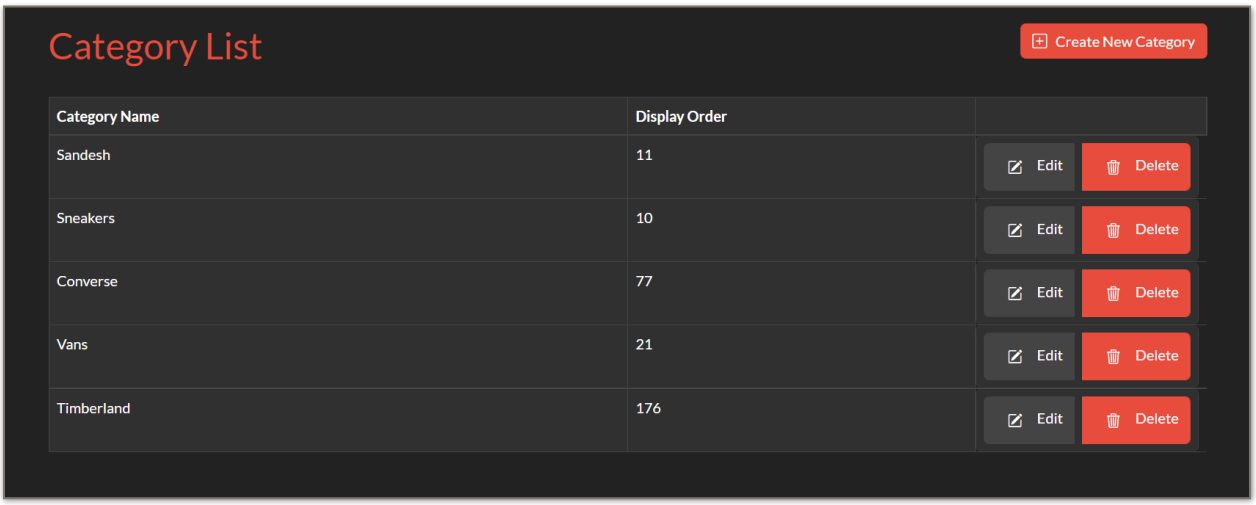


Figure 13: Product Category Page of web application

This webpage above is a 'Category Page' which displays all the available product categories within the website. The heading clearly shows the purpose of webpage, followed by a button on right hand side which permits admin to create new category. Below that is a table that comprise of three columns. The first column lists name of categories, second column shows the quantity of orders and the third column consist of two buttons: edit and delete. 'Edit' and 'Delete' buttons are denoted with text as well as icons that most likely represent their respective actions.

Design Decisions & Reasons

- The 'Create New Category' is kept separately so that it is easy for users to recognize it instantly while visiting the page.
- The table is bordered lightly so that it differentiates from background. Moreover, text color is set to white so that high contrast is maintained for strong readability feature.
- After all, user experience is the most important thing in design. Therefore, icons that denotes action are used (pencil-note icon for edit and trash icon for delete)

ii. Product Page and Details

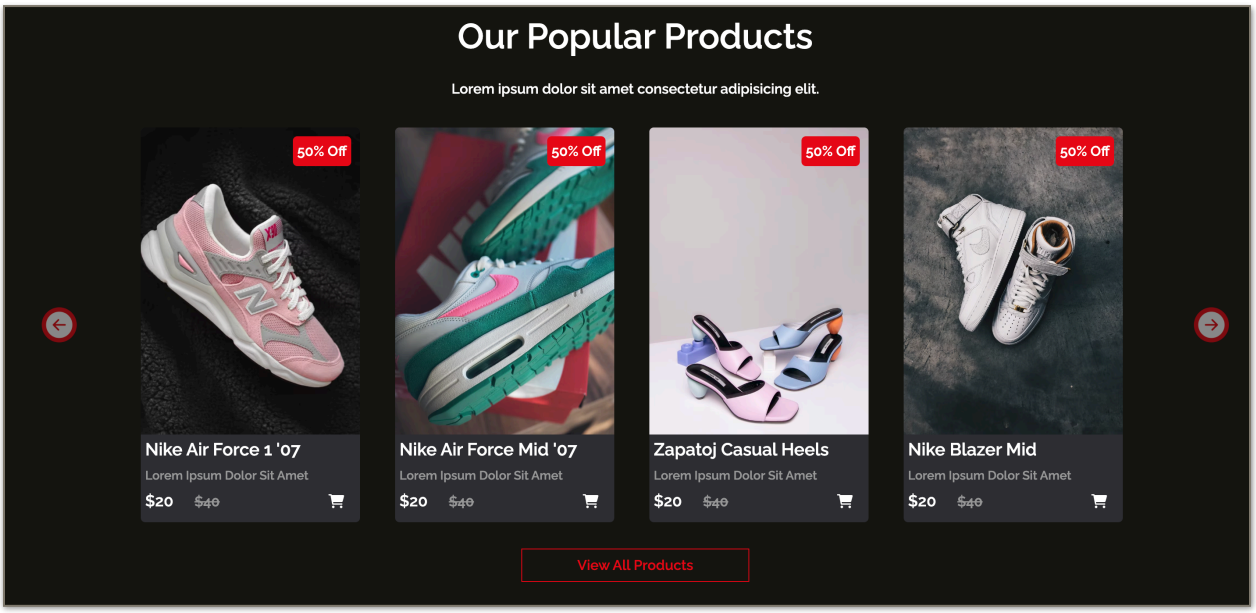


Figure 14: Product Page Interface of web application

The User Interface above is a 'Product Page' where website's product (i.e, shoes) are displayed in different formats. The section basically contains of product cards, which has further information about product details (i.e, name, price, etc). The interface is designed in such a way that users can browse more and more products as they want. Users can either use carousels or browse a product to view details. Talking about actions, an admin can perform all CRUD tasks with these products whereas customers can only make purchase and add to cart.

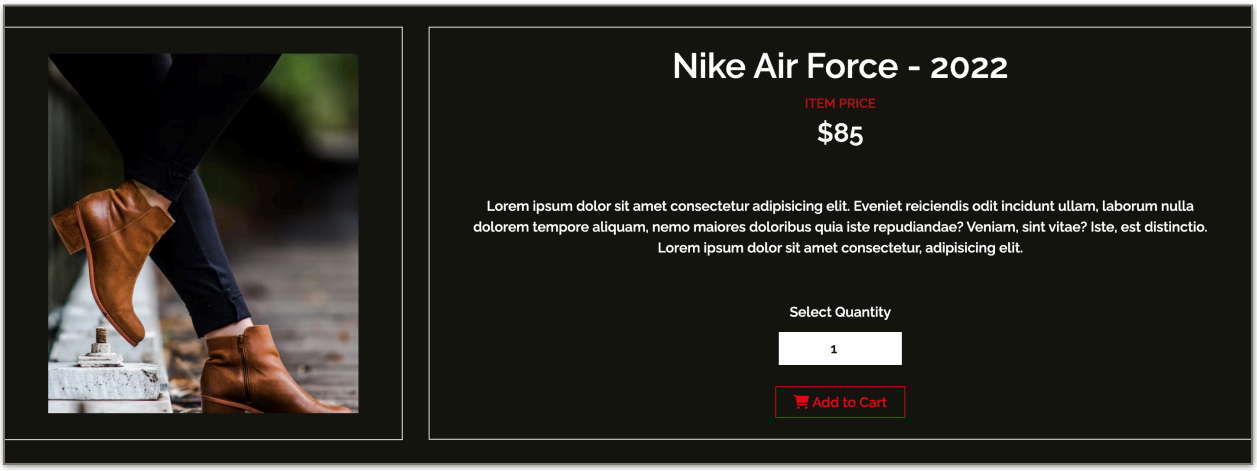


Figure 15: Product Details Interface of web application

Design Decisions & Reasons

```
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;

namespace MingoShoesWeb.Models
{
    16 references
    public class Category
    {
        [Key]
        3 references
        public int Id { get; set; }
        [Required]
        9 references
        public string Name { get; set; }
        [DisplayName("Display Order")]
        [Range(1, 1000, ErrorMessage = "Display Order must be between 1 and 1000 only!")]
        9 references
        public int DisplayOrder { get; set; }
        0 references
        public DateTime CreateDateTime { get; set; } = DateTime.Now;
    }
}
```

- Clear and visually attractive cards to make user design minimal and understandable
- Consistency in color like other web pages
- Clear buttons representing their unique purposes
- Distinctive chips used to visualize important messages

iii. Login and Register Webpages

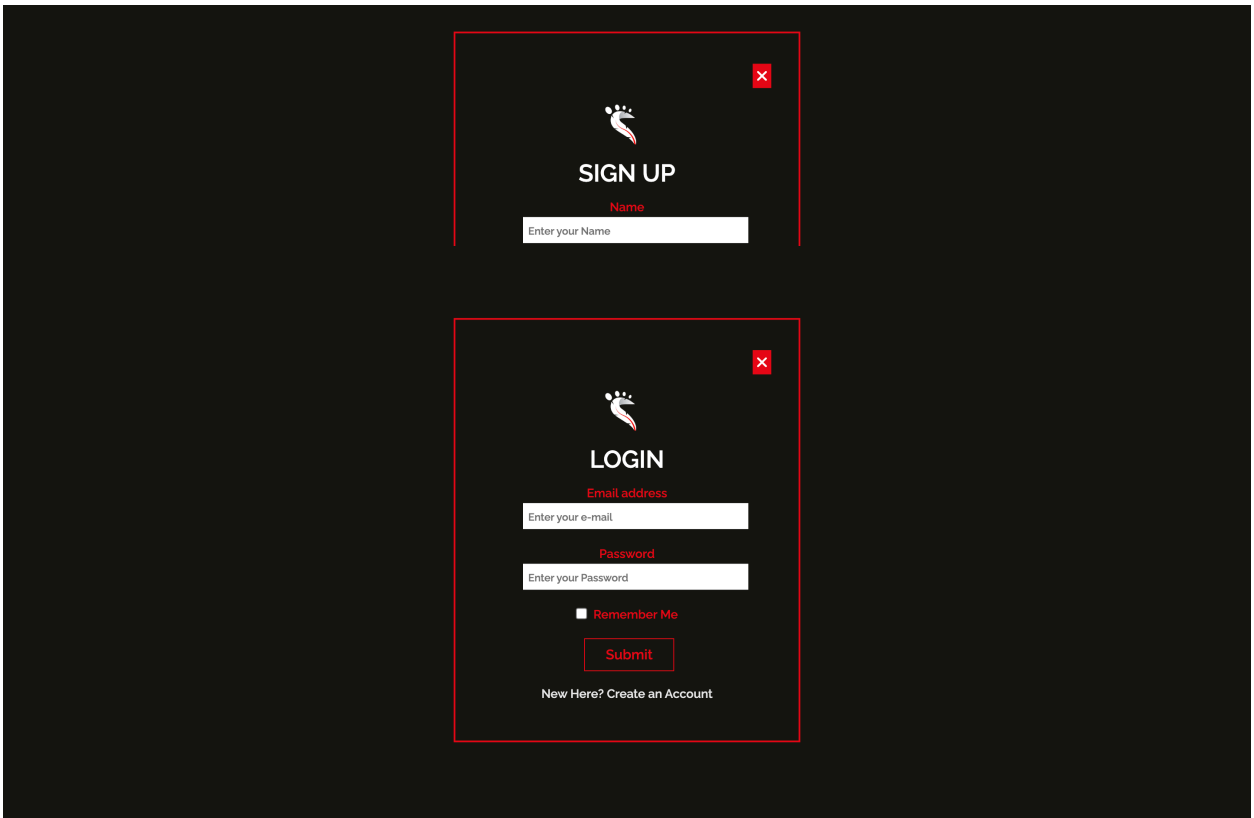


Figure 16: Register and Login Interface of web application

3.5. Navigational Architecture

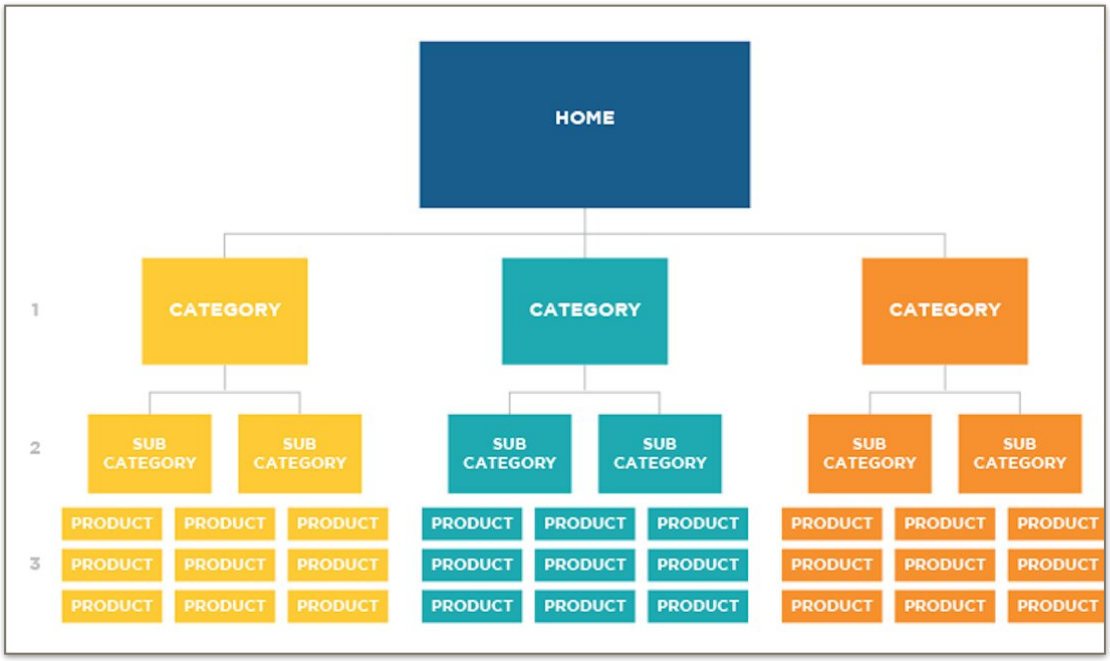


Figure 17: Hierarchy of Navigational Structure

4. Implementation

4.1. Factors related to design of user interface for real-world

There are several factors which were considered during the User Interface design process. The process began through problem statements and personas, followed by wireframes. Then, the system was tested with colleagues to find usability and several changes were made during that period. Those usability readings are shown in appendices section at the end of this report. The goal was to keep user front and centered so that there are less pinpoints and users can easily navigate through the website. Some of the major factors considered were :

- Human-Oriented Design
- Goal-Focused Design
- Responsiveness

4.3. Implementation of the system

Since we are using Model View Controller architecture, we need to create model. Whatever table we have in database, we need a corresponding code for that. Several packages are downloaded throughout the development process. There are lot of packages required, such as Microsoft.EntityFramework.Core, Microsoft.EntityFramework.Tools and SQL server packages were installed from package manager in Visual Studio. Then, another vital thing is data base connection where we need to establish database connection between Visual Studio and SQL Server Management Studio as shown in the diagram below :

Architecture and Framework

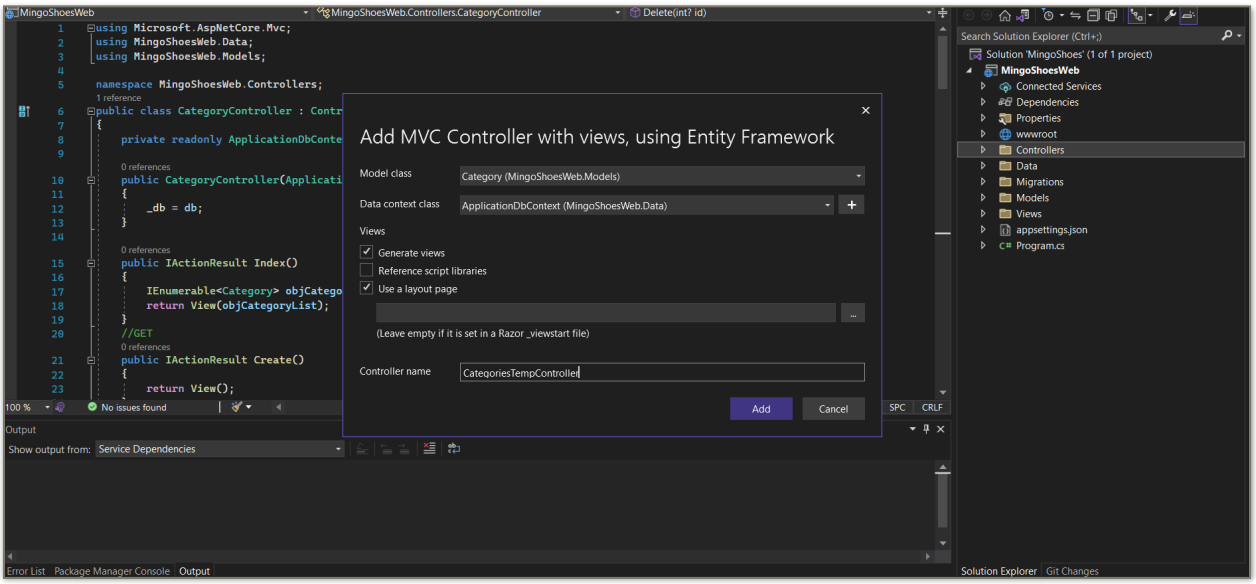


Figure 18: Adding MVC using Entity Framework

Moving on, there were places where we needed to play with keys and validations. Therefore, while coding, there are usage of key and data annotations. For example, [Key] is used as Primary Key whereas [Required] is used to make a property mandatory. Similarly, form validation is also done which is one of the major requirement of project. Different sorts of validations such as

RangeValidations are used so that user has clear ideas and there are no errors while navigating throughout the application.

Keys and Form Validation

Figure 19: Use of Keys and Validation

Demonstration of Form Validation

Figure 20: Implementation of form validation in UI

As seen on the User Interface above, the system provides instant feedback to users during any sort of exceptions. In our case, the name text-field is empty but we made it 'Required' so user cannot leave it blank. Similarly, we used a range in delivery orders where users can only enter amount between 1 and 1000. If someone enters invalid number, they will be notified and asked to enter a valid figure.

CRUD Operations

Create Operation

Figure 21: Implementation of Create (GET & POST)

Figure 22: Demonstration of Create Operation

Edit Operation

```
40
41      //GET
        0 references
42      public IActionResult Edit(int? id)
43      {
44          if (id == null || id == 0)
45          {
46              return NotFound();
47          }
48          var categoryFromDb = _db.Categories.Find(id);
49          if (categoryFromDb == null)
50          {
51              return NotFound();
52          }
53
54          return View(categoryFromDb);
55      }
56
57      //POST
58      [HttpPost]
59      [ValidateAntiForgeryToken]
        0 references
60      public IActionResult Edit(Category obj)
61      {
62          if (ModelState.IsValid)
63          {
64              _db.Categories.Update(obj);
65              _db.SaveChanges();
66              TempData["success"] = "Category Updated Successfully";
67              return RedirectToAction("Index");
68          }
69          return View();
70      }
71
```

Figure 23: Implementation of Edit (GET & POST)

Figure 24: Demonstration of Edit/Update Operation

Delete Operation

```
//GET
0 references
public IActionResult Delete(int? id)
{
    if (id == null || id == 0)
    {
        return NotFound();
    }
    var categoryFromDb = _db.Categories.Find(id);
    if (categoryFromDb == null)
    {
        return NotFound();
    }

    return View(categoryFromDb);
}

//POST
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public IActionResult DeletePOST(int? id)
{
    var obj = _db.Categories.Find(id);
    if (obj == null)
    {
        return NotFound();
    }

    _db.Categories.Remove(obj);
    _db.SaveChanges();
    TempData["success"] = "Category Deleted Successfully";
    return RedirectToAction("Index");
```

Figure 25: Implementation of Edit (GET & POST)

User Authentication

```
public string login(string username, string password)
{
    SqlDataAdapter da = new SqlDataAdapter("select * from [user] where username='" +username + "'
      and password = '" +password + "'", con);

    DataTable dt = new DataTable();
    da.Fill(dt);
    if(dt.Rows.Count>0)
    {


        return "valid username and password";


    }
    else
    {
        return "invalid";
    }

  //  return "invalid";

}
```

Figure 26: Implementation of Login Code

Implementation of Layout

```
<form method="post">
    <div class="border p-3 mt-4">
        <div class="row pb-2">
            <h2 class="text-danger">
                Create Category
            </h2>
            <hr />
        </div>
        <div asp-validation-summary="All" class="text-danger">
        </div>
        <div class="mb-3">
            <label asp-for="Name">
            </label>
            <input asp-for="Name" class="form-control" />
            <span asp-validation-for="Name" class="text-danger"></span>
        </div>
        <div class="mb-3">
            <label asp-for="DisplayOrder">
            </label>
            <input asp-for="DisplayOrder" class="form-control" />
            <span asp-validation-for="DisplayOrder" class="text-danger"></span>
        </div>
        <button type="submit" class="btn btn-danger" style="width:150px">
            Create
        </button>
        <a asp-controller="Category" asp-action="Index" class="btn btn-dark text-light border-danger" style="width:150px">
            Back to List
        </a>
    </div>
</form>

@section Scripts{
@{
    <partial name="_ValidationScriptsPartial" />
}
```

Figure 27: Implementation of Layout for Category

Implementation of Themes and Cdn

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - MingoShoesWeb</title>
    <link rel="stylesheet" href="~/css/bootSwatchTheme.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/MingoShoesWeb.styles.css" asp-append-version="true" />
    <link rel="stylesheet" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.css" />
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.9.1/font/bootstrap-icons.css">
</head>
```

Figure 28: Implementation of Bootstrap themes and Toastr cdn

```
@if(TempData["success"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js">
    </script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>
    <script type="text/javascript">toastr.success('@TempData["success"]')</script>
}
@if(TempData["error"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js">
    </script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>
    <script type="text/javascript">toastr.success('@TempData["error"]')</script>
```

Figure 29: Implementation of Toastr for attractive notifications popups

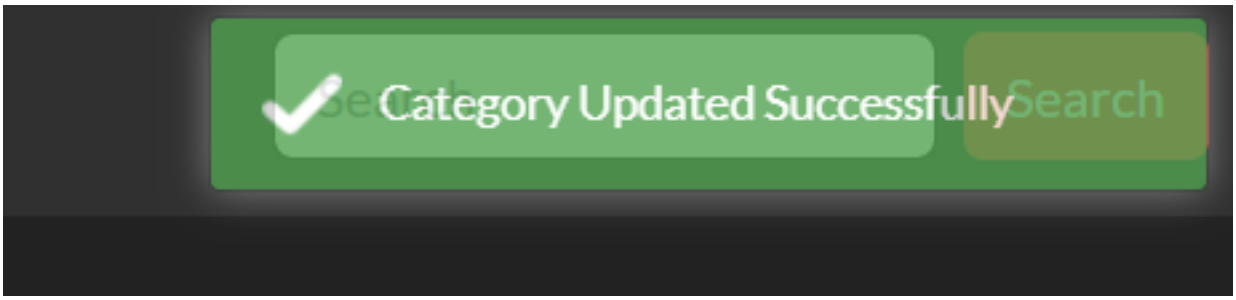


Figure 30: Demonstration of notifications popup using Toastr

5. Conclusion

To conclude, web-based application is a program that uses a web browser's interface to connect users to a distant server. We were provided with the task to create a functional, dynamic and interactive web-based application using .NET technology. Considering given requirements, Mingo Shoes is developed which is an online-retail website that serves people with varieties of footwear products and accessories. The project is done using advanced technologies in both front and back-end. HTML, Bootstrap and other UI frameworks are used for front-end whereas MVC architecture alongside entity framework is used for our project with ASP.NET Core. There are a lot of techniques used such as form validations, user guidance, data annotations, authentication, CRUD operations and many more. A user can have a e-commerce experience footwear shopping with plethora of features available within it.

References

Bidgoli, H. (2003). Encyclopedia of Information Systems: E-J. Amsterdam University Press.

Bourne, K. C., & Brown, C. (2013, October 15). Application Administrators Handbook: Installing, Updating and Troubleshooting Software [English]. In Design (1st ed.). Morgan Kaufmann. https://doi.org/10.1016/C2012-0-01093-9

Repository pattern (2018) *Scribd*. Scribd. Available at: https://www.scribd.com/document/299298662/Repository-Pattern (Accessed: October 21, 2022).

Sutcliffe, A. (2016). Designing for User Experience and Engagement. *Why Engagement Matters*, 105–126. https://doi.org/10.1007/978-3-319-27446-1_5

The technical guy (no date) *The Technical Guy*. Available at: https://www.halvorsen.blog/ (Accessed: October 10, 2022).

Troyer, O. D., Casteleyn, S., & Plessers, P. (2008). WSDM: Web Semantics Design Method. *Web Engineering: Modelling and Implementing Web Applications*, 303–351. https://doi.org/10.1007/978-1-84628-923-1_11

Wang, S., Tanaka, K., Zhou, S., Ling, T. W., Guan, J., Yang, D., Grandi, F., Mangina, E., Song, I. Y., & Mayr, H. C. (2005). Conceptual Modeling for Advanced Application Domains: ER 2004

Workshops CoMoGIS, CoMWIM, ECDM, CoMoA, DGOV, and eCOMO, Shanghai, China, November 8-12, 2004. Proceedings. Springer Publishing.

Whitaker, R. (2020) “Web content accessibility guidelines for Mobile,” *Developing Inclusive Mobile Apps*, pp. 51–66. Available at: https://doi.org/10.1007/978-1-4842-5814-9_3.

Appendices

A. Formative Evaluation

Internal formative evaluation was carried for web application interfaces with iterative design methodology. With this, interface were refined after testing and revision so that it puts user front and centered to make user experience better.

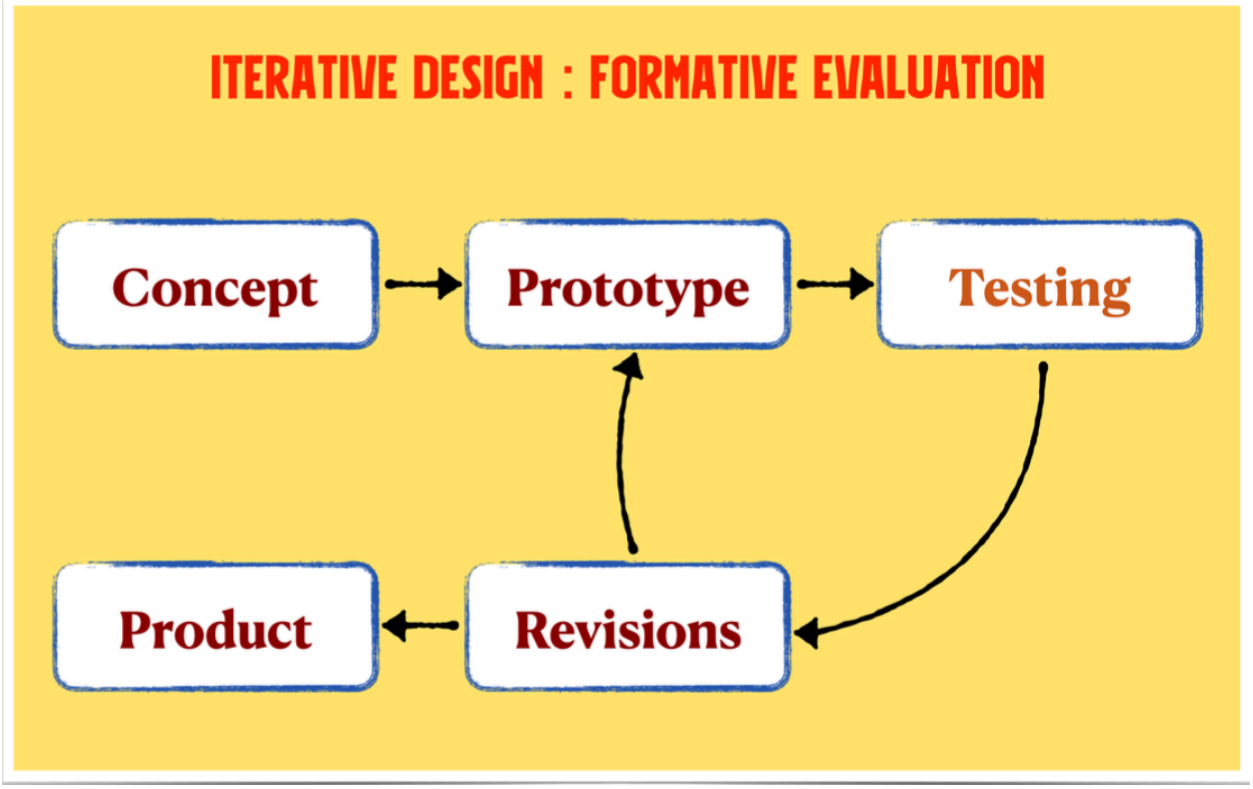


Figure 31: Iterative Design Process for system design (Made in Figma)

B. Literature Reviews

In order to get detailed knowledge about technology, design patterns and the way to work with those technologies, a detailed literature review is done. Throughout this, a large number of research articles, journals, reviews and research papers were studied. This helped to understand how and why, front-end and backend technologies were introduced in web education. Few of the articles that were studied is demonstrated below :

Defining the mobile web app experience

As mentioned earlier, this guide focuses on building mobile web apps. In this section we will examine the defining characteristics of a modern mobile web app. These characteristics are born of best practices and provide a useful framework upon which you may plan and design the features of your own app.

Mobile web apps should be

- Lightweight and responsive
- Designed to suit each device's capabilities and constraints
- Include a rich, platform-agnostic user interface
- Built with forward-thinking practices

Lightweight and responsive

Mobile devices may be more powerful than the computers we owned in 1995, but they remain quite constrained compared to the desktop computers we use today. A slower processor not only impacts the overall speed of the browser, but can also influence the speed at which content is accessed from the network, the redraw rate for effects and animations, and the responsiveness of the view as a user interacts with it. Mobile devices are also often used in contexts where bandwidth may be poor or prone to unexpected latency.

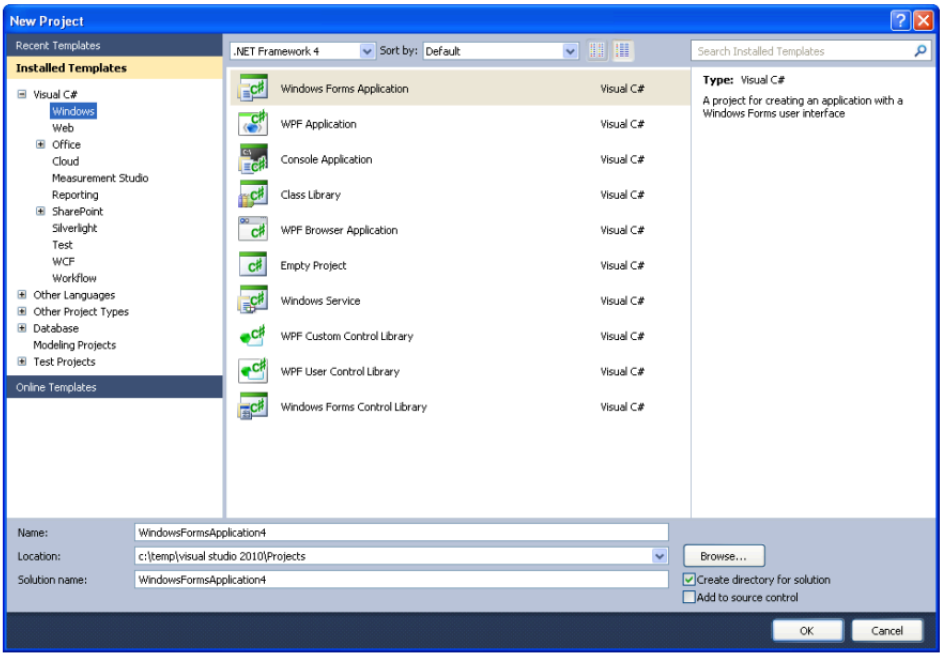
Mobile apps should therefore be lightweight and not impose additional latency through unnecessarily heavy markup, poor data management, or use of gratuitous and unnecessary effects. A good way to determine an appropriate size for your app is to consider how long you would like users to wait as the page loads.

The table below illustrates average wait times for a 1MB web page moving at an average data rate on various network types. **These times are average and do not account for network latency and the time it will take for the browser to render the content once it's downloaded. (1)**

14.4Kbps	568 seconds (~10 minutes)	Typical of a 2G connection
77.5Kbps	105 seconds (~2 minutes)	3G connection
550Kbps	15 seconds	4G connection

Recent statistics indicate (2) that a size of 1MB or greater is now quite common on the desktop. And while 3G is now widely available in many developed economies (reaching over 56% in the US), global 3G

Figure 32 : Modern Web Apps Article (Whitaker, 2020)



1.2 C#

C# is pronounced “see sharp”. C# is an object-oriented programming language and part of the .NET family from Microsoft. C# is very similar to C++ and Java. C# is developed by Microsoft and works only on the Windows platform.

1.3 .NET Framework

The .NET Framework (pronounced “dot net”) is a software framework that runs primarily on Microsoft Windows. It includes a large library and supports several programming languages which allow language interoperability (each language can use code written in other languages). The .NET library is available to all the programming languages that .NET supports. Programs written for the .NET Framework execute in a software environment, known as the Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework.

1.4 Object-Oriented Programming (OOP)

Figure 33 : .NET and web programming (The technical guy)

Repository and Unit of Work Patterns with Entity Framework in ASP.NET MVC

Repository Pattern: A Repository is just a class of all kind of possible operations that can be performed on an entity/object. May be the operations are CRUD (Create, Retrieve, Update and Delete). Repository pattern based application is that where all the entities are based on their respective repository class (generic/non-generic). This kind of class can be non-generic in a sense that every entity will have a separate class with the set of operations whereas it can be generic in a sense that every entity will have a common class which is designed in a special way hence offering a single access point of repository.

Repository Pattern consists of database entity models. Repository pattern ensures SOC (Separation of Concern) in a sense that the application need not know about the data source.

Following coding steps (1-5) can be realized to implement Repository and Unit of Work Patterns with Entity Framework in ASP.NET MVC.

Step 1: Create an interface so that other generic repository class can implement it and hence ensuring multiple inheritance

```
public interface IGenericRepository<TEntity>
   {
      IEnumerable<TEntity> Get();
      TEntity GetByID(object id);
      void Insert(TEntity entity);
      void Delete(object id);
      void Delete(TEntity entityToDelete);
      void Update(TEntity entityToUpdate);
   }
```

Step 2: Create a generic repository class for the entities so that it can be imagined as if it was created for a single entity and hence reducing code duplicity

```
public class GenericRepository<TEntity> : IGenericRepository<TEntity> where TEntity : class
   {
      internal MvcAdvancedEntities context;
```

Figure 34 : Study of Repository Pattern (Repository pattern 2018)

C. Usability Study

This specific paper comprise of all the instructions to be followed by users throughout this test. We would be grateful if all participants express their real feedbacks (both positives and negatives) once this test is performed.

Task 1 : System Inspection

This is the first step of usability testing where users will simply open the app and use up some time on interfaces. This is to observe the first and instant response of users while using the application.

Task Steps :

- Open the application
- Simply inspect the opening interfaces in overall

Task 2 : User Registration

Considering this to be your first time using the application, you need to sign up to use system features.

Task Steps :

- Click the 'Create an Account' option
- Enter your details (Name, Phone Number, Email and Password)
- Click 'Sign Up' button
- Click 'Confirm' to ensure all your details

Task 3 : Accessing the Home Page

Once you create an account, you can get access to all available features. Now, you will explore the 'Home Page' in particular.

Task Steps :

- Examine the design (color, order and typography)
- Notice whether 'Weekly Offer' is distinct and captivating
- Chick the heart icon and add foods to favorites list

Task 4 : Using Search Bar and Shortcuts

At the top of home page, there is a search bar indicated with a magnifying glass. Examine how convenient and facile it is.

Task Steps :

- Click on the search bar and see whether it recommends any food items
- Use shortcuts to find foods more easily
- Observe whether the search list provides enough information

Task 5 : Explore the dashboard

When you click the ham-burger icon on the top-left of home page, you can see several features. Try checking each of them.

Task Steps :

- Check FAQs and see if they are useful
- Click on social media buttons
- Click on 'Daily Offers'
- Click on 'Feedbacks' and post anything
- Click on 'Favorites' to see selected foods

Task 6 : Order Baskets

The 'Order Basket' is where you can see what you have ordered.

Task Steps :

- Check order baskets to see the list of your ordered foods
- Check their status (whether it is pending or accepted)

Task 7 : Profile Page

By clicking on the profile icon, you can make necessary changes about yourself.

Task Steps :

- View account details to see your information
- Click on 'Forgot Password?' to request password change
- Click on 'Update Location' to change your current location

Task 8 : Signing Out

The sign out option will take user out of the application and clear user details. This will keep your account secured from others.

Task Steps :

- At the top-right of 'Profile' page, click on the 'log out' icon
- Refresh and check whether you are actually logged out

Task 9 : Log In using credentials

At this point, you are already a registered user. So, to conclude the test, use your updated credentials and see if you can log in to the application.

Task Steps :

- Enter your credentials (Email/Phone number and password)
- Click on 'Log in' button

Thank You !

Outcomes

Task Rating

Examiners	T. 1	T. 2	T. 3	T. 4	T. 5	T. 6	T. 7	T. 8	T. 9
No. 1	✓	✓	✓	✓	✓	✓	✓	✓	✓
No. 2	✓	✓	✓	✓			✓	✓	
No. 3	✓		✓	✓		✓	✓	✓	✓
No. 4	✓	✓	✓	✓	✓	✓	✓	✓	✓
No. 5	✓	✓	✓	✓	✓	✓	✓	✓	✓
No. 6	✓	✓	✓			✓	✓	✓	✓
No. 7	✓	✓	✓	✓	✓	✓	✓	✓	✓
No. 8	✓	✓	✓	✓		✓	✓	✓	
No. 9	✓		✓	✓		✓	✓	✓	
No. 10	✓	✓	✓	✓	✓	✓	✓	✓	✓
No. Task	10	8	10	9	5	9	10	10	7
Rate (%)	100%	80%	100%	90%	50%	90%	100%	100%	70%

Note : T. = Task number

D. Additional Screenshots

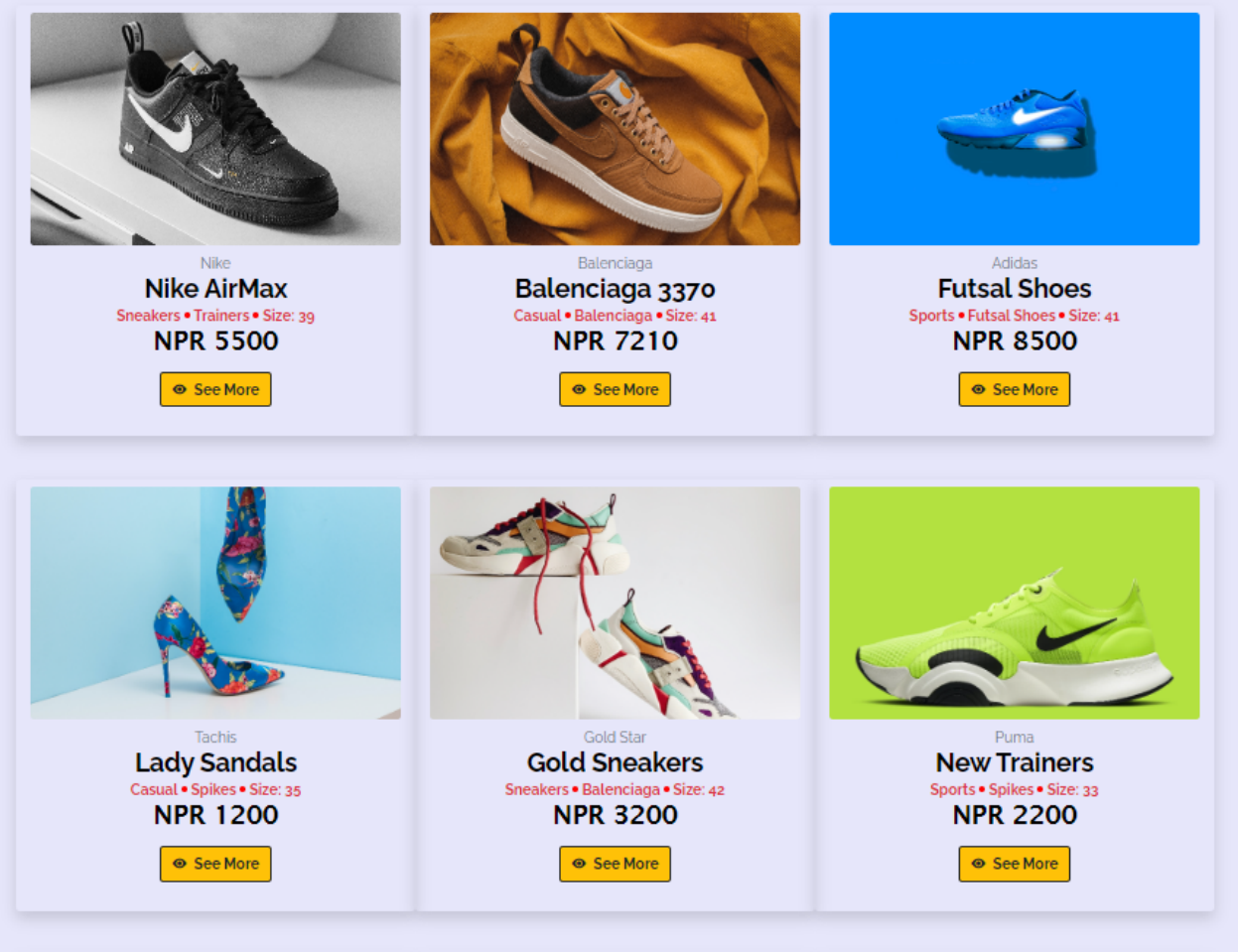


Figure 35 : Browsing available products

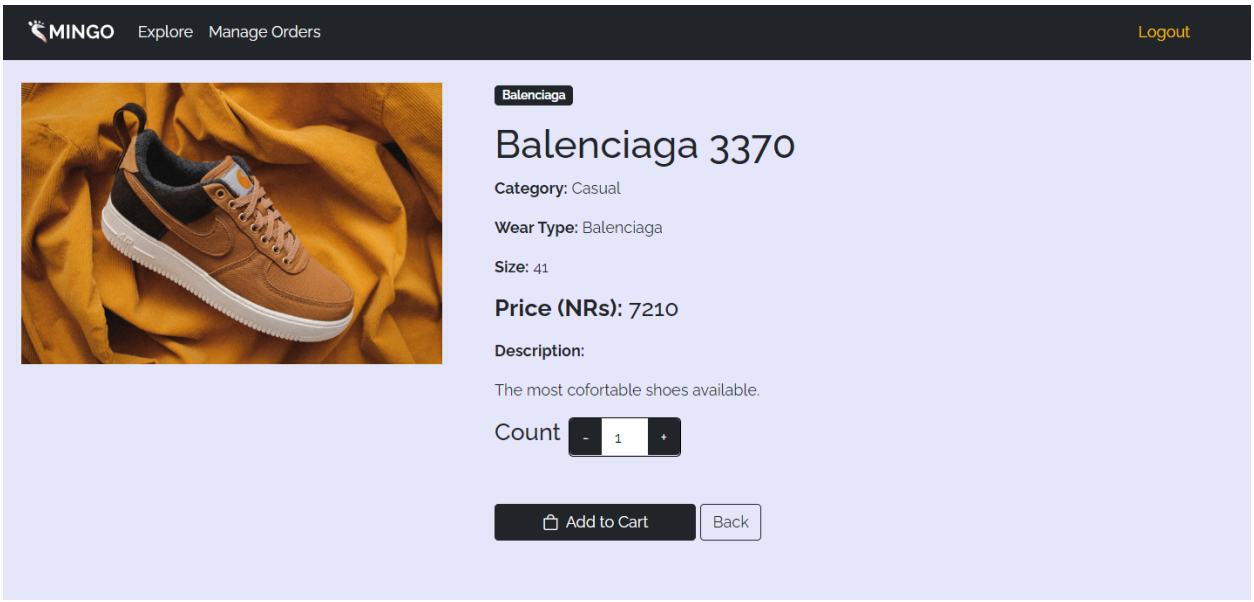


Figure 36 : Product Description Page

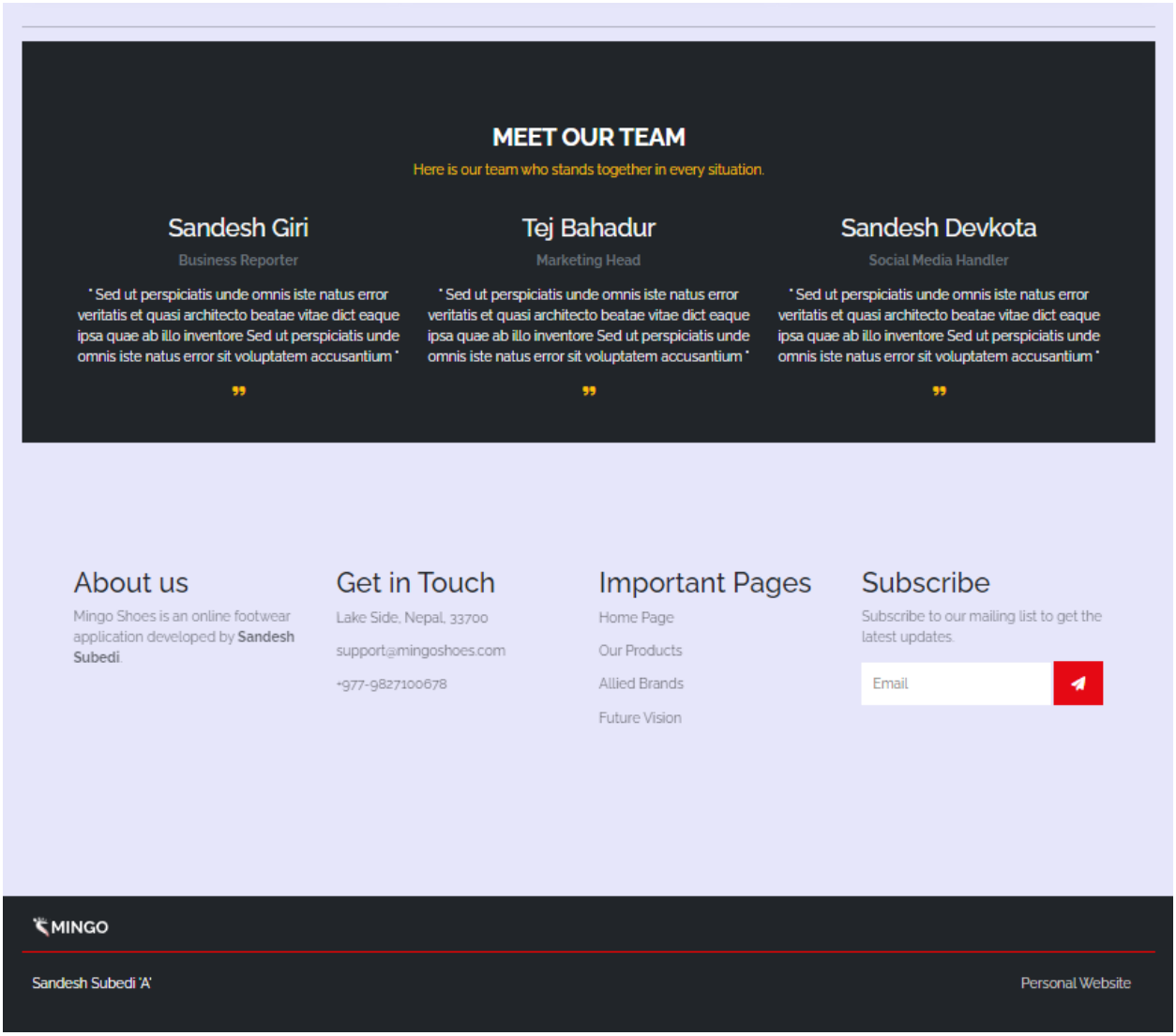


Figure 37 : Team testimonial and Footer Section

End of Documentation