Dashing Through the Snow

System Design Document

MCDA 5510(Software Development in Business Environment)

M.Sc. in Computing and Data Analytics

**Saint Mary’s University**

Halifax, NS B3H 3C3, Canada

Date – 26th Nov 2022

**Content**

[1 INTRODUCTION 4](#_Toc120510263)

[**1.1** **Purpose** 4](#_Toc120510264)

[**1.2** **Definitions, Acronyms and Abbreviations** 4](#_Toc120510265)

[**1.3** **Scope** 5](#_Toc120510266)

[**1.4** **References** 6](#_Toc120510267)

[**1.5** **Overview** 6](#_Toc120510268)

[2 SYSTEM OVERVIEW 6](#_Toc120510269)

[**2.1** **System Characteristics** 7](#_Toc120510270)

[**2.2** **System Architecture** 8](#_Toc120510271)

[**2.3** **Infrastructure Services** 10](#_Toc120510272)

[**2.4** **System Context** 11](#_Toc120510273)

[**2.5** **Standards and conventions** 11](#_Toc120510274)

[**2.6** **Software development tools** 12](#_Toc120510275)

[3 COMPONENT DESCRIPTION 12](#_Toc120510276)

[**3.1** **UI/UX** 12](#_Toc120510277)

[**3.2** **Structural Components** 13](#_Toc120510278)

[**3.3** **Client Component** 13](#_Toc120510279)

[**3.4** **Server Component** 13](#_Toc120510280)

[4 APPENDIX 13](#_Toc120510281)

[**4.1** **Record of Changes** 13](#_Toc120510282)

[**4.2** **Approvals** 13](#_Toc120510283)

List of Figures

[Figure 1 - System Architecture 7](#_Toc120272438)

[Figure 2 - Context Diagram 9](#_Toc120272439)

List of Tables

[Table 1- Abbreviation 3](#_Toc120285466)

[Table 2- Development Tools 10](#_Toc120285467)

[Table 3- Record of Changes 11](#_Toc120285468)

[Table 4- Approvals 11](#_Toc120285469)

# INTRODUCTION

The System Design Document (SDD) describes how the functional and nonfunctional requirements are recorded. The SDD describes design goals and considerations, provides a high-level overview of the system architecture, and describes the data design associated with the system. The high-level system design is further decomposed into low-level detailed design specifications for each system component, including hardware, internal communications, software, system integrity controls, and external interfaces.

## **Purpose**

The purpose of this document is to describe the system design for the project “dashing through the Snow”. This document will explain the scope, architecture, characteristics, and the components of the application. The SD document is intended for the team members working on the application and the faculty who are the stakeholders of the project.

## **Definitions, Acronyms and Abbreviations**

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| FR | Functional Requirement |
| NFR | Non-Functional Requirement |
| IR | Interface Requirement |
| DB | Database |
| Frontend | Client-side User Interface |
| SD | System Design |
| Backend | Server |
| SQL | Structured Query Language |

Table 1- Abbreviation

## **Scope**

### **In Scope**

1. Landing Homepage
   1. Carousels
2. About Us
   1. Brief Overview
   2. Journey
   3. Contact Information
3. Shop
   1. Categories
   2. Search
   3. Filters
   4. Card view of Items
   5. Add items to the cart
4. Cart
   1. View items in cart
   2. Remove items from cart
   3. Checkout
5. Profile
   1. Edit profile
   2. Add/Remove addresses
   3. Order History
6. Login / Register
   1. Login
   2. Log in with Google or Facebook
   3. Register
   4. Register with Google or Facebook
7. Payment Gateway
   1. Card Payment
   2. Wallet Payment

### **Out of Scope**

1. Vendor Screen
2. Customer Support

## **References**

1. [Angular with .NET](https://learn.microsoft.com/en-us/visualstudio/javascript/tutorial-asp-net-core-with-angular?view=vs-2022)
2. [Payment Using Stripe](1.%09https:/www.positronx.io/how-to-integrate-stripe-card-checkout-payment-gateway-in-angular/)
3. [Web applications Azure architecture design](https://learn.microsoft.com/en-us/azure/architecture/guide/web/web-start-here)
4. [Hosting Web Application on Azure](https://jasonwatmore.com/post/2020/01/08/angular-net-core-sql-on-azure-how-to-deploy-a-full-stack-app-to-microsoft-azure)

## **Overview**

The web application is dynamic, responsive, and secure powered by Azure services. The application has all the basic features of a small e-commerce website.

# SYSTEM OVERVIEW

Our System is designed to have 4 layers integrate and interact with one another. The administration space encompasses management and documentation aspects. Our client layer holds all client related hardware which interacts with our server side via a rest api.

Our server-side layer is further divided into an API layer, application layer, and database layer. Our payment layer is a separate layer that deals with the payment gateway for our application and is further integrated with our generator services layer.

The main goal of the above system is to create a functional and efficient E-commerce System capable of enabling users to buy their own Christmas items from our inventory in an easy and user friendly manner.

## **System Characteristics**

The web application is expected to be available round-the-clock from anywhere in Halifax, and usable from any device for any screen size. Security is a must for end users’ data and should be flexible, and scalable to meet reasonable demand occasionally. Additionally, it should have rich user experiences built on the client.

### **Availability**

“Dashing Through the Snow” is planned to be hosted on Azure and will be available only in the city of Halifax.

### **Performance**

The delay in loading the web application should be less than 1 second.

### **Scalability**

The application is scaled to handle 20 users in real-time.

### **Security**

Security is a very important factor, and the application follows industry standards and guidelines.

* Password encryptions, namely SSL encryption is to be used.
* Azure Web Application Firewall is to be enabled.

### **Functional**

The website is designed to address selling local festive goods and items such as lights, decorations, cakes, candles, dresses and many more.

### **Ease of use**

The website is designed to be easy to use with friendly and quick navigation to move around different pages on the site.

### **Content**

Website content is relevant, informative and user-centric.

### **Optimization**

The website and its content are to be optimized for different devices, browsers, data speed, search engines, and users.

### **Responsive**

“Dashing Through the Snow” will be a responsive website. It should change its layout and options to fit the device and browser size.

## **System Architecture**

Diagram

Description automatically generated

Figure 1 - System Architecture

Chart, waterfall chart

Description automatically generated

Figure 2- Data Model

## **Infrastructure Services**

Application infrastructure consists of the components that are necessary for its function. The following components are implemented to support the application features and service delivery.

### **Azure App Service**

It will be used to deploy the application for any platform or device and deploy them on a scalable cloud infrastructure.

### **Azure Web Application Firewall**

Azure Web Application Firewall is a cloud-native service that will be used to protect the application from common web-hacking techniques such as SQL injection and security vulnerabilities such as cross-site scripting.

### **Azure Monitor**

Azure Monitor will be used for end-to-end observability of the application, infrastructure, and network.

### **SQL Server**

The application uses SQL Server as a backend to fulfill user requests, display product catalogues, and store user history (such as orders, addresses etc.).

## **System Context**

Diagram, PowerPoint

Description automatically generated

Figure 3 - Context Diagram

## **Standards and conventions**

* Logo Placement - Logos should be placed highest up on the page and 99% of the time, placed on the left side of the page.
* Main Navigation - Main navigation should be placed along of the top of the page, either to the right of the logo or below the logo.
* Content Hierarchy - Content should be made up of headings and paragraphs. Heading One should be used only once and the highest up on the page.
* Link Styling - Cross linking between pages and linking to other sources is a common practice but in order to differentiate link text from page text, use a different color for each.
* Buttons - The buttons should use adequate color contrast and proper sizing; users should recognize buttons when they see them.
* Colors – Use color with adequate contrast.
* Responsive – The website should adjust accordingly for different screens.

## **Software development tools**

|  |  |  |
| --- | --- | --- |
| **Software** | **Version** | **Purpose** |
| Node js | 16.17.0 | Server-side programming, websites and back-end API services |
| Angular | 15.0.0 | Building client applications using HTML and TypeScript |
| .NET | 6.0.0 | Build internet-connected application |
| MS SQL SERVER | 2019 Preview 3 | Manage data and store information |
| Microsoft Visual Studio | 17.3.6 | Edit, debug, and build code, and then publish an application |
| Visual Studio Code | 1.73.1 | Streamlined code editor with support for development operations like debugging, task running, and version control |
| Dbeaver Community | 22.3.0 | Integrated environment for managing any SQL infrastructure |

Table 2- Development Tools

# COMPONENT DESCRIPTION

The application has two different codes (sub-programs) running side-by-side.

* Client-side Code - The code that is in the browser and responds to some user input
* Server-side Code - The code that is on the server and responds to the HTTP requests

## **UI/UX**

This is the user-interface designed from sketches. It should have all the pages in scope with detailed information. These components have nothing to do with the operation of the web application architecture. Instead, they are part of the interface layout plan of a web app.

## **Structural Components**

The two major structural components of a web app are client and server sides.

## **Client Component**

The client component is developed in Angular (SCSS, HTML, and Typescript). As it exists within the user’s web browser, there is no need for operating system or device-related adjustments. The client component is a representation of a web application’s functionality that the end-user interacts with.

## **Server Component**

The server component is built using a combination of several programming languages and frameworks .Net, PLSQL, C#. The server component has two parts:

* App logic - The main control center of the web application.
* Database - All the persistent data is stored.

# APPENDIX

## **Record of Changes**

| Version | Date | Author | Description of Change |
| --- | --- | --- | --- |
| 1.0 | 26/11/2022 | Piyush Priyam  A00464893 | The document review was done by the team members and all the comments are incorporated. |

Table 3- Record of Changes

## **Approvals**

The undersigned acknowledge that they have reviewed the SD and agree with the information presented within this document. Changes to this SD will be coordinated with and approved by the undersigned, or their designated representatives.

|  |  |
| --- | --- |
| **Approved By** | **Date Approved** |
| Rashad Ahmed Imtiaz, A00466845 | 26/11/2022 |
| Shiney Prabhakar, A00466455 | 26/11/2022 |

Table 4- Approvals