Online Pharmacy App

This document details the requirement specifications for the above-named project. Reach out to your SME / Trainer for any query.

Technology	.Net FSD
Document Type	RLL Requirement Specifications
Author	MLA
Current Version	1.0
Current Version Date	28-Oct-2022
Status	Active

Document Control

Version	Change Date	Change Description	Changed By
1.0	28-Septermber-2021	Document Creation	Shrinivasan Shridharan
1.1	6-Nov-2022	Document Update	Shabarinath KP

Domain: Pharma

Project Objective:

Create a dynamic and responsive .Net web application which allows users to book medicines.

Tools and Technologies:

- **Front-End:** Html, CSS, Javascript, Angular and other front-end technologies
- **Server-side:** Asp.Net MVC / Asp.Net MVC Core
- **Back-end:** Sql Server
- **Data Access Layer:** Ado.Net / Ado.Net Entity Framework
- **Server:** IIS Express / IIS / Azure based deployment / Docker Container

Note: NO Auto Generated Code to be used in the design and development of the project. All Html, CSS, Javascripts, Server-Side code etc. have to be written manually.

Background of the project:

Contoso Ltd is a company which builds a software system catering to various business needs.

Contoso Ltd plans to develop " Online Pharmacy" - web application, where users can order pharmacy products online. The team decided to hire a Full Stack developer who could develop a web application with a rich and user-friendly interface. You are hired as the Full Stack .Net developer and are asked to develop the web application. The management team has provided you with the requirements and their business model so that you can easily arrange different components of the application.

Functional Requirements:

Below are the key responsibilities and functionalities to be implemented in the admin portal.

The admin user should be able to:

1. Dashboard
2. CRUD operations for Customer details
3. Display medicines in stock
4. Able to replenish the stock

Below are the key responsibilities and functionalities to be implemented in the user portal.

The user should be able to:

1. See list of medicines
2. User can place order for the medicine
3. Calculate the total price basis the medicines purchased once the user checkout of the system
4. First order will be eligible for discount and provide discounts to regular users
5. Display list of previous orders
6. Suggest alternative medicine if the searched product is not available
7. Order Cancellation
8. Download the report of medicines purchased

Common Features

1. Welcome Page
2. Login Screen with captcha
3. Registration Page with captcha
4. Forgot Password / Reset Password Page with captcha
5. All users should be able to View and edit profile.
6. Suggest Alternate Medicines incase the requested medicine is not available.
7. Implement validations wherever required for example: login page, registration page etc.
8. UI-UX should be user friendly.
9. Back button should be disabled after logout / sign-out

Phase 1: Database Schema Design

1. Identify domain objects and their attributes as per the requirement.
2. Create a Database tables with necessary relationship as per the requirement.

Phase 2: Front End Development

Develop web pages as per requirements for the web application using Front-End Technologies.

Phase 3: Back End Development

Develop a RESTful Web API to perform CRUD operations on Domain objects as per requirements using Asp.Net Web Api and Sql server database

Steps to develop a Restful Web API.

1. Identify the domain objects and their attributes as per requirements.
2. Design Database Schema as per requirements.
3. Create Entity class for each domain object with required attributes.
4. Create DAL class for performing CRUD operations for each Entity.
5. Create a Service class to invoke DAO class methods for each Entity.
6. Create a Controller class to build the RESTful Web API using Service class using required annotations.

Phase 4: Unit Testing

1. Perform Unit Testing using NUNIT / XUnit frameworks for all the functional requirements.
2. Perform Functional Testing using POSTMAN / Swagger for all REST end points.

Note:

- Use proper .Net Naming Conventions (package, class and interface, variable names)
- Use Interfaces for loose coupling as and when required
- Use standard HTTP status codes: 404,500,200,201,401,403 while implementing RESTful Web API