**Login and Registration System**

**Task Documentation**

**Overview**

This is a basic login and registration system implemented using Asp.Net core and Razor pages with SQL server for database. It allows user to register, login and view their information.

**Table of Contents**

1. Setup instructions

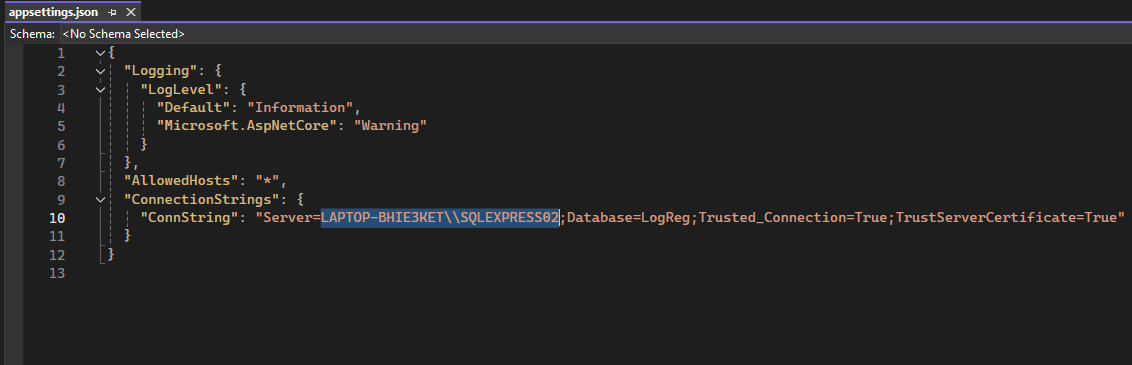
2. Technology Stack

3. Code structure and Design Decisions

4. Assumptions and Limitations

**Setup instructions**

*Database setup:*

1. Ensure SQL server database is running
2. Open appsetting.json and replace highlighted text in the screenshot bellow with name of database server on your local device.
3. Start packet manager console on Visual studio and run the following commands to migrate and setup database:

*Add-Migration name*

*Update-Database*

Now you can run the application IIS Express on Visual Studio.

**Technology Stack**

* **ASP.NET Core Razor Pages:**

Razor Pages is a page-based model for building server rendered web UI

* **Identity on ASP.NET Core:**

Is an API that supports user interface (UI) login functionality.

Manages users, passwords, profile data, roles, claims, tokens, email confirmation, and more.

* **Bootstrap for ccs framework**
* **SQL server for database.**
* **Entity Framework:**

Modern object-relation mapper that lets you build a clean, portable, and high-level data access layer with .NET (C#) across a variety of databases.

**Code structure and Design Decisions**

**Code structure:**

* **Program.cs:** Serves as the entry point and orchestrator of the ASP.NET Core Web API application. It sets up the web host, defines configurations, and links to the Startup class where the bulk of application configurations and services are configured.
* **appsettings.json**: JSON-formatted configuration file used to store application-specific settings configuration data. Connection string is setup in this file.
* **.cshtml** Pages: Razor pages. Very simple to use and hence best for a simple login/registration page.
* **.cshtml.cs** : This serves as the controller for the application.
* **ViewModels**: contain the data to display on the pages (views). Using a viewmodel to pass data to a view allows the view to take advantage of strong type checking

**Assumptions and Limitations**

* Password hashing used is built-in feature, so better hashing option maybe used.
* Only username can be used to login, email can’t.