HealthApp Project Report

This document serves as a presentation-style walkthrough and explanation of the HealthApp project structure, features, and technical decisions. The project is designed as a full-stack health management system built with ASP.NET Core and Razor Pages.

# Project Overview

HealthApp is a full-stack web application intended to manage user interactions between doctors and patients. The application allows user registration, login, and role-based logic using ASP.NET Identity.

# Project Structure

## HealthApp.Domain

This folder contains all the core models and business logic of the app. It defines models like ApplicationUser and interfaces for domain expectations. It promotes clean architecture, separation of concerns, and maintainability.

## HealthApp.Razor

The main front-end of the application built using Razor Pages. It handles the UI, routes, and uses ASP.NET Identity for authentication and authorization. The registration form includes role selection and should conditionally require a specialization field only for doctors (currently this validation needs fixing).

## HealthApp.API

A scaffolded backend API project meant to expose RESTful endpoints for future front-end frameworks like React or Angular. Not integrated yet, but structured for future implementation.

## HealthApp.Test

This folder contains setup for unit testing using xUnit. The goal was to test domain logic independently from the UI. Some tests are stubbed out but not fully implemented.

# User Secrets & Configuration

User Secrets is used to manage sensitive data like connection strings securely in development. The UserSecretsId is stored in the Razor project .csproj file and can be managed with the dotnet CLI.

# Tech Stack Summary

- ASP.NET Core Razor Pages  
- ASP.NET Identity  
- Entity Framework Core  
- SQL Server (LocalDB)  
- Bogus for dummy data generation  
- xUnit for unit testing  
- Clean layered architecture

# Known Issues / Work in Progress

- Specialization field validation needs to be role-specific (only for doctors)  
- API is not yet connected  
- Tests are partially implemented

# What's Working

- User registration and login  
- Role selection during registration  
- Clean, scalable folder and project structure

# Final Words

Although the project is not fully completed, the foundation is strong and follows clean architecture principles. With a bit more time, final touches like conditional logic, API integration, and testing can be added easily. This report serves to outline what was achieved and the direction the application is headed.