Socially Anxious Hub

Technical Document

Student: Virag Szabo (4727444)

Date: Summer of 2025

Subject: Threading in C#

School: NHL Stenden

NHL STENDEN hogeschool

Table of contents

Contents

1 Introduction

This technical document outlines the development plan for the **Socially Anxious Hub**, a C# .NET MAUI application designed to help users manage their social life through music and personal memories. Developed by Virag Szabo, this document provides a comprehensive overview of the project's components, requirements, functionalities, and development lifecycle.

2 Description

The **Socially Anxious Hub** aims to provide users with a calming and reassuring tool for self-expression. Leveraging modern multi-threading techniques like async/await and the capabilities of .NET MAUI, the application offers cross-platform compatibility for both Android and iOS devices. The core features include integrating with the Spotify API to create and manage personalized playlists and a digital memory board for users to store and revisit their cherished moments.

3 Components

The project consists of several key components:

3.1 Requirements

Components	Description
C# .NET	Utilize C# .NET 8 or higher for application development.
Asynchronous Programming	Implement asynchronous programming using the async/await pattern to ensure a responsive and non-blocking user interface.
.NET MAUI	Build the application using .NET MAUI for a unified codebase across Android and iOS.
Secure Storage	Use SecureStorage and other native encryption methods to protect sensitive user data.
Version Control	Implement version control with Git to track changes and manage the development process.

3.2 Functionalities

Components	Description
Spotify API Integration	Allow users to authenticate with Spotify to search for songs and manage playlists.
Playlist Management	Enable users to create, add, remove, and sort songs within their personalized playlists.
Memory Board	Provide a digital board for users to create and store memories, including titles, descriptions, and images.
Local Data Storage	Implement a local SQLite database for persistent storage of user playlists and memory board data.
Responsive UI	Design a clean and intuitive graphical user interface (GUI) that adapts to different screen sizes and orientations.

4 UML Diagrams

4.1 Description

Class	Description
SpotifyService	Responsible for handling user authentication (PKCE flow) and making API calls to Spotify.
DatabaseService	Manages all local data storage operations (CRUD) for the MemoryItem and Song objects using SQLite.
Song	A data model representing a song, including properties like Title, Artist, Album, and SpotifyUrl.
MemoryItem	A data model representing a memory, including properties for a title, description, and an image path.
MainViewModel	The central view model that manages the app's overall authentication state and navigation.

5 Features

The project consists of several key features:

Name	Version	Note
Visual Studio 2022	N/A	The official Integrated Development Environment (IDE) for the project.
.NET MAUI	N/A	The framework for building native cross-platform applications from a single C# codebase.
GitHub	N/A	The platform and version control system used for managing the codebase.
Asynchronous Programming	N/A	A core concept in C# for creating responsive applications by performing I/O operations without blocking the main thread.
SQLite	N/A	A lightweight, file-based database for local data persistence.

6 Development Lifecycle

6.1 Design

- **Database:** Plan the structure of a local SQLite database to store Song and MemoryItem data. This involves defining the tables and their columns.
- **UI/UX:** Create mock-ups for the user interface of the MainPage, PlaylistPage, and MemoryBoardPage.

6.1.2 Mock-ups

- Create mockups for the user interface (UI) of the application using ADOBE XD.
- Design UI elements such as buttons, forms, charts, and graphs to visualize social media analytics data.
- Incorporate feedback from stakeholders and potential users to refine the mockups.
- Ensure consistency in UI design across different screens and platforms (e.g., desktop, mobile).

6.2 Development

- **Environment Setup:** Configure the development environment with Visual Studio and the .NET MAUI SDK.
- **Backend Development:** Implement the SpotifyService for secure authentication and the DatabaseService for local data management.
- **Frontend Development:** Develop the user interface components using XAML and C# code-behind, leveraging the MVVM pattern.
- **Authentication:** Implement the Authorization Code with PKCE flow for secure Spotify integration.

6.3 Testing

- **Unit Testing:** Develop and execute unit tests for critical business logic within the SpotifyService and DatabaseService.
- **Integration Testing:** Verify the functionality of components working together, such as authenticating with Spotify and then fetching songs.
- **User Acceptance Testing:** Conduct testing to validate the overall user experience and find any bugs before release.
- **Performance Testing:** Use profiling tools to ensure the app is fast, responsive, and doesn't have memory leaks.

6.4 Presentation

- **Preparation:** Prepare a presentation summarizing project objectives, features, achievements, and challenges.
- **Delivery:** Showcase the project to stakeholders, including instructors and classmates.

7 Maintenance and Support

#	Title	Description
1	**Feature Enhancement**	Continuously improve and enhance the application by adding new features, such as video support for memories or a multi-user playlist feature.
2	**Performance Optimization**	Optimize the application's performance by refining algorithms or implementing caching mechanisms for better responsiveness.
3	**UI/UX Refinement**	Conduct user testing and iterate on the user interface (UI) and user experience (UX) to make the application more intuitive and user-friendly.
4	**Mobile Optimization**	Further optimize the application for mobile devices by implementing responsive design principles and enhancing touch interactions.
5	**Data Privacy and Security**	Strengthen the application's data privacy and security measures by implementing additional encryption protocols and access controls.
6	**Community Engagement**	Foster a community around the application by engaging with users and soliciting feedback.
7	**Portfolio Building**	Showcase the application as part of your portfolio to demonstrate skills, expertise, and achievements to potential employers.
8	**Continuous Learning**	Stay updated with the latest technologies, tools, and best practices in software development to further enhance my skills and knowledge.