

Technical Design Document

(CogniOpen - Nurturing Memory Wellness for Cognitive Impairment)

Submitted to:

University of Maryland Global Campus in partial fulfillment of the course Software Engineering Capstone (SWEN 670)

Submitted By: Team B

Fall Semester 2023

Version 1.0

October 5, 2023

Department of Software Engineering

University of Maryland Global Campus (UMGC)

Table of Contents

[Table of Contents 1](#_Toc149373328)

[Revision History 3](#_Toc149373329)

[1. Introduction 4](#_Toc149373330)

[1.1 Purpose 4](#_Toc149373331)

[1.2 Scope 5](#_Toc149373332)

[1.3 Objectives 5](#_Toc149373333)

[1.4 Project Documents 6](#_Toc149373334)

[1.5. Definitions, Acronyms, and Abbreviations 7](#_Toc149373335)

[1.6 References 9](#_Toc149373336)

[2. System Overview 9](#_Toc149373337)

[2.1. In Scope 9](#_Toc149373338)

[2.2. Out of Scope 9](#_Toc149373339)

[3. System Architecture 10](#_Toc149373340)

[3.1. Architectural Design 10](#_Toc149373341)

[3.2 Decomposition Description 11](#_Toc149373342)

[3.2.1 Database 11](#_Toc149373343)

[3.2.2 Transcription Service 12](#_Toc149373344)

[3.2.3 ChatGPT Service 13](#_Toc149373345)

[3.2.4 Rekognition Service 14](#_Toc149373346)

[3.3. Exception Handling 15](#_Toc149373347)

[4. Data Design 16](#_Toc149373348)

[4.1. Data Description 16](#_Toc149373349)

[4.2. Data Dictionary 18](#_Toc149373350)

[4.2.1 Users Table 19](#_Toc149373351)

[4.2.2 Audio Table 19](#_Toc149373352)

[4.2.3 Recognized Sounds Table 20](#_Toc149373353)

[4.2.4 Transcribed Texts Table 20](#_Toc149373354)

[4.2.5 Summarized Chats Table 21](#_Toc149373355)

[5.1. Class: homeScreen 21](#_Toc149373356)

[5.2. Class: assistantScreen 22](#_Toc149373357)

[5.3. Class: audioScreen 23](#_Toc149373358)

[5.4. Class: registrationScreen 24](#_Toc149373359)

[5.5. Class: loginScreen 24](#_Toc149373360)

[5.6. Class: profileScreen 25](#_Toc149373361)

[6. Application Interface Design 26](#_Toc149373362)

[6.1. User Interface Design 26](#_Toc149373363)

[6.2. Considerations for Designing the User Interface 26](#_Toc149373364)

[6.3 Screen Image 27](#_Toc149373365)

[6.3.1: User Login Credential Screen 28](#_Toc149373366)

[6.3.2: User Application Registration Screen 29](#_Toc149373367)

[6.3.3: Home Interface Screen 30](#_Toc149373368)

[6.3.4: Virtual Assistant Interface Screen 31](#_Toc149373369)

[6.3.5: Record Audio Interface Screens 33](#_Toc149373370)

[6.3.6: Search Interface Screen 35](#_Toc149373371)

[6.3.7: Previously Asked Questions Interface Screen 35](#_Toc149373372)

[6.3.8: Menu Interface Screen 36](#_Toc149373373)

[6.3.9: Profile Interface Screen 37](#_Toc149373374)

[6.3.10: Guided Tour Interface Screens 39](#_Toc149373375)

[7. Requirements Matrix 52](#_Toc149373376)

[8. Appendices 54](#_Toc149373377)

[8.1 Speech to Text service provider comparison 54](#_Toc149373378)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Description | Author | Date |
| 1.0 | Initial Draft | Team B | 09/03/2023 |
| 1.1 | Update document with image explanations | Zachary Cappella | 10/01/2023 |
|  |  |  |  |
|  |  |  |  |

# 1. Introduction

## 1.1 Purpose

The Technical Design Document (TDD) for the mobile application CogniOpen is a crucial resource for numerous stakeholders involved in the development process and has multiple purposes. The following section details primary goals for the document:

**Reference and Direction:** The development team is led by a clear and organized technical roadmap provided by the TDD throughout the project lifecycle. It is a point of reference for programmers, graphic designers, and other team members to comprehend the technical architecture, design tenets, and specific implementation details.

**Communication:** Effective project stakeholder communication is facilitated through the TDD, resulting in a shared comprehension of the technical details of the mobile app among all parties. This encourages cooperation and aligns objectives and expectations.

**Quality Assurance:** The document outlines design factors, testing methods, and best practices that improve the overall reliability and quality of the mobile application. It ensures the finished product meets performance, security, and usability requirements.

**Risk Reduction:** By considering security, privacy, and scalability issues, the TDD helps identify potential risks and offers techniques for reducing them. During development, it promotes proactive risk management.

**Documentation:** It is an exhaustive account of all technological choices made during development. This documentation is helpful for maintenance, troubleshooting, and future reference.

**Onboarding and Training:** The TDD can be used to onboard new team members to the project. Their onboarding process is accelerated due to their ability to understand the technological architecture and design ideas quickly.

**Project Management:** Project managers can use the TDD to keep track of milestones, monitor development, and ensure the development team complies with all agreed upon technical standards and specifications.

**Compliance and Governance:** The TDD describes how the application complies with regulatory or compliance requirements, enabling audits and governance procedures for projects subject to these requirements.

The TDD will offer thorough technical guidance and documentation for the creation of the CogniOpen application, simplifying efficient communication, ensuring quality, and supporting all phases of the project from inception to maintenance.

## 1.2 Scope

The TDD for the CogniOpen is intended to provide a comprehensive technical blueprint that governs the entire application development lifecycle. This document addresses crucial aspects of the application, including architecture and design, database structure, backend services, front-end design, and application logic. The TDD is an exhaustive resource for the development team and stakeholders, ensuring that the mobile application is crafted efficiently, securely, and following the project's objectives. It describes the technical nuances, specifications, and guidelines necessary for the mobile application's successful development, testing, deployment, and ongoing support.

## 1.3 Objectives

The goal of CogniOpen is to develop a cross-platform, technically superior, and user-friendly app, prioritizing security, scalability, and performance while easily interacting with outside services. The app's dependability and continuous improvement relies on effective testing, deployment, monitoring, and maintenance procedures. These pillars of dependability work together to produce a high-quality, user-centered, and technically sound mobile application that is in line with the project's objectives. From a technological standpoint, the following objectives specify the application's purpose and objectives:

**Technical Excellence**: Ensure that the mobile application is developed that satisfies high requirements for performance, security, and scalability.

**Cross-Platform Compatibility**: Create the software with the goal of having it run smoothly across a variety of platforms, such as iOS, Android, and the web, while if necessary, adhering to platform-specific design standards.

**User Experience (UX):** Develop a user-friendly interface that is visually appealing while adhering to UX design principles. A great user experience will increase user happiness and engagement.

**Security and Privacy**: Implement strict security controls, such as data encryption, authentication, and authorization procedures to safeguard user information and privacy while upholding user confidence.

**Improved Performance**: To ensure quick load times and seamless user interactions, improve the application's performance through effective coding techniques, suitable caching systems, and responsive design.

**Scalability**: Create an architecture and backend services that are scalable so that the application can accommodate growing user demand and data volumes.

**Seamless Integration:** Integrate third-party services and APIs easily to improve the functionality of the application and give users more features.

**Resilient Backend**: To support the functionality of the application, establish a durable backend infrastructure with strong APIs, data storage, and server components.

The development team is directed by these objectives, which jointly describe the technical goals of "CogniOpen," to provide a high-quality, user-friendly, and technically sound product consistent with the project's overarching goals and vision.

## 1.4 Project Documents

This TDD will be included in the project deliverables' documentation. Throughout the project lifetime, every document is developed with the goal of assisting in the understanding, implementation, and maintenance of the CogniOpen application. The documents listed below will be included as project deliverables.

|  |  |  |
| --- | --- | --- |
| Document | Version | Date |
| Project Plan (PPL) | 1.2 | 10/28/2023 |
| Software Requirements Specification (SRS) | 1.2 | 10/28/2023 |
| Technical Design Document (TDD | 1.1 | 10/28/2023 |
| Test Plan (TP) | 1.1 | 10/28/2023 |
| Programmer Guide (PG) | 1.0 | 10/28/2023 |
| Deployment and Operations Guide (DOG) | 1.0 | 10/28/2023 |
| Software Test Report (STR) |  |  |
| User Guide (UG) |  |  |
| Traceability Matrix (TM) |  |  |

Table 1 - Project Documents

## 1.5. Definitions, Acronyms, and Abbreviations

Throughout this TDD, various terms and acronyms are used that are specifically related to the CogniOpen application. Defining these terms and acronyms guarantees complete clarity and comprehension, which in turn aids in understanding the application's features and parts. This inclusive glossary serves as a valuable point of reference for developers and stakeholders involved in the project, allowing for effective communication and alignment of understanding. To ensure clarity, their explanations are given below.

|  |  |
| --- | --- |
| Terms | Definition |
| API | Application Programming Interface |
| App | A program that is included on the App User’s mobile device |
| ASR | Automatic Speech Recognition |
| AWS | Amazon Web Services |
| Azure | Azure Web Services, a cloud provider of many services. |
| Deployment and Operations Guide | A document that explains how to deploy the program |
| DevOps | Development and Operations |
| DOM | Document Object Model |
| ERD | Entity Relationship Diagram |
| EULA | End User License Agreement |
| Flutter | A software framework for developing cross-platform mobile applications |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| iOS | iPhone Operating System |
| JSON | JavaScript Object Notation |
| Mobile Device | A smart phone, tablet, or some other portable computer with either the iOS or Android operating system |
| NPM | Node Package Manager |
| OO | Object Oriented |
| PG | Programmer’s Guide |
| PMP | Project Management Plan |
| PPL | Project Plan |
| REST | Representational State Transfer |
| SRS | Software Requirements Specification |
| STML | Short-Term Memory Loss |
| STP | Software Test Plan |
| STT | Speech to Text |
| TCP | Transmission Control Protocol |
| TDD | Technical Design Document |
| Term | Definition |
| TM | Traceability Matrix. A document that traces defects and test cases back to their requirement. |
| TR | Test Report |
| UG | User Guide |
| UI/UX | User Interface / User Experience |
| URL | Uniform Resource Locator |

Table 2 - Acronyms, Definitions, and Abbreviations

## 1.6 References

*University of Maryland Global Campus(UMGC) SWEN 670 Capstone Project Management System - CaPPMS*. (2023). Azurewebsites.net; CaPPMS. https://umgc-cappms.azurewebsites.net/previousprojects

# 2. System Overview

## 2.1. In Scope

The scope of this TDD includes crucial aspects of the application, including architecture and design, database structure, backend services, front-end design, and application logic. System design, database design, and associated services will include data flow diagrams. The component design and user interface design will be considered in scope for the previously mentioned areas of focus.

## 2.2. Out of Scope

This document will not cover design aspects related to the video recording interface, photo gallery interface, video gallery interface, or audio gallery interface screens. This work will be covered by Team A (Dream Team), including front-end and back-end services. Team A will provide a separate TDD including their design choices.

# 3. System Architecture

## 3.1. Architectural Design

This section describes the system architecture of CogniOpen. CogniOpen is composed of four main service domains: the Local Storage Service; Database Storage; AWS Transcribe, a STT service from AWS; and the ChatGPT API service. Each service is able to send, transform, and respond to data that is passed from the front-end services of CogniOpen to the back-end services. The high-level structural flow of data is described in the below diagram:

A diagram of several types of software

Description automatically generated with medium confidence

Figure 1 – Architectural Overview

In the architecture of the CogniOpen application, everything is driven by the user and their personal phone. The only interaction the CogniOpen application will how with another application on the user’s phone is with their preferred Calendar application. Since the audio portion of the application is the only set of features in scope for Team B, the diagram shows how actions the user may take will flow through the system. The user may request to create an audio recording or use the default audio recording on startup. This audio data will be fed into the CogniOpen Audio Processor which will interface with two third-party tools – AWS Transcribe and ChatGPT. The specific interactions with these tools will be described below.

## 3.2 Decomposition Description

## **3.2.1 Database**

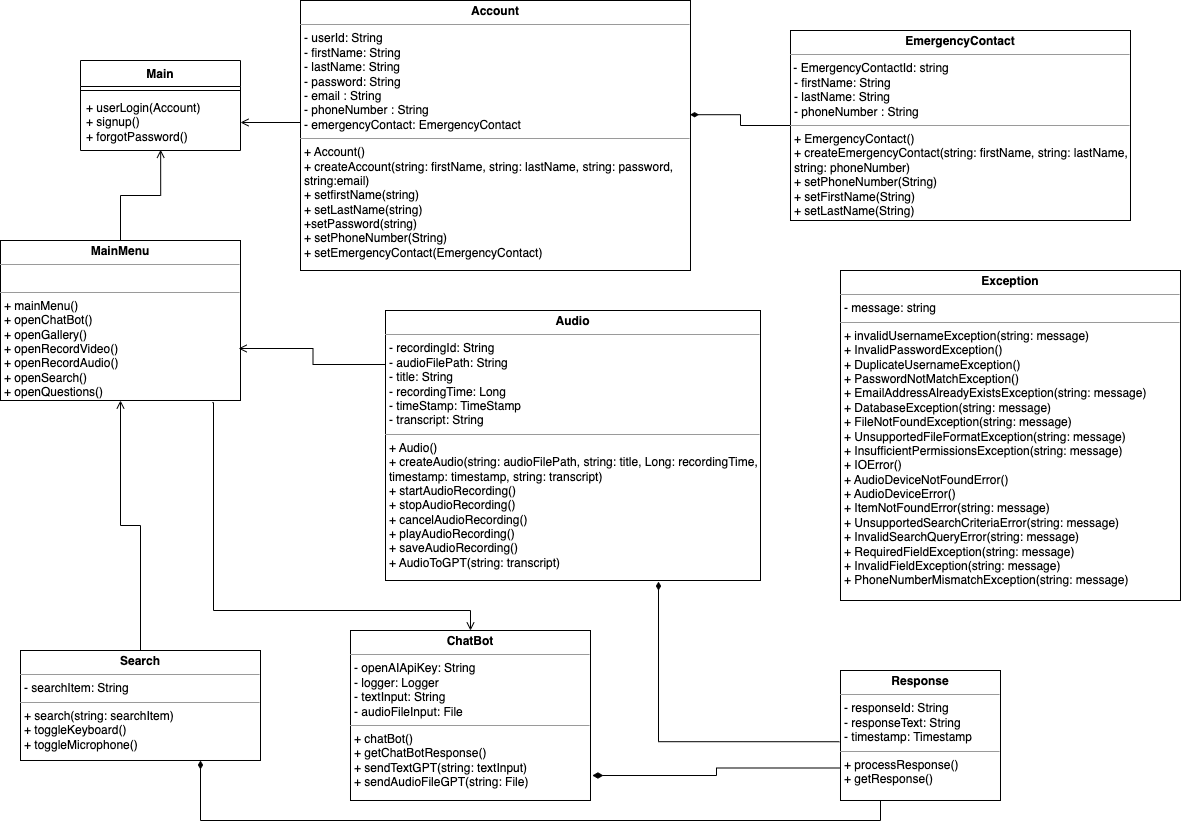
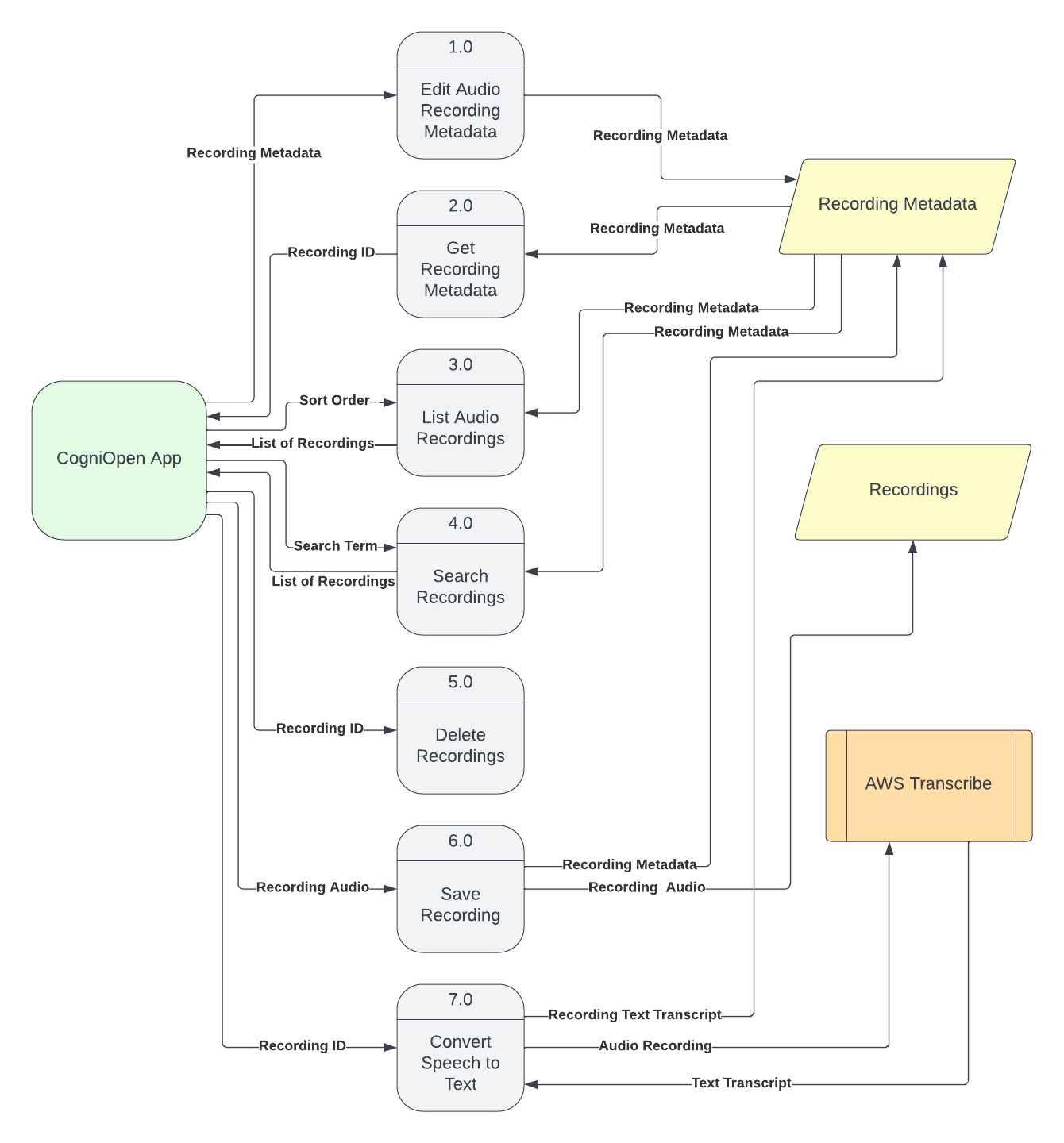


Figure 2 – Class Diagram

The diagram above provides a high-level overview of the application’s classes, class attributes, inheritance, functions, parameters, and return types. While subject to change as development continues, this provides implementing software developers a blueprint for how to start. Primarily driven by the MainMenu, inherited from Main, the Search, Audio, and VirtualAssistant classes each play an active role in providing integration services to the MainMenu class. The Response class will service the querying components of the Search, VirtualAssistant, and Audio classes. Finally, the EmergencyContact and Account classes will all provide critical user information that will be used with the Main class.

### 3.2.2 Transcription Service

Figure 3 – Transcription Service Dataflow

The Transcription Service will be how the CogniOpen application will interface and utilize AWS Transcribe and process audio data. There are several ways to interact with audio data. Some of the ways may include editing the metadata associated to an audio recording, querying the metadata, seeing all the available audio recordings that have been saved, searching the saved audio recordings, deleting a saved audio recording, saving an audio recording, and converting an audio recording to text. This is not an exhaustive list, but some of the various ways that the CogniOpen application plans to manipulate audio data. The application will pull saved recording metadata from the database and pull the physical files from cloud storage. When the application is ready to convert speech recordings into text, it will send the audio recording to AWS Transcribe and receive and textual transcription back. With this textual transcription, the application can query ChatGPT, which will be outlined in a section below.

### 3.2.3 ChatGPT Service

A diagram of a program

Description automatically generated with medium confidenceFigure 4 – ChatGPT Dataflow

ChatGPT will be used to query textual data for context. There are three primary ways that the CogniOpen team is planning to utilize ChatGPT, but there are many more possible uses that might be utilized as development continues. The three ways the team plans to utilize ChatGPT include keyword textual analysis, phrase contextual analysis, and emergency service requests. When performing keyword textual analysis, the system is trying to find the usage of a single word in a series of conversations. This may be useful to remember when you spoke to your doctor about an issue, the user may search for the term “fibromyalgia.” Phrase contextual analysis will most likely be the most common use for textual analysis, allowing the users to ask complex questions or phrases. When performing contextual analysis, a User may ask “did I commit to any appointments yesterday?” and the application can utilize ChatGPT to determine if any commitments were made. Finally, the CogniOpen application may send periodic requests to ChatGPT to parse audio conversations for emergency service needs. Using a customized set of textual and contextual analyses, the CogniOpen app could be used to determine if someone experienced a physical or emotional emergency.

### 3.2.4 Rekognition Service

The CogniOpen Application will use the AWS Rekognition Service for ingesting, processing, and inspecting video data. This specific service is out of scope for this TDD, but our accompanying team will cover the use of the AWS Rekognition Service in their supporting documentation.

## 3.3. Exception Handling

|  |  |
| --- | --- |
| Exception | Description |
| InvalidUsernameException | Username does not exist. |
| InvalidPasswordException | The password is invalid. |
| DuplicateUsernameException | The Username already exists. |
| PasswordNotMatchException | The passwords do not match. |
| EmailAddressAlreadyExistsException | An account with this email address already exists. |
| DatabaseException | An error occurred while searching the database |
| FileNotFoundException | Audio file could not be found. |
| UnsupportedFileFormatException | The audio file format is not supported. |
| InsufficientPermissionsException | The user does not have permission to access the audio file. |
| IOError | An error occurred while reading or writing the audio file. |
| AudioDeviceNotFoundError | The audio device is not available. |
| AudioDeviceError | An error occurred while using the audio device. |
| ItemNotFoundError | The item being searched for could not be found |
| UnsupportedSearchCriteriaError | The search criteria is not supported. |
| InvalidSearchQueryError | The search query is invalid. |
| RequiredFieldException | A required field is empty. |
| InvalidFieldException | a field contains invalid data. |
| PhoneNumberMismatchException | The phone number and confirm phone number fields do not match. |

Table 3 - Exceptions

# 4. Data Design

## 4.1. Data Description

This section explains the data design of the CogniOpen App. Each component within this architectural foundation is precisely tailored to support one or more functionalities of the app. The aspect of our strategy is the local storage of data on the device's file system, which offers rapid access while enhancing user data privacy.

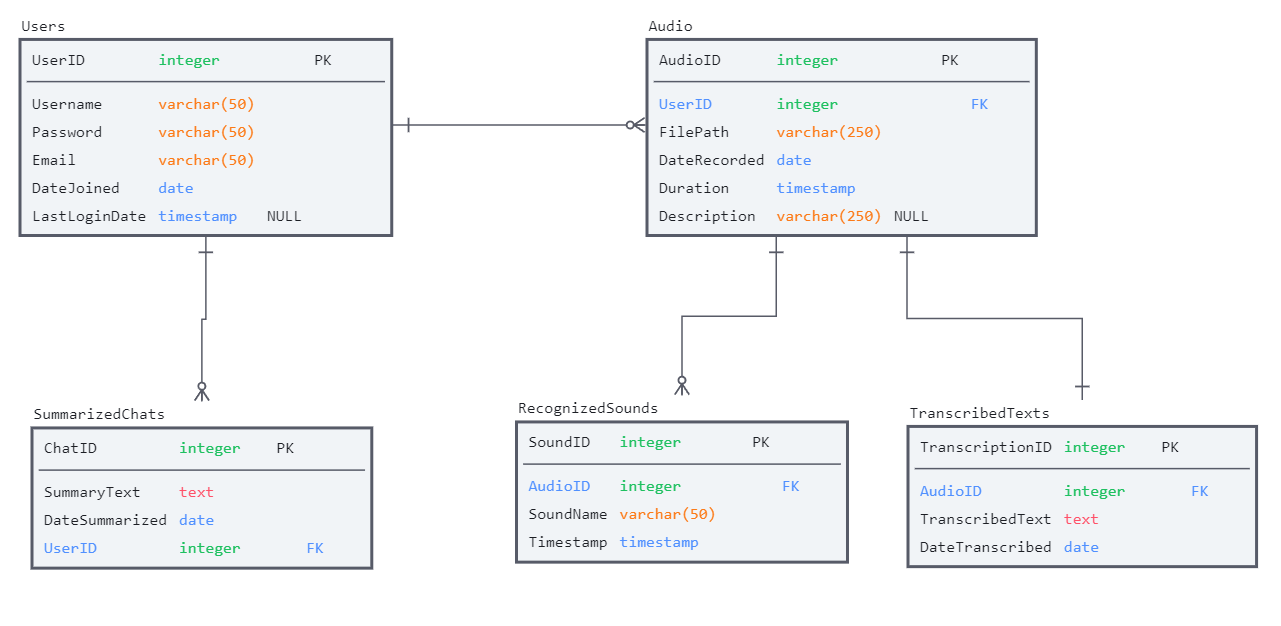
The Users Table acts as a repository for user credentials and activity logs. Its structured format streamlines the user authentication process and provides a blueprint for a seamless and enhanced user experience. The Audio Table plays a crucial role, systematically cataloguing audio recordings. Every entry in this table captures essential details, including file paths and recording durations, ensuring users can easily access their stored memories easily. The RecognizedSounds Table uses the JSON format, with data accessible via “recognized\_sounds.json”. Additionally, the TranscribedTexts Table and SummarizedChats Table are designed to provide users with easy access to transcriptions of their recordings and concise conversation summaries, respectively. Through this design, CogniOpen exemplifies its dedication to delivering structured organization and a user-centric experience.



Figure 5 – File Format Example

In the above JSON structure, the root object has a key `recognized\_sounds` which holds an array of sound entries. Each sound entry is an object that corresponds to a row in the `RecognizedSounds Table` with keys representing the columns: `SoundID`, `AudioID`, `SoundName`, and `Timestamp`.

**Figure: ERD Diagram**

Figure 6 – Entity Relationship Diagram

The figure above is an ERD representing how the database tables will be constructed and their relationships to one another. The section below will go into detail about the individual table names, the column names in each table, the data type of each column, a brief description of what that field is used for, and any constraints that field may have.

## 4.2. Data Dictionary

### 4.2.1 Users Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints |
| Users | UserID | INT | Unique identifier for each user | Primary Key, Not Null |
|  | Username | VARCHAR | User's chosen name | Not Null, Unique |
|  | Password | VARCHAR | Hashed version of the user's password | Not Null |
|  | Email | VARCHAR | User's email address | Not Null, Unique |
|  | DateJoined | DATE | Date when the user registered | Not Null |
|  | LastLoginDate | TIMESTAMP | Most recent date and time the user logged in | Nullable |

Table 4 – Users Table

### 4.2.2 Audio Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints |
| Audio | AudioID | INT | Unique identifier for each audio recording | Primary Key, Not Null |
|  | UserID | INT | Foreign key referencing Users table | Foreign Key, Not Null |
|  | FilePath | VARCHAR | Path to the stored audio file | Not Null |
|  | DateRecorded | DATE | Date when the audio was recorded | Not Null |
|  | Duration | TIME | Duration of the audio recording | Not Null |
|  | Description | VARCHAR | Optional notes added by the user about the recording | Nullable |

Table 5 – Audio Table

### 4.2.3 Recognized Sounds Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints |
| RecognizedSounds | SoundID | INT | Unique identifier for each recognized sound | Primary Key, Not Null |
|  | AudioID | INT | Foreign key referencing Audio table | Foreign Key, Not Null |
|  | SoundName | VARCHAR | Name or description of the recognized sound | Not Null |
|  | Timestamp | TIME | Exact time in the audio when the sound was recognized | Not Null |

Table 6 – Recognized Sounds Table

### 4.2.4 Transcribed Texts Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints |
| TranscribedTexts | TranscriptionID | INT | Unique identifier for each transcription | Primary Key, Not Null |
|  | AudioID | INT | Foreign key referencing Audio table | Foreign Key, Not Null |
|  | TranscribedText | TEXT | Textual version of the audio recording | Not Null |
|  | DateTranscribed | DATE | Date when the audio was transcribed | Not Null |

Table 7 – Transcribed Texts Table

### 4.2.5 Summarized Chats Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints |
| SummarizedChats | ChatID | INT | Unique identifier for each summarized chat | Primary Key, Not Null |
|  | AudioID | INT | Foreign key referencing Audio table | Foreign Key, Not Null |
|  | SummaryText | TEXT | Concise summary of the chat | Not Null |
|  | DateSummarized | Date | Date when the chat was summarized | Not Null |

Table 8 – Summarized Chats Table 5. Component Design

## 5.1. Class: homeScreen

**Class Description:** *This class allows users to access the home screen.*

**Class Inheritance:** *This class inherits from the StatefulWidget class.*

**Class Attributes:**

*const Color*

*const SizedBox*

*const Text*

*const Icon*

*const PopupMenuItem<String>*

*const BoxDecoration*

*const Padding*

*const TextStyle*

**Exceptions Thrown:**

*N/A*

**Class Constructors:**

*N/A*

**Class Methods:**

*N/A*

## 5.2. Class: assistantScreen

**Class Description:** *This class allows users to access the virtual assistant*

**Class Inheritance:** *This class inherits from the StatefulWidget class*

**Class Attributes:**

*late FlutterTts*

*bool isPlaying*

*TextEditingController \_messageController*

*ScrollController \_scrollController*

*List<ChatMessage> \_chatMessages*

*late Future<bool> goodAPIKey*

*bool \_isTyping*

*final Function toggleTTS*

*final String messageText*

*final bool isUserMessage*

**Exceptions Thrown:**

*RequestFailedException*

*UnknownException*

**Class Constructors:**

*Conversation(this.summary, Media media)*

*ChatMessage({*

*required this.messageText,*

*required this.isUserMessage,*

*required this.toggleTTS,*

*});*

**Class Methods:**

*void initState()*

*void dispose()*

*void initTTS()*

*void toggleTTS(String text)*

*Future<void> \_handleUserMessage(String messageText, bool display)*

*void \_scrollDown()*

*Future<String> getChatGPTResponse(String userMessage)*

*Widget build(BuildContext context)*

*Future<bool> loadAPIKey()*

*FutureOr \_showAlert(String title, String message)*

## 5.3. Class: audioScreen

**Class Description:** *This class provides the main interface for audio recording.*

**Class Inheritance:** *This class inherits from the StatefulWidget class*

**Class Attributes:**

*enum MediaFormat*

*const List<MediaFormat> mediaFormats*

*FlutterSoundRecorder? \_recorder*

*FlutterSoundPlayer? \_player*

*bool \_isRecording*

*bool \_isPlaying*

*Duration \_duration*

*String? \_pathToSaveRecording*

*Timer? \_timer*

*final \_bucketName*

*final service*

*var key2*

*S3Bucket s3Connection*

*String transcription*

*void initState()*

**Exceptions Thrown:**

*N/A*

**Class Constructors:**

*N/A*

**Class Methods:**

*Future<void> \_initializeRecorder()*

*Future<bool> \_requestPermissions()*

*void dispose()*

*Future<void> \_startRecording()*

*Future<void> \_stopRecording()*

*Future<void> \_startPlayback()*

*Future<void> \_stopPlayback()*

*Future<void> \_transcribeAudio(String s3Url)*

*Future<void> \_saveTranscriptionToFile(String transcriptionJobName)*

*Widget build(BuildContext context)*

## 5.4. Class: registrationScreen

**Class Description:** This class allows the user to register their account.

**Class Inheritance:** *This class inherits from the StatefulWidget class*

**Class Attributes:**

*final GlobalKey<FormState> \_formKey*

*TextEditingController \_firstNameController*

*TextEditingController \_lastNameController*

*TextEditingController \_emailController*

*bool \_useFaceID*

*bool \_isButtonActive*

**Exceptions Thrown:**

N/A

**Class Constructors:**

N/A

**Class Methods:**

*Future<String> get \_localPath*

*Future<File> get \_localFile*

*Future<File> writeUserData(String data)*

*Widget build(BuildContext context)*

## 5.5. Class: loginScreen

**Class Description:** *This class will allow the manipulation of media data within the application.*

**Class Inheritance:** *This class inherits from the StatefulWidget class.*

**Class Attributes:**

*final LocalAuthentication \_localAuth*

**Exceptions Thrown:**

*N/A*

**Class Constructors:**

*N/A*

**Class Methods:**

*Future<void> \_authenticateWithAllMethods()*

*Widget build(BuildContext context)*

## 5.6. Class: profileScreen

**Class Description:** This class allows users to read and write their profile information.

**Class Inheritance:** *This class inherits from the StatefulWidget class.*

**Class Attributes:**

*final GlobalKey<FormState> \_formKey*

*TextEditingController \_firstNameController*

*TextEditingController \_lastNameController*

*TextEditingController \_emailController*

*TextEditingController \_phoneController*

*TextEditingController \_emergencyFirstNameController*

*TextEditingController \_emergencyLastNameController*

*TextEditingController \_emergencyPhoneController*

*String \_biometricAuth*

**Exceptions Thrown:**

*N/A*

**Class Constructors:**

*Photo(this.associatedImage, Media media)*

**Class Methods:**

*Future<String> get \_localPath*

*Future<File> get \_localFile*

*Future<String> readUserData()*

*Future<File> writeUserData(String data)*

*Future<void> populateData()*

*Future<void> setBiometricAuth()*

*void initState()*

*Widget build(BuildContext context)*

# 6. Application Interface Design

## 6.1. User Interface Design

This section provides an overview of the user interface (UI) of the CogniOpen application, explaining how users can easily navigate the system for different tasks. It includes screen images accompanied by textual descriptions of the essential aspects of the UI. The section also highlights design considerations for the UI to meet the needs of cognitively impaired individuals. This document serves as a technical reference for system developers to ensure that the application design adheres to industry standards while delivering a user-friendly experience.

The application will provide six main features including an AI-powered Virtual Assistant, gallery, video recording, audio recording, search, and recent questions/requests. Users can access the AI feature to ask questions and receive information through voice or text input. The platform also enables users to create, store, and manage their video, audio, and photo recordings. Additionally, users can search for trigger words within their files, and view a list of previously asked questions/requests. CogniOpen offers a help center with FAQs and a guided application tour, along with customizable settings to adjust the AI, audio, and video features to meet specific needs.

## 6.2. Considerations for Designing the User Interface

CogniOpen aims to meet the specific requirements for the Mobile App Rating Scale Assessment (MARS) to ensure the needs of individuals suffering from cognitive impairments, particularly those with dementia, are met. MARS is an assessment tool designed to evaluate the quality, functionality, and usability of mobile applications. The MARS assessment tool consists of five distinct components:

1. Engagement: This component assesses the application's ability to captivate and maintain the user's attention, keeping them engaged with the app.

2. Functionality: This component evaluates the application's basic features, functions, and capabilities, including ease of use, performance, and reliability.

3. Aesthetics: This component assesses the application's visual design, layout, and overall UI.

4. Information Quality: This component evaluates the application's ability to provide accurate, relevant, and useful information to the user.

5. Subjective Quality: This component assesses the overall user experience, including user satisfaction, willingness to use the application in the future, and likelihood of recommending the app to others.

CogniOpen will use the MARS assessment tool to evaluate the application's features and functions based on these components. By utilizing the MARS assessment tool, CogniOpen aims to provide a user-friendly and accessible platform that helps individuals with cognitive impairments enhance their quality of life.

## 6.3 Screen Image

Below are the mock views of the most important aspects of the CogniOpen application's graphical user interface (GUI), along with a description of their user experience. The list of screens is as follows:

* User login credential screen
* User application registration screen
* Home interface screen
* Virtual Assistant interface screen
* Record audio interface screens
* Record video interface screens - Out of Scope for Team B
* Photo gallery interface screens - Out of Scope for Team B
* Video gallery interface screens - Out of Scope for Team B
* Audio gallery interface screens - Out of Scope for Team B
* Search interface screen
* Previously asked questions interface screen
* Menu interface screen
* Profile interface screen
* Guided tour interface screens
* Onboarding interface screens

### 6.3.1: User Login Credential Screen

This is the first user interface screen that the user shall see upon launching the application. The user will be greeted with a login screen that utilizes a PIN and FaceTime biometric credentials for secure access to their account. With the FaceTime biometric option, the user can simply use their facial features to log in to their account. This provides an extra layer of security, making it harder for unauthorized users to gain access to the account. In addition to FaceTime, the login interface will also display a PIN login option that users can use to input their login credentials and access their account. If, by chance, the user forgets their PIN, they can reset it by selecting the "Forgot PIN" option. In case the user is a first-time user, they will be presented with the option of creating a new account by clicking on "Create Account", and the "Create Account" screen will populate.

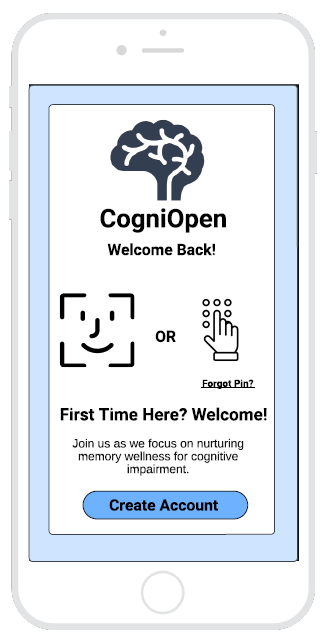


Figure 6: User Login Screen

### 6.3.2: User Application Registration Screen

If the user needs to create a new account, the application shall present the Figure 2 screen. This screen shall feature four text fields – first name, last name, email address, and pin. Also, there will be an on and off slider bar for the FaceTime biometric option, which will allow the users to enable or disable it. If the FaceTime biometric option is turned on, the user can simply use their facial features to log in to their account. Additionally, the screen shall include a "Create Account" button, which shall only become active once all four text fields have been correctly inputted. Once the account creation process is complete, the application shall redirect the user to the onboarding page as shown in Figure . The screen shall also provide separate links beneath the "Create Account" button, which the user can select to access the application's "Privacy Policy" and "Terms and Conditions".

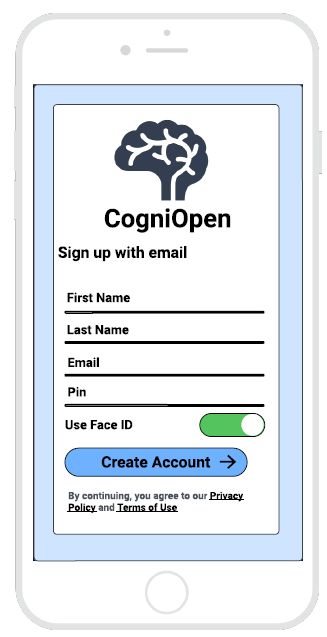


Figure 7: User Registration Screen

### 6.3.3: Home Interface Screen

Upon successful login to the application, the user will be presented with Figure 8, showcasing the main features available in the form of buttons. These features include an artificial intelligent (AI) Virtual Assistant, gallery, video recording, audio recording, search, and recent questions/requests. All screens, except the user login and registration, will display static features for essential functions at the top and bottom of the screen. The top-left corner of the screen will feature a menu that opens a dropdown with Profile, Help Center, Customizable Application, and Logout options. The top-right corner of the screen will display a back button, allowing users to navigate to the previous screen. The bottom of the screen will show a navigation bar that provides users with quick access to the Home screen, Search screen, Gallery screen, and Virtual Assistant screen.

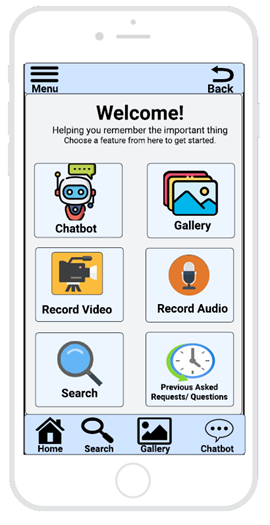


Figure 8: Home Screen

### 6.3.4: Virtual Assistant Interface Screen

The Virtual Assistant screen is a critical component of the application designed to facilitate user interaction with an AI assistant. This screen, as seen in Figure 9, features a text box where users can input questions or requests either by speaking through their phone's microphone (Figure 10) or typing into the keyboard text box (Figure 11). The Virtual Assistant responds to these queries with text-based replies or presents the user with options to choose from. Additionally, the screen may include buttons or links that direct users to specific functions or areas within the application. The chat history of the conversation between the user and the Virtual Assistant is also displayed on the screen, and the Virtual Assistant will continue to answer the user's questions and requests until the user closes the application or navigates to a different screen.

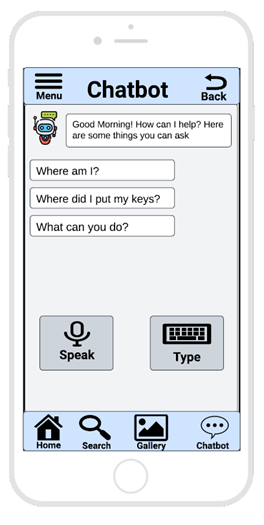


Figure 9: Virtual Assistant Screen

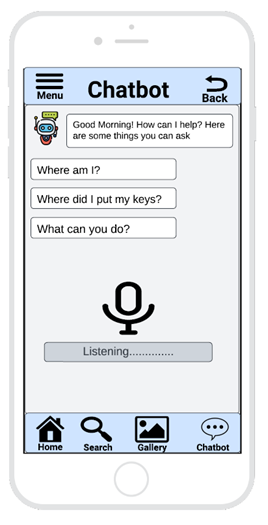


Figure 10: Virtual Assistant Active Listening

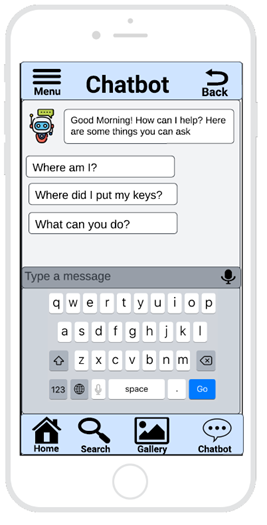


Figure 11: Virtual Assistant Textbox

### 6.3.5: Record Audio Interface Screens

The Record Audio screen, Figure 12, shall display a circular button at the center of the screen that the user can tap to initiate the recording process. The screen shall also feature a timer that displays the length of the recording, starting at 00:00:00. During the recording process, Figure 13 shall display a square-shaped button that replaces the circular button, indicating that the recording is in progress. The timer shall begin counting up, indicating the duration of the recording, and a "Stop" button shall be displayed to allow users to end the recording at any time. Once the recording is stopped, Figure 14 shall display a preview of the recorded audio file, and two buttons shall be displayed at the bottom of the screen. A "Save" button shall be present to allow users to save the recorded audio file to the device's storage, and a "Cancel" button shall be present to discard the recording and return to the Record Audio screen.



Figure 12: Audio Screen



Figure 13: Audio InProgress Screen

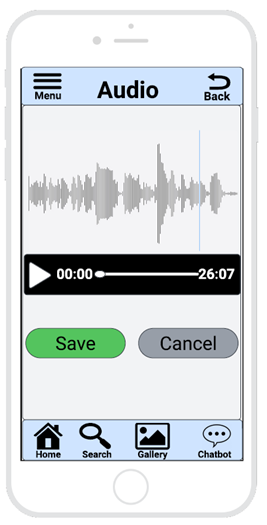


Figure 14: Save or Cancel Audio Screen

### 6.3.6: Search Interface Screen

The Search screen features shall have one text field to allow users to enter a keyword or phrase to search for specific content. The user can input their search content either by speaking through their phone's microphone or typing into a text box. The microphone button shall be present on the keyboard to allow users to switch between the typing and speaking input modes.

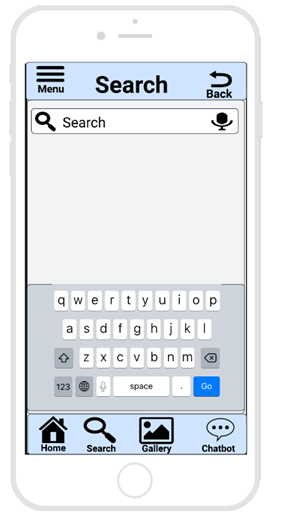


Figure 15: Search Screen

### 6.3.7: Previously Asked Questions Interface Screen

For Figure 16, the Previously Asked Questions interface screen shall display a list of recent questions or requests in the form of text boxes. Users can tap on any of these text boxes to quickly view the corresponding question or request. Additionally, this screen features two buttons that allow users to input questions or requests either by speaking through their phone's microphone or typing into the keyboard text box.

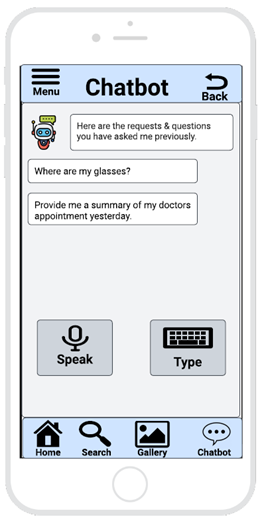


Figure 16: Previously Asked Questions Screen

### 6.3.8: Menu Interface Screen

In the menu screen, the drop-down menu will be located at the top of the screen and will consist of four button functions: My Profile, Guided Tour, Logout and Exit. Users can click on the menu button and select any of the functions to proceed. The 'My Profile' button will allow users to view and edit their personal information. The 'Guided Tour' button will provide users with a guided tour of the application's features and functionalities. The 'Logout' button will log the user out of the application, while the 'Exit' or the “X” button will close the menu drop-down window.



Figure 17: Menu Screen

### 6.3.9: Profile Interface Screen

The profile screen of the application will have ten text fields allowing users to easily edit their personal information, including their first name, last name, email address, phone number, emergency first name, emergency last name, emergency phone number, and pin . In addition to these text fields, the screen will also feature a "Save" button and a "Cancel" button. These buttons will allow users to either save the changes they have made to their profile or cancel any changes they no longer wish to make. This interface will enable users to easily manage their personal information and make necessary updates whenever needed.

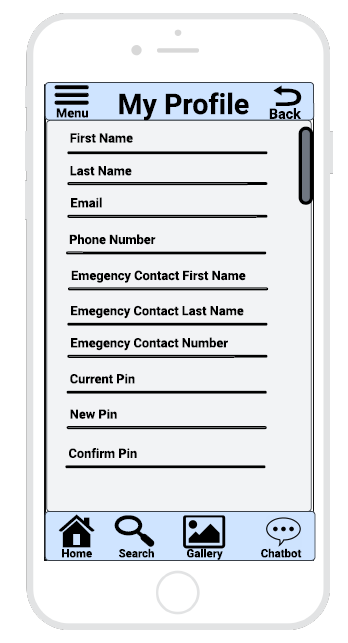


Figure 18: My Profile Screen

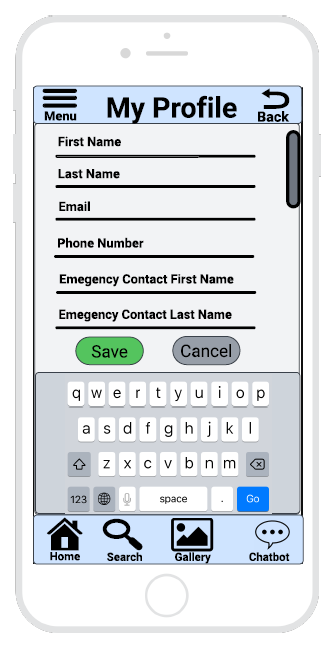


Figure 19: My Profile Screen Top View with Keyboard

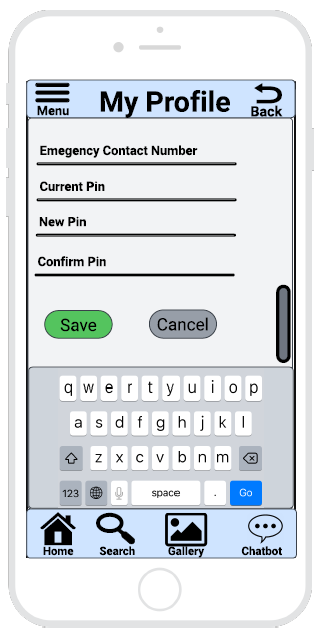


Figure 20: My Profile Screen Bottom View with Keyboard

### 6.3.10: Guided Tour Interface Screens

For Figure 21 to 31, the screens displayed will be for the guided tour. This feature is designed to show users how to navigate throughout the application. The screens will provide step-by-step instructions and visual aids to help users become familiar with the application's features and functionality. The guided tour is an important part of the user experience, as it can help users feel more comfortable and confident using the application. At the bottom of the screen, there shall be two arrows so users can navigate through the guided tour, moving forward or backward. Additionally, there will be an "X" button that will close the guided tour window and return the user to the home screen. This will provide users with the option to exit the guided tour at any time and return to using the application on their own.

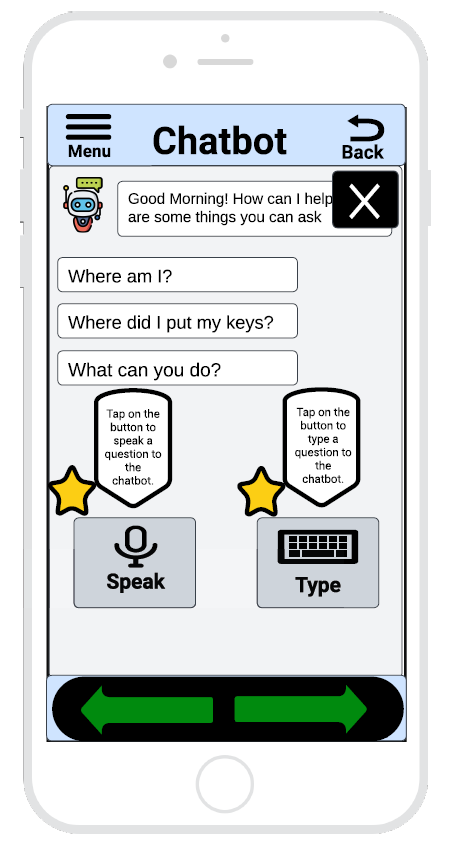


Figure 21: Virtual Assistant Guided Tour

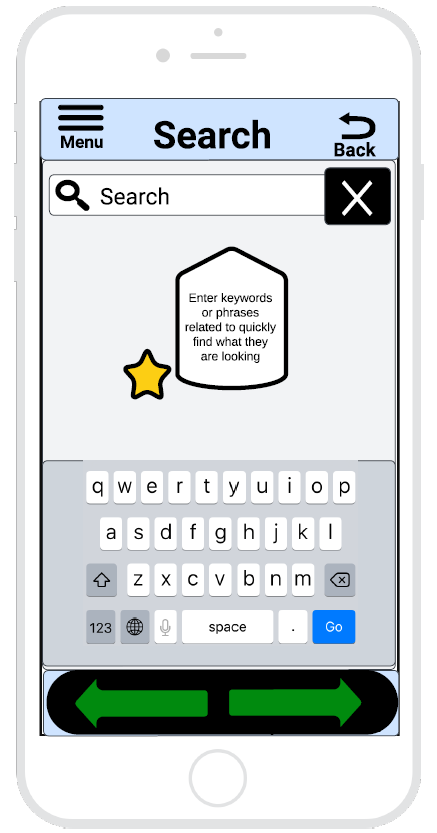


Figure 22: Search Guided Tour

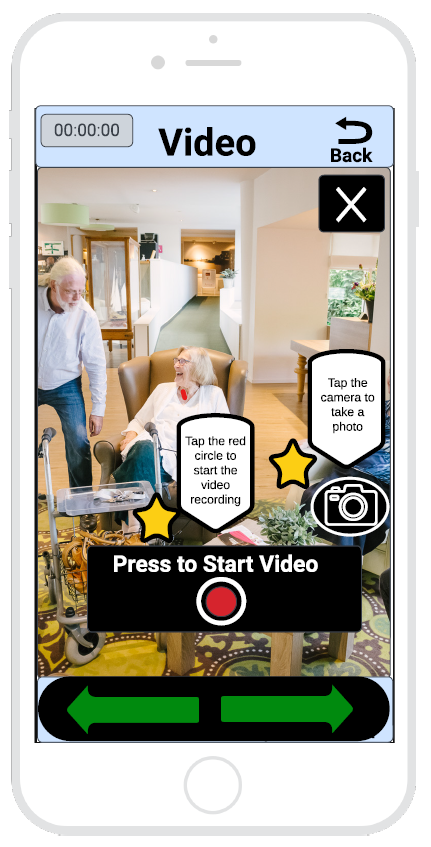


Figure 23: Video Guided Tour

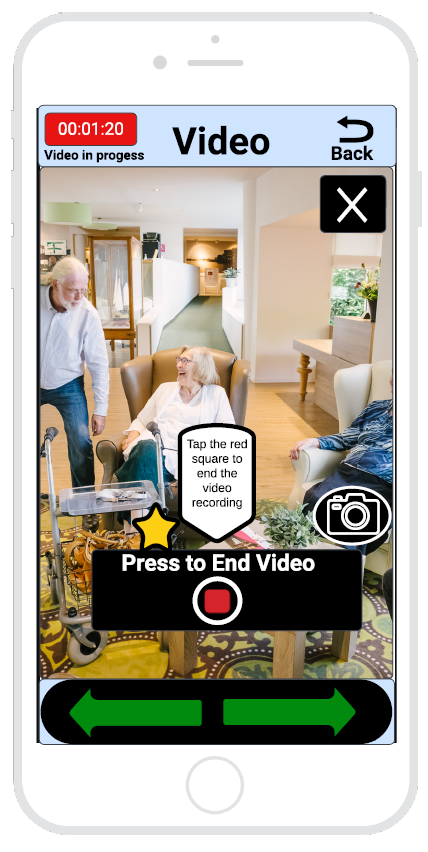


Figure 24: Video Guided Tour

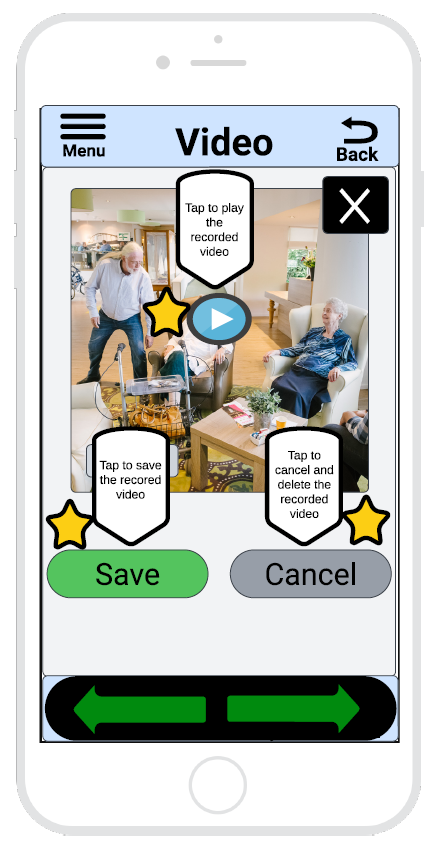


Figure 25: Video Guided Tour

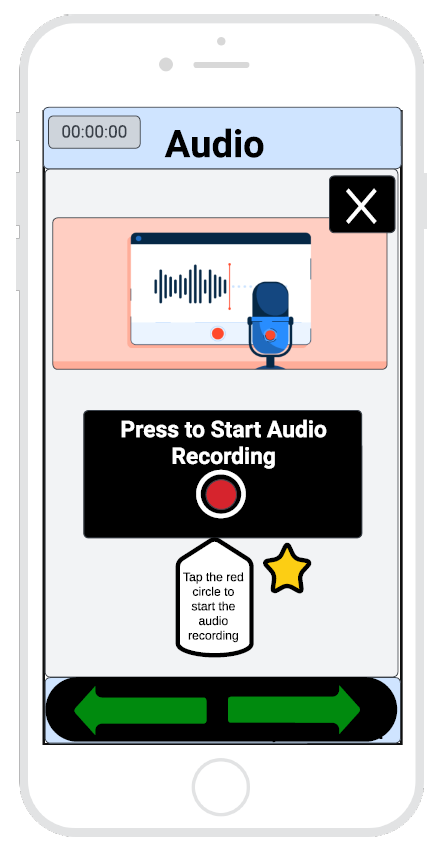


Figure 26: Audio Guided Tour



Figure 27: Audio Guided Tour



Figure 28: Audio Guided Tour

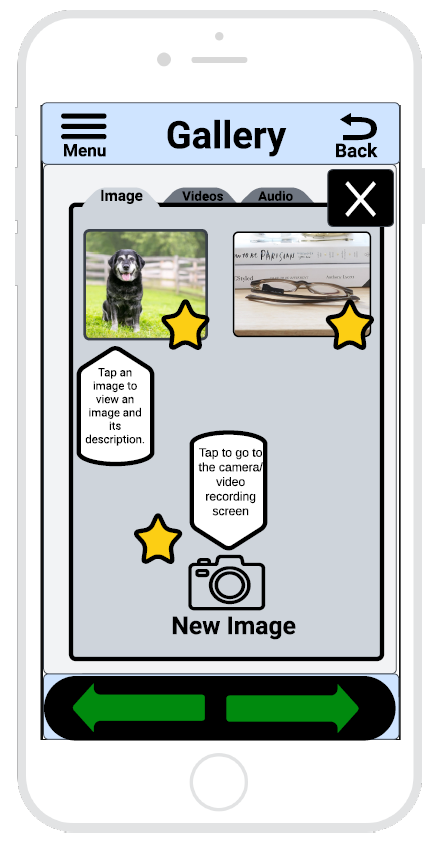


Figure 29: Gallery Guided Tour



Figure 30: Gallery Guided Tour



Figure 31: Gallery Guided Tour

6.3.11 Onboarding Interface Screens

For Figure 32 to 36, the screens displayed the onboarding interface screens. Upon completing the user registration, the user will be presented with an interactive onboarding interface that will guide them on how to use the application's virtual assistance. The interface will be powered by a human-like artificial intelligence that will have an introductory conversation with the user and receive real-time feedback. The virtual assistant will ask the user questions and respond in a human-like manner, making the conversation feel more natural and engaging. The AI-powered assistant will also be able to store the user's verbal input, allowing it to learn more about the user's preferences and usage patterns over time. To move to the next step, the interface will include a "Next" button feature on the screen and the user can also say "Hey Sam, next page" to proceed at their own pace. At the end of the conversation, the user will be given the option to request a summary of the conversation. Upon requesting a summary, the virtual assistant will provide a detailed summary of the conversation, highlighting key points and recommendations discussed during the conversation. In Figure 36, the user will request to be directed to the guided tour section, which will provide step-by-step instructions on how to use the application's features. This will enable the user to gain a better understanding of how to navigate through the application.

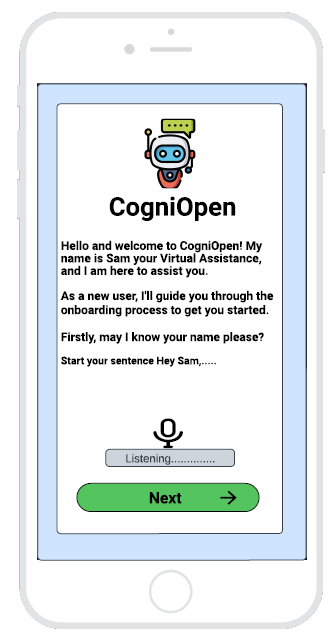


Figure 32: Onboarding Introduction Interface Screen

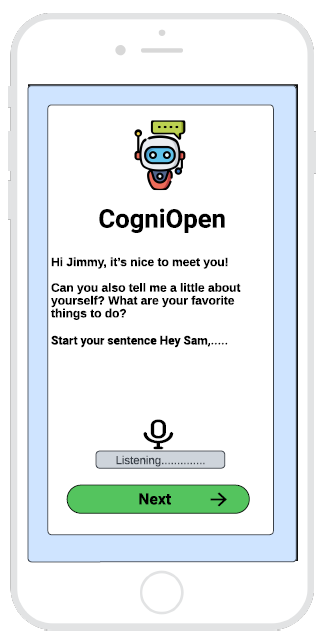


Figure 33: Onboarding Conversation Interface Screen

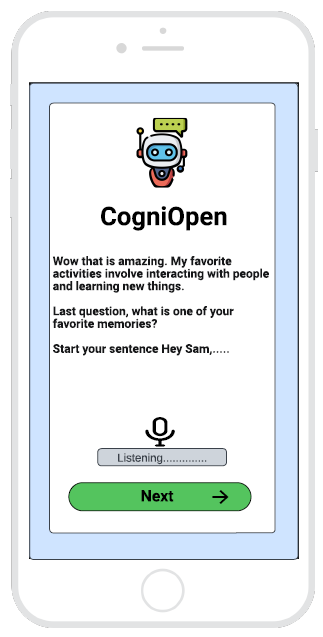


Figure 34: Onboarding Conversation Interface Screen

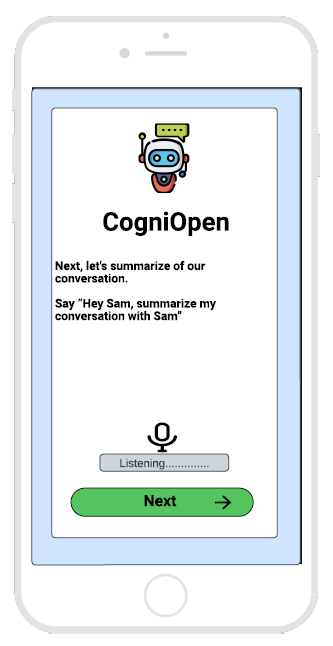


Figure 35: Onboarding Request Summary Interface Screen

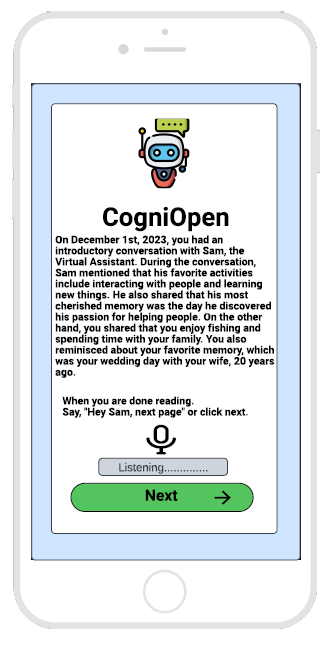


Figure 36: Onboarding Summary Interface Screen

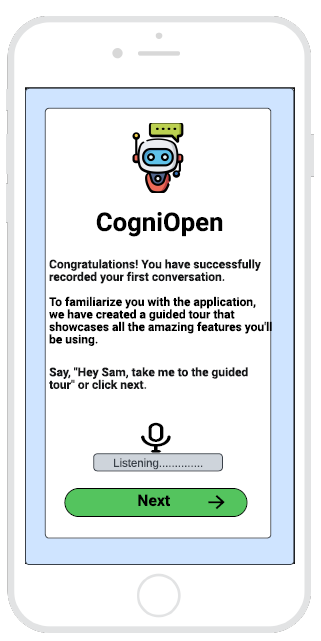


Figure 36: Onboarding Redirect to Guided Tour Interface Screen

# 7. Requirements Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| SRS Requirement Number | Requirement Description | Application Component | UI Screen |
| 3.1.1 | Accessing the application | **5.1** UserLogin  **5.2** UserRegistration | **6.3.1** User Login Credential Screen  **6.3.2** User Application Registration Screen |
| 3.1.2 | Interacting with the Virtual Assistant | **5.4** Virtual Assistant | **6.3.4** Virtual Assistant Interface Screen |
| 3.1.3 | Create a video recording | Out of Scope for Team B | |
| 3.1.4 | View a recorded video | Out of Scope for Team B | |
| 3.1.5 | Delete a recorded video | Out of Scope for Team B | |
| 3.1.6 | Create an audio recording | **5.5** Audio | **6.3.5** Record Audio Interface Screen |
| 3.1.7 | Listen to an audio recording | **5.5** Audio | **6.3.5** Record Audio Interface Screen |
| 3.1.8 | Delete an audio recording | **5.5** Audio | **6.3.5** Record Audio Interface Screen |
| 3.1.9 | Create a new photo | Out of Scope for Team B | |
| 3.1.10 | View saved photos | Out of Scope for Team B | |
| 3.1.11 | Delete a photo | Out of Scope for Team B | |
| 3.1.12 | Utilize the search function | **5.6** Search | **6.3.6** Search Interface Screen |
| 3.1.13 | Accessing the recent questions/request | **5.7**  PreviouslyAskedQuestions | **6.3.7** Previously asked questions Screen |
| 3.1.14 | View the menu | **5.8** MenuInterfaceScreen | **6.3.8** Menu interface screen |

Table 9 – Requirements Matrix

# 8. Appendices

## 8.1 Speech to Text service provider comparison

There are many providers of speech to text (STT) services suitable for the CogniOpen app. The table below offers a comparison of the providers that were considered and identifies the pros and cons of each. To minimize development costs, only providers with a free trial or free usage were considered. The accuracy of each service was not evaluated for the same reason and because STT accuracy is dependent on the quality of the audio input, which was not reliably known at the time of comparison. It is recommended that this comparison is revisited prior to usage in any production environment.

|  |  |  |
| --- | --- | --- |
| Service Name | Pros | Cons |
| Amazon Transcribe | * + Free tier includes 60 minutes per month   + Able to support multiple languages   + Can differentiate up to 10 unique speakers   + Supports custom vocabularies to correct transcription inaccuracies | * + Asynchronous processing requires uploading files to an S3 bucket   + Costs $0.012 per minute after free minutes are used |
| OpenAI Whisper | * + Uses the same OpenAI account and API key as ChatGPT   + Files are sent directly to transcription service   + New accounts are provided $5 in free credit   + Accepts m4a, mp3, mp4, mpeg, mpga, wav, and webm formats | * + Use of free credit conflicts with ChatGPT usage   + Does not include attribution   + Costs $0.006 per minute |

Table 10 – Speech to Text Comparison

|  |  |  |
| --- | --- | --- |
| Service Name | Pros | Cons |
| Microsoft Azure | * + Free tier includes 5 hours per month   + Includes speaker attribution   + Can provide confidence score and change the display form of recognized text | * + API only supports .wav files   + Batch processing requires a dedicated Docker container   + Costs $1 per hour after free time is used |
| Google Cloud Speech-to-Text | * + Free tier includes 60 minutes per month   + Offers speaker attribution   + Offers multiple models, each optimized for a different type of audio   + Can be run locally on device to improve data security and privacy | * + Audio must be embedded in the API request or stored in Google Cloud storage   + Costs $0.024 per minute after free time is used |

Table 11 – Speech to Text Comparison Continued