|  |  |  |
| --- | --- | --- |
| amblem | **ÇANKAYA UNIVERSITY**  **Software Engineering Department** | **A circular logo with colorful arrows  Description automatically generated** |

**SENG 491 – 492 Graduation Project**

**Software Requirements**

**Specification**

**<<Project Name>>**

**<<Project Members>>**

**Version <<1.0>>**

Table of Contents

[1 Introduction 1](#_Toc170745019)

[1.1 Purpose 1](#_Toc170745020)

[1.2 Scope 1](#_Toc170745021)

[1.3 Definitions, acronyms, and abbreviations 1](#_Toc170745022)

[1.4 References 1](#_Toc170745023)

[1.5 Overview 1](#_Toc170745024)

[1.6 Version History 1](#_Toc170745025)

[2 Overall description 2](#_Toc170745026)

[2.1 Product perspective 2](#_Toc170745027)

[2.1.1 System interfaces 2](#_Toc170745028)

[2.1.2 User interfaces 2](#_Toc170745029)

[2.1.3 Hardware interfaces 3](#_Toc170745030)

[2.1.4 Software interfaces 3](#_Toc170745031)

[2.1.5 Communications interfaces 3](#_Toc170745032)

[2.1.6 Memory constraints 3](#_Toc170745033)

[2.1.7 Operations 3](#_Toc170745034)

[2.1.8 Site adaptation requirements 4](#_Toc170745035)

[2.2 Product functions 4](#_Toc170745036)

[2.3 User characteristics 4](#_Toc170745037)

[2.4 Constraints 4](#_Toc170745038)

[2.5 Assumptions and dependencies 5](#_Toc170745039)

[2.6 Apportioning of requirements 5](#_Toc170745040)

[3 Specific Requirements 5](#_Toc170745041)

[3.1 External Interface Requirements 5](#_Toc170745042)

[3.1.1 User interfaces 6](#_Toc170745043)

[3.1.2 Hardware interfaces 6](#_Toc170745044)

[3.1.3 Software interfaces 6](#_Toc170745045)

[3.1.4 Communications interfaces 6](#_Toc170745046)

[3.2 Functional Requirements 6](#_Toc170745047)

[3.2.1 User class 1 7](#_Toc170745048)

[3.2.2 User class 2 7](#_Toc170745049)

[3.3 Performance requirements 7](#_Toc170745050)

[3.4 Logical database requirements 8](#_Toc170745051)

[3.5 Design constraints 8](#_Toc170745052)

[3.5.1 Standards compliance 8](#_Toc170745053)

[3.6 Software system attributes 8](#_Toc170745054)

[3.6.1 Reliability 8](#_Toc170745055)

[3.6.2 Availability 9](#_Toc170745056)

[3.6.3 Security 9](#_Toc170745057)

[3.6.4 Maintainability 9](#_Toc170745058)

[3.6.5 Portability 9](#_Toc170745059)

[4 References 9](#_Toc170745060)

# Introduction

## Purpose

This subsection should

1. Delineate the purpose of the SRS;
2. Specify the intended audience for the SRS.

## Scope

This subsection should

1. Identify the software product(s) to be produced by name (e.g., Host DBMS, Report Generator, etc.);
2. Explain what the software product(s) will, and, if necessary, will not do;
3. Describe the application of the software being specified, including relevant benefits, objectives, and goals;
4. Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist.

## Definitions, acronyms, and abbreviations

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.

## References

This subsection should

1. Provide a complete list of all documents referenced elsewhere in the SRS;
2. Identify each document by title, report number (if applicable), date, and publishing organization;
3. Specify the sources from which the references can be obtained.

If the list of references is too long, it should be provided at the end of the document as a separate section.

## Overview

This subsection should

1. Describe what the rest of the SRS contains;
2. Explain how the SRS is organized.

## Version History

Provide a table of table of changes to this document

|  |  |  |
| --- | --- | --- |
| **Version No** | **Description of change** | **Date** |
| 1.0 | Initial Release | 15.10.2024 |
|  |  |  |

# Overall description

## Product perspective

### This project aims to develop a multi-modal psychology chatbot that analyzes users' emotional states and contexts and helps people with psychological guadline. This chatbot will perform emotional analysis using biometric data obtained from voice, facial expressions and wearable devices (e.g. smart watches) and provide personalized feedback. Users can use this application when they need instant support, check their old data and perform emotional state analysis. The application can be authenticated with the authentication API. Users' previous conversations will be deleted after a certain period of time in the database.

### System interfaces

### User Interface: The interface through which users interact with the chatbot. This interface receives input from the user, such as voice, facial expressions, biometric data, and text input.

### Voice Recognition Interface: An interface that analyzes the user's voice commands and emotional tone.

### Image Processing Interface: A system interface that analyzes the user's facial expressions.

### Biometric Data Interface: An interface that analyzes the biometric data the user receives from wearable devices (e.g. smart watches).

### API Interfaces: An interface that analyzes the data from the user in real time and provides instant feedback and suggestions by enabling the chatbot to integrate with external services (emotion analysis, biometric data collection, etc.).

### Database Interface: An interface where user data is stored and queried.

### User interfaces

The login screen should be large enough to enter the username and password. It should not be large enough to make clicking difficult for the user or to impair the readability of the interface. The buttons on the home page where the user will click on the desired data type should be of sufficient size and the font sizes should be readable. The home screen should provide users with easy access to the new conversation, previous chats, and the user page. The menu structure should be intuitive so that users can easily access the information they need. The main menu should provide users with quick access to the desired section. Error messages should provide clear and direct information to the user. The interface should allow users to perform functions quickly and effectively. The steps required for any operation should be minimized. Unnecessary complexity should be avoided and a user should be able to easily use this application with an average of 5 minutes of training. In addition, the interfaces should be designed in a way that the user can easily remember how to use the application.

### Hardware interfaces

This application is a mobile app and will be coded for Android. However, supported updates for IOS will be added later.

### Software interfaces

### User Interface

### Description: This is the interface where users interact with the chatbot. This interface receives various inputs from the user, such as voice, facial expressions, biometric data, and text input. It will be developed to have a user-friendly design, so that users can easily communicate with the chatbot.

### Voice Recognition Interface

### Description: This is an interface that analyzes the user's voice commands and emotional tone. This interface ensures that voice commands are correctly perceived and that user emotions are extracted from the tone of voice. Real-time processing of the user's voice is a critical component for instant responses.

### Image Processing Interface

### Description: This is a system interface that analyzes the user's facial expressions. This interface processes the necessary image data to determine the user's emotional state using face recognition and emotion analysis technologies. Real-time face recognition and emotion detection play an important role in the chatbot providing personalized feedback to the user.

### Biometric Data Interface

### Description: An interface that analyzes biometric data received from wearable devices (e.g. smartwatches). This interface collects and interprets information such as heart rate, stress levels, and other health data. Integration of biometric data allows for more accurate feedback by providing a deeper understanding of the user's emotional state.

### API Interfaces

### Description: An interface that analyzes data from users in real time and provides instant feedback and recommendations. This interface is used to provide sentiment analysis, biometric data collection, and integration with other external services. API interfaces enhance the functionality of the chatbot, enriching the user experience.

### Database Interface

### Description: An interface where user data is stored and queried. This interface securely stores users' past sentiments, interactions, and other important information, and provides quick access when needed. The database interface is critical for managing and analyzing user data.

### Communications interfaces

The application will not communicate with any other application.

### Memory constraints

Users' chats will be kept in the database for a certain period of time. Then the data will be permanently deleted from the database.

### Operations

Users can register and log in to the application.

They can choose what type of data to send.

They can change the data they entered when registering.

They can access and delete previous chats.

### Site adaptation requirements

**?**The application will initially support Android, but updates supporting IOS will be added later.

## Product functions

* In accordance with the purpose of the project, users should be able to access the application at any time they want. The application should be available 24/7.
* The application will analyze the users' emotional state and provide real-time feedback.
* The application should comply with ethical standards while providing advice on mental health and should be sensitive to the needs of the users.
* The application will initially be in English. (It can be multilingual later)
* Users can log in to the application by registering.
* Users can send data to the application with text, images and audio, biometric data.
* The user should be able to choose what type of data to send from the homepage. (only text, image and audio, etc.)
* Users can start new chats whenever they want and access and delete old chats from the user page.
* Users can update and delete the information they entered while registering to the application.
* With the user-friendly interface, the user's interface will be easy to learn and remember with the use of the application.
* Wearable devices or existing health applications will be used for biometric data integration.

## User characteristics

The intended users of SenseAI are primarily individuals seeking therapeutic assistance through a digital platform. Users are expected to have a general understanding of using mobile applications. No specific educational background is required; however, the app will be designed to cater to users with diverse educational levels, from high school students to professionals.

Given the nature of SenseAI as a therapy app using AI, users may have varying levels of comfort with AI interactions. Some users may feel hesitant or cautious about AI’s ability to understand emotional context. Consequently, the app will prioritize transparent communication about data handling, privacy, and the limitations of AI responses to build user trust and enhance user comfort

## Constraints

* The size and image resolution of the videos that users will send will be determined later by the developer team and will have certain limits. Users who try to add data with an unsupported feature will be shown an error message appropriate for the error.
* The text received from users will have a character limit that will be determined later. Users who exceed the limit will be shown an error message appropriate for the error.
* User data must be stored and processed in accordance with data privacy laws.
* Users' previous chats will continue to be kept in the database for a period to be determined during development.
* 95% of the responses returned to the user should be under 1 minute.
* In the first stage, 10 users will be supported, and this number will be increased to 100 users later.

## Assumptions and dependencies

User Device Specifications

Description: The characteristics of the devices that users use to access the software can affect the performance of the application. For example, optimization requirements may arise for devices with lower hardware specifications.

Network Conditions

Description: Users' Internet connection speeds and network conditions can affect the performance of the software. Conditions such as longer response times or data loss for low-bandwidth connections may require reconsideration of the requirements.

Audio and Visual Quality

Description: The audio and video quality delivered to users can significantly impact the user experience. Inadequate audio or video quality can make communication difficult to understand and can reduce the effectiveness of the application. Therefore, the requirements should be updated to ensure high-quality audio and video streaming.

## Apportioning of requirements

Additional Emotion Recognition Modes

Additional emotion recognition modes (e.g. stress management or relaxation techniques) added to better understand the user’s mood.

More Integration Options

A wider range of integrations with different wearable devices. For example, connection with more brands and models of smartwatches or fitness trackers.

Enhanced UI Features

Added more customization options in the UI design to enhance the user experience.

Multilingual Support

Added multilingual support features to enable the app to be used in different languages.

# Specific Requirements

This section of the SRS should contain all of the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output. As this is often the largest and most important part of the SRS, the following principles apply:

1. Specific requirements should be stated in conformance with all the characteristics described in 4.3 of the standards.
2. Specific requirements should be cross-referenced to earlier documents that relate.
3. All requirements should be uniquely identifiable.
4. Careful attention should be given to organizing the requirements to maximize readability.

## External Interface Requirements

This should be a detailed description of all inputs into and outputs from the software system. It should complement the interface descriptions in section 2 and should not repeat information there.

It should include both content and format as follows:

1. Name of item;
2. Description of purpose;
3. Source of input or destination of output;
4. Valid range, accuracy, and/or tolerance;
5. Units of measure;
6. Timing;
7. Relationships to other inputs/outputs;
8. Screen formats/organization;
9. Window formats/organization;
10. Data formats;
11. Command formats;
12. End messages.

### User interfaces

### Hardware interfaces

### Software interfaces

### Communications interfaces

## Functional Requirements

Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as "shall" statements starting with "The system shall."

These include

1. Validity checks on the inputs
2. Exact sequence of operations
3. Responses to abnormal situations, including
   1. Overflow
   2. Communication facilities
   3. Error handling and recovery
4. Effect of parameters
5. Relationship of outputs to inputs, including
   1. Input/output sequences
   2. Formulas for input to output conversion

It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does not imply that the software design will also be partitioned that way.

### User class 1

#### Feature 1.1

##### Introduction/Purpose of feature

##### Stimulus/Response sequence

##### Associated functional requirements

#### Feature 1.2

##### Introduction/Purpose of feature

##### Stimulus/Response sequence

##### Associated functional requirements

.

.

.

### User class 2

.

.

.

## Performance requirements

This subsection should specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole. Static numerical requirements may include the following:

1. The number of terminals to be supported;
2. The number of simultaneous users to be supported;
3. Amount and type of information to be handled.

Static numerical requirements are sometimes identified under a separate section entitled Capacity.

Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.

All of these requirements should be stated in measurable terms. For example,

*95%* *o*f *the* *transactions* *shall* *be* *processed* *in* *less* *than* 1 *s.*

rather than,

*An* *operator* *shall* *not* *have* *to* *wait* *for* *the* *transaction* *to* *be completed.*

NOTE-Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.

## Logical database requirements

This should specify the logical requirements for any information that is to be placed into a database. This may include the following:

1. Types of information used by various functions;
2. Frequency of use;
3. Accessing capabilities;
4. Data entities and their relationships;
5. Integrity constraints;
6. Data retention requirements.

## Design constraints

This should specify design constraints that can be imposed by other standards, hardware limitations, etc.

### Standards compliance

This subsection should specify the requirements derived from existing standards or regulations. They may include the following:

1. Report format;
2. Data naming;
3. Accounting procedures;
4. Audit tracing.

For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.

## Software system attributes

There are a number of attributes of software that can serve as requirements. It is important that required attributes be specified so that their achievement can be objectively verified.

### Reliability

This should specify the factors required to establish the required reliability of the software system at time of delivery.

### Availability

This should specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart.

### Security

This should specify the factors that protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to

1. Utilize certain cryptographical techniques;
2. Keep specific log or history data sets;
3. Assign certain functions to different modules;
4. Restrict communications between some areas of the program;
5. Check data integrity for critical variables.

### Maintainability

This should specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices.

### Portability

This should specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include the following:

1. Percentage of components with host-dependent code;
2. Percentage of code that is host dependent;
3. Use of a proven portable language;
4. Use of a particular compiler or language subset;
5. Use of a particular operating system.

# References

The resources used to produce this document should be listed here and referenced in the relevant text of this document

|  |  |
| --- | --- |
| [1] | Software Engineeering Department, "Graduation Projects," SENG, [Online]. Available: https://seng.cankaya.edu.tr/graduation-projects/. [Accessed 28 June 2024]. |