**Product Design**

|  |  |  |  |
| --- | --- | --- | --- |
| **Team:**  **Members:** | | Ctrl-Alt-Elite  Austin Hoffman,  Akash Patel,  Hadia Bilal,  Shawana Tahseen,  Timmy Akindunni | |
| ***Revision Number*** | ***Revision Date*** | | ***Summary of Changes*** | | ***Author(s)*** |
| *Initial* | *09/25/2023* | | *Initial Creation* | | Austin,  Akash,  Hadia,  Shawana, Timmy |

# Class Diagram(s)

A screenshot of a computer

Description automatically generated

# ER Diagram(s)

# A diagram of a computer Description automatically generated

# User Interface Wireframe(s)/Screenshot(s)

# \*Wireframe in Sprint 1\* ?

A screenshot of a login screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a login

Description automatically generated

Link Analysis:

A screenshot of a computer screen

Description automatically generated

# Design Summary

Our class diagram simple outline: (You can possibly use this when making the class diagram, it’s a simple idea overview)

Class Diagram:

User:

Attributes: userID, username, password, chatHistory, feelingsLog

Methods: login(), logout(), logFeeling(), viewHistory()

AIChatbot:

Attributes: chatLog, trainedModel

Methods: initiateChat(), analyzeResponse(), correlateEvents()

FeelingsLog:

Attributes: date, emotion, description

Methods: addLog(), viewLogs()

ChatHistory:

Attributes: chatID, chatDate, chatContent

Methods: storeChat(), retrieveChat()

We will need to think of more methods and variables and expand this throughout project development.

ER Diagram:

Entities:

User

ChatHistory

FeelingsLog

Relationships:

User and ChatHistory: One to Many (One user can have many chat histories)

User and FeelingsLog: One to Many (One user can have many logs)

Again, very simple and will need to be expanded.

# Design Rationale

**Considered Designs:**

**Cloud-based AI**: We initially considered leveraging cloud-based AI systems for the chatbot feature. However, this was ruled out in favor of local AI execution, emphasizing user privacy and data security.

**Application Development Environments**: We briefly considered using Unity or Xamarin Forms to create the application because of some of the features each tool possessed. We disregarded these different tools though because there exist better and more focused tools for our use case.

**Selected Design and Rationale:**

**Flutter Mobile Application:** Flutter was chosen for its cross-platform capabilities, allowing a single codebase to cater to both iOS and Android users. This optimizes development resources and ensures consistent user experience across devices. Flutter also runs on Dart, which is a language like C#, which simplifies a lot of the tediousness of working with C, a language all members of the team are familiar with.

Our decision to use flutter was also in part to Flutters ability to simplify UI creation. With Flutter, making intuitive and easy to use UI is much easier.

**Local AI execution:** Keeping the AI operations local ensures data privacy. Users will be more inclined to be honest and open if they know their sensitive data isn't leaving their device. Keeping the AI local also makes it so that our application does not rely on external servers, so the project will continue to work even without future lack of support.

**User Log Feature:** Allows users to actively record their feelings, fostering a sense of engagement and responsibility towards their mental health journey. We decided on this because it further encourages the use and appreciation of the apps correlation abilities. Beyond being able to just converse with the AI, you can also use the app to directly store your emotions. Using this feature in tandem with chatting with the AI, allows the user to see possible correlations between events from their day and how they are feeling.

**Simple AI Chatbot Design:** Instead of a complex, multifunctional chatbot, we opted for simplicity to ensure it's approachable, friendly, and doesn't overwhelm the user. Its primary function is listening and drawing insights, in line with our objective to provide users a virtual companion. This design choice decreases the broadness of the AI’s knowledge, but allows the AI to run on simpler devices, and to keep its responsibilities strict so it performs as best it can with its resources.

**Trade-offs and Considerations:**

Local AI execution might limit the computational capabilities compared to cloud-based solutions. However, we believe the privacy advantage far outweighs the potential drawbacks. Users will be sharing sensitive information, and we want to make sure they are absolutely assured their secrets are secret.

Simplicity of the AI may sometimes not capture all complexities of a conversation. We accepted this trade-off for the sake of user-friendliness, preventing misinterpretations that can arise from information hallucinations, and resource usage. By creating a simpler, more focused AI, older devices may be able to run the application because of the lowered device resource requirements. Making our AI model simpler also allows us to personally curate training data due to the smaller dataset needed for training, which will keep responses more helpful and reduce the chance of harmful responses.