**AI INFUSTED CHATBOT FOR TREATMENT**

**OF MENTAL ILLNESS**

1Anish. M 2Monesh. R 3Rajasudar. M 4Akash. R 5D. Meenakshi

1234 – UG – Computer Science and Engineering, Tagore Engineering College, Chennai

5 Associate Professor, Computer Science and Engineering, Tagore Engineering College, Chennai.

[a2anish03@gmail.com](mailto:a2anish03@gmail.com) | [monesh@gmail.com](mailto:monesh@gmail.com) | [rajasudar@gmail.com](mailto:rajasudar@gmail.com)

[akash@gmail.com](mailto:akash@gmail.com)

**ABSTRACT**

**Mental health is a crucial aspect of human well-being, yet many people face barriers to access professional help and support. Chatbots, powered by artificial intelligence and machine learning, offer a promising solution to provide information, guidance, and empathy to individuals seeking mental health assistance. AI and ML chatbot for mental health suggestion is a type of application that uses artificial intelligence and machine learning to provide users with a safe and confidential space to talk about their mental health concerns and receive support and guidance from a trained professional. We use ChatGPT which is a powerful language model that has been trained on a large dataset of human conversation and is able to generate human-like responses to user input. This makes ChatGPT an ideal choice for building a chatbot that can provide support and guidance to users seeking help with their mental health.**

**Keywords:** Artificial Intelligence, Machine Learning, Amazon Web Services, Docker, Containers, Elastic Cloud Computing, Elastic Container Service, Elastic Container Registry.

1. **INTRODUCTION**

A chatbot is a system that can converse and interact with human users using spoken, written, and visual languages. In recent years, chatbots have been used more frequently in various industries, including retail, customer service, education, and so on because of the advances in artificial intelligence (AI) and machine learning (ML) domains. Facebook Messenger currently offers more than 300,000 text-based

chatbots. Chatbots have primarily been used for commercial purposes and profitable businesses. However, more recent research

has demonstrated that chatbots have considerable promise in the health care industry in treating patients and offering them support in a cost-effective and convenient manner.

In the context of mental health (MH), chatbots may encourage interaction with those who have traditionally been reluctant to seek health-related advice because of stigmatization. Chatbots are an emerging technology that shows potential for mobile MH apps to boost user engagement and adherence. The effectiveness of chatbots has been explored for self-disclosure and expressive writing. Young people with MH issues have experienced various types of social

support such as appraisal, informational, emotional, and instrumental support from chatbots. In addition, chatbots have been designed to educate underprivileged communities on MH and stigmatized topics. Emerging evidence has shown user acceptance of chatbots for supporting various MH issues and early promises in boosting health outcomes in the physical and MH domains.

The adoption of new technology, especially those heavily related to AI and ML, relies first on ascertaining the levels of safety, effectiveness, and user comfort. Despite the increasing adoption and benefits of emerging technologies such as chatbots to support MH and well-being, little research has been conducted to gain an understanding of consumers’ real-life user experiences of interacting with MH chatbot apps. Recent research on MH apps in general points out that patient safety is rarely examined, health outcomes are evaluated on a small scale, and no standard evaluation methods are present, and these findings also apply to MH chatbot apps. Similar to many other emerging technologies, recent developments in chatbots are because of a massive technology push, with little attention paid to human needs and experiences. This can lead to unintended negative consequences, such as biases, inadequate and failed responses, and privacy issues, all of which can negatively affect the quality of the experience of chatbots as a source of support. Thus, it is critical to gain an understanding of the nuances in users’ perceptions and experiences of using MH chatbots.

Commercially available MH chatbot apps for popular platforms are used by a large user base with varying demographic backgrounds. These users can provide feedback through ratings and text reviews. These platforms can be leveraged to gain a holistic understanding of the features that recently developed MH chatbots offer and how users assess them. Knowledge of user perceptions from real-life experiences can inform future research and the design of more effective chatbots. Previous studies have identified user reviews as a great source for understanding the benefits and drawbacks of technology. This allows

researchers to incorporate community values and needs into product design and improves user-friendliness. Consumers often make decisions about using new tools based on user rating scores and reviews in web-based marketplaces. According to previous studies, users trust reviews and feel at ease based on their decisions them. Moreover, previous literature emphasizes analyzing user reviews of mobile MH apps that have chatbot features to obtain in-depth knowledge about this new technology intervention in mobile MH apps. For this study, we decided to analyze commercially available well-known chatbot-based mobile MH apps and their corresponding user reviews from the Apple App Store and Google Play Store. To obtain a comprehensive overview of these apps and understand the nuances of user opinions, we aimed to answer the following 2 research questions (RQs):

RQ1: What are the state-of-the-art features and properties of chatbot-based mobile MH apps?

RQ2: What concerns and opinions are expressed in user reviews published on web-based app store platforms regarding the usability and efficiency of chatbot-based mobile MH apps?

We conducted an exploratory observation of 10 apps that offer support and treatment for a variety of MH concerns with a built-in chatbot feature and qualitatively analyzed their user reviews available on the Google Play Store and Apple App Store. Publicly available data (user reviews) provide in-depth analyses of consumers’ personal app user experiences. We found that although chatbots’ personalized, humanlike interactions were positively received by users, improper responses and assumptions about the personalities of users led to a loss of interest. As chatbots are always accessible and convenient, users can become overly attached to them and prefer them over interacting with their friends and family members. Furthermore, a chatbot may offer support for a crisis whenever the user needs it because of its 24/7 availability, but even the recently developed chatbots

lack the understanding of properly identifying a crisis. Chatbots in this study fostered a judgment-free environment and helped users feel more comfortable sharing sensitive information.

Before implementing a technological solution for MH, researchers in digital health communities are constantly interested in the support needs and preferences of groups or communities. Researchers have analyzed the effectiveness of technologies used for MH assistance, proposing ethical concerns, policy recommendations, and designing automated or human-in-the-loop interactive systems. These studies stressed the significance of designing and evaluating systems for susceptible populations, such as people with MH issues, from the perspective of users. To contribute to this body of work, we discussed our study’s findings with respect to the research and design implications for future MH chatbots. We outlined specific recommendations for customizing certain features, careful consideration of incorporating persuasive strategies, and trust building. Finally, we discussed the impact of excessive reliance on chatbots for MH support. We believe that considering these insights while developing a chatbot-based MH support system will make the design user centric and, thus, more effective.

1. **EXISTING SYSTEM**

The existing systems for chatbots designed for the treatment of mental illness vary widely in terms of complexity, functionality, and effectiveness.

**Rule-Based Chatbots:** These chatbots use a set of predefined rules to respond to user inputs. Disadvantages include limited ability to understand natural language and lack of adaptability to new or complex situations.

**Scripted Chatbots:** Scripted chatbots use predefined scripts to simulate conversation. While they can provide useful information, they lack the ability to engage in meaningful dialogue or provide personalized responses.

**Machine Learning-Based Chatbots:** These chatbots use machine learning algorithms to improve their responses over time. However, they may require large amounts of training data and ongoing maintenance to remain effective.

**Hybrid Chatbots:** Hybrid chatbots combine rule-based and machine learning approaches to improve their performance. However, they can be complex to develop and may still struggle with complex or nuanced conversations.

**3.1 DISADVANTAGES OF EXISTING SYSTEM:**

There are several drawbacks associated with the current system:

**Ethical and Privacy Concerns:** All chatbots for mental health must address concerns about privacy, confidentiality, and the potential for harm. Ensuring advice is crucial.

**Limited Understanding of Context:** Many existing chatbots struggle to understand the context of a user's conversation, leading to responses that may be irrelevant or confusing.

**Lack of Emotional Intelligence:** While some chatbots attempt to simulate empathy, they often fall short in providing the kind of emotional support that a human therapist can offer.

**Dependency on Internet Connection:** Most chatbots require an internet connection to function, which may limit their accessibility in areas with poor connectivity.

1. **PROPOSED SYSTEM**

The proposed system for the chatbot for the treatment of mental illness combines several key components to deliver effective and user-friendly support. The frontend interface will be developed using Python Gradio, providing an intuitive platform for users to interact with the chatbot. The backend, also implemented in Python, will manage the communication between the frontend and OpenAI's powerful GPT-3.5 Turbo model. This model, trained on a vast dataset, will generate responses to user inputs, enabling the chatbot to engage in meaningful and empathetic conversations. The entire system will be deployed on the AWS cloud using Docker containers, ensuring scalability, reliability, and security. This architecture leverages the strengths of each component to create a chatbot that offers personalized and effective mental health support.

**4.1 ADVANTAGES OF PROPOSED SYSTEM:**

**Effective Support:** The use of OpenAI's GPT-3.5 Turbo model allows the chatbot to provide personalized and effective mental health support, enhancing the user experience.

**User-Friendly Interface:** The frontend interface developed using Python Gradio provides an intuitive platform for users to interact with the chatbot, making it more accessible and easier to use.

**Scalability:** Deploying the entire system on the AWS cloud using Docker containers ensures scalability, allowing the chatbot to handle a large number of users and adapt to changing demands.

**Reliability:** The use of Docker containers and AWS cloud ensures reliable performance, with high availability and minimal downtime.

**Security:** Deploying on AWS cloud provides robust security features, protecting user data and ensuring compliance with data protection regulations.

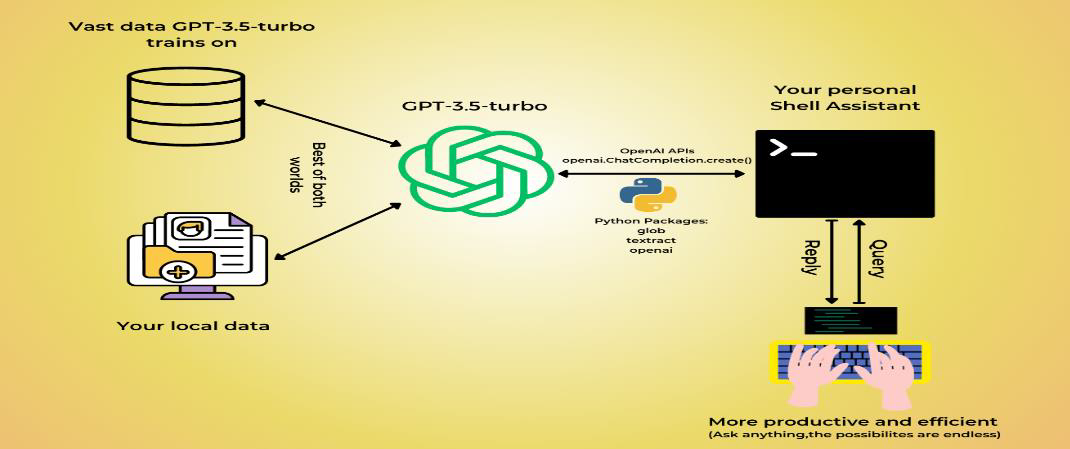


Figure 4.1 Proposed System Model

1. **SYSTEM DESIGN**

The proposed chatbot system for the treatment of mental illness integrates several key components to provide personalized and empathetic support to users. The frontend interface, developed using Python Gradio, offers a user-friendly platform for interactions. The backend, implemented in Python, manages communication with OpenAI's GPT-3.5 Turbo model, which generates responses based on user inputs. Deployment on the AWS cloud using Docker containers ensures scalability and reliability. This system architecture leverages the strengths of each component to create a chatbot that delivers effective and user-friendly mental health support.

* 1. **PYTHON FRONTEND (GRADIO):**

The Frontend Interface implemented using Gradio for the mental health chatbot offers a highly interactive and user-friendly experience, leveraging the platform's capabilities for rapid development of machine learning interfaces. Here's a breakdown of its key components:

**Input Elements:** Gradio allows the inclusion of various input elements such as text boxes, dropdowns, and sliders, enabling users to input their queries or select options in a structured manner. These elements are designed to be intuitive and easy to use, enhancing the overall user experience.

**Output Display:** The interface includes a designated area to display the chatbot's responses. These responses are generated by the backend AI model and are presented in a format that is clear and easy to understand, ensuring effective communication between the user and the chatbot.

**Dynamic Interaction:** Gradio's dynamic interface features enable real-time updates and interactions. For example, as users type their queries, the chatbot may suggest completions or provide immediate feedback, enhancing the conversational flow.

**Customization Options:** Gradio offers a range of customization options for the interface's appearance and behavior. This includes the ability to customize colors, fonts, and layouts to match the chatbot's branding or to create a visually appealing interface that aligns with the chatbot's purpose.

**Accessibility Features:** Gradio interfaces can be designed to be accessible to users with disabilities. This includes support for screen readers, keyboard

navigation, and other accessibility features to ensure that all users can engage with the chatbot effectively.

**Error Handling:** The interface includes robust error handling mechanisms to manage situations where the chatbot may not understand a query or encounter other issues. Clear error messages and suggestions for rephrasing queries help users navigate these situations smoothly.

* 1. **PYTHON BACKEND (API CALL) :**

The Backend module of the proposed chatbot system for mental health support manages the communication between the Frontend Interface and OpenAI's GPT-3.5 Turbo model, facilitating the exchange of messages and responses. It consists of several key components:

**Input Processing:** Upon receiving a user input from the Frontend Interface, the Backend module processes the input to remove any unnecessary characters or API call.

**ChatGPT API Integration:** The Backend module interacts with the ChatGPT API to send the processed user input and receive a response from the AI model. This integration is crucial for leveraging the AI model's capabilities in generating meaningful and empathetic responses.

**Response Handling:** Once the Backend receives a response from the ChatGPT API, it processes the response to format it appropriately for display in the Frontend Interface. This may involve adding formatting, such as line breaks or emojis, to make the response more visually appealing and easier to read.

**Error Handling:** The Backend module includes error handling mechanisms to manage any issues that may arise during the API call or response processing. This ensures that the chatbot remains functional and responsive even in the event of API errors or other issues.

**Feedback Loop:** The Backend module may include a feedback loop mechanism to collect feedback from users on the chatbot's responses. This feedback can be used to improve the chatbot's performance over time, making it more effective in providing mental health support.

* 1. **FINE-TUNE AN OPENAI CHATGPT MODE:**

The ChatGPT 3.5 Turbo module is a key component of the proposed chatbot system for mental health support, serving as the AI model responsible for generating responses to user inputs. This module is based on OpenAI's GPT-3.5 architecture, which is a state-of-the-art language model known for its ability to understand and generate human-like text.

the following steps to build a fine-tuned model

* Preparing the training data
* Uploading the training data to OpenAI servers
* Creating a fine-tuned model with the training data

**5.3.1 PREPARING THE TRAINING DATA**

The fine-tuning process typically begins with a dataset that is carefully curated and labeled, and it involves raining the model on this dataset using techniques such as transfer learning. The model’s parameters are adjusted during fine-tuning to make it more accurate and contextually appropriate to generate responses in the target domain. The model can acquire domain-specific knowledge, language patterns, and nuances by fine-tuning, enabling it to generate more relevant and coherent responses for specific applications or use cases.

During the fine-tuning process, it is necessary to provide the training data as a JSON file. You should create a diverse set of target conversations that expect the model to perform after the fine-tuning process is completed.

This training data can come from a variety of sources, such as books, articles, specialized datasets, or we can prepare it manually.

**Training Data Format**

The training data should be in conversational chat format and it is required to fine-tune *gpt-3.5-turbo*. For example, for our usecase, we have created the *input\_data.jsonl* which has the required pattern. Note that each message appears in a new line.

{"messages": [{"role": "user", "content": "Reset my password for account ID 12345"}, {"role": "assistant", "content": "Your password has been reset. Please check your email for the new login details."}]}

{"messages": [{"role": "user", "content": "How do I update my billing information?"}, {"role": "assistant", "content": "To update your billing information, log in to your account, go to the 'Billing' section, and follow the prompts."}]}

To start the fine-tuning a model, we must provide at least 10 examples. For optimal results surpassing base models, we should aim to provide a few hundred or more high-quality examples, preferably vetted by human experts. The right number of examples varies on the exact use case and the level of required accuracy.

**5.3.2 UPLOADING THE TRAINING DATA TO OPENAI SERVERS:**

The next step towards fine-tuning a model is to upload the training data file to OpenAI server using the **“client.files.create()”** API.

After the file is uploaded, a file ID is generated that we can refer to every time we need without uploading the file again and again.

**“File has been uploaded to OpenAI with id file EVvrt7QWovkOH4gz2NdrC20U”**

After we upload the file, it may take some time to process. While the file is processing, we can still create a fine-tuning job but it will not start until the file processing has completed.

The maximum file upload size is 1 GB, though we do not suggest fine-tuning with that amount of data since you are unlikely to need that large of an amount to see improvements. 43

**5.3.3 FINE-TUNING A MODEL WITH THE UPLOADED TRAINING DATA**

After the file has been uploaded to the OpenAI server, we can use the **“client.fine\_tuning.jobs.create()”** API to fine-tune the selected chatGPT model with the supplied training data.The API returns a Job ID which refers to an asynchronous Job that has been created on the backend.

**“Fine Tune Job has been created with id ftjob-vhLTPOwAW1rSQT42vZLeXxtH”**

It’s important to note that the process of creating the model may vary in duration, taking anywhere from a few minutes to hours. The OpenAI servers will continue processing your fine-tuned model until it reaches completion.

We can enquire about the progress of the fine-tuning job using the “**client.fine\_tuning.jobs.list\_events()”** API that returns the latest N statuses related to the

job.Once the job has been completed, we can get the ID of the fine-tuned model in the final status message:

**“{FineTuningJobEvent(id='ftevent-OqhBL7WqEkkYTQqFCjGS1BSu', created\_at=1702289919,level='info', message='New fine-tuned model created:ft:gpt-3.5-turbo-0613:personal::8UXexX8R', object='fine\_tuning.job.event',data={}, type='message')”}**

* 1. **DEPLOYMENT IN CLOUD**

**AWS (AMAZON WEB SERVICES):**

**Purpose:** AWS provides a scalable and reliable cloud computing platform for deploying and managing applications. It offers a wide range of services that facilitate the deployment and operation of containerized applications.

**Functionality:** AWS services such as Amazon Elastic Container Service (ECS) or Amazon Elastic Kubernetes Service (EKS) can be used to deploy and manage Docker containers. These services provide features like automatic scaling, load balancing, and monitoring, making it easier to deploy and operate containerized applications on AWS.

**DOCKER CONTAINERS:**

**Purpose:** Docker containers package the chatbot system and its dependencies into a standardized unit that can run on any platform where Docker is installed, ensuring consistency across different environments.

**Functionality:** Docker containers provide an isolated environment for the chatbot system to run, ensuring that it is not affected by changes or issues in the underlying infrastructure. Docker images are used to create containers, which can then be deployed on AWS using services like ECS or EKS.

**DEPLOYMENT PROCESS:**

**Build Docker Image:** The chatbot system and its dependencies are packaged into a Docker image using a Dockerfile.

**Push Image to Docker Registry:** The Docker image is pushed to a Docker registry, such as Docker Hub or Amazon Elastic Container Registry (ECR), where it can be accessed by AWS services.

**Create ECS/EKS Cluster:** A cluster is created on ECS or EKS to run the Docker containers. The cluster provides the underlying infrastructure for running and managing the containers.

**Deploy Containers:** The Docker containers are deployed to the ECS or EKS cluster, where they are started and managed by the platform. The platform handles tasks like scaling, load balancing, and monitoring of the containers.

**Access the Chatbot:** Once deployed, the chatbot is accessible over the internet, and users can interact with it through the frontend interface.

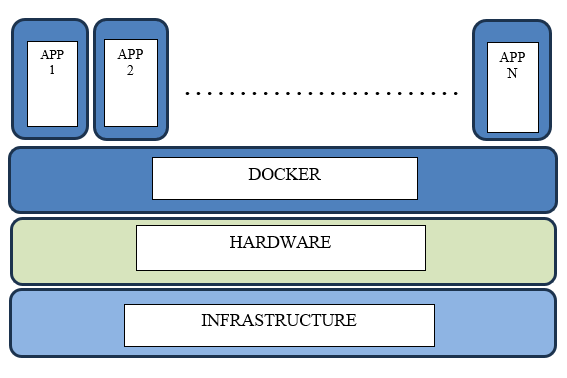
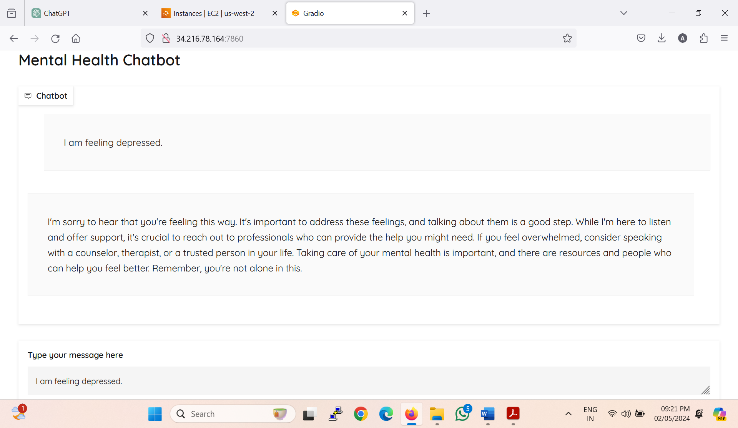


Figure 5.4.1 Docker Deployment

1. **RESULT**

****

Our chatbot for mental health treatment, with a frontend in Python Gradio for user interaction and a backend in Python managing communication with OpenAI's GPT-3.5 Turbo model. Deployed on AWS using Docker containers, I aim to provide personalized and effective support for mental health issues.

1. **CONCLUSION**

In conclusion, the integrated system for the mental health chatbot demonstrates a sophisticated architecture that harmoniously blends frontend, backend, and AI components. The Python Gradio frontend stands out for its user-friendly interface, enabling individuals to interact with the chatbot intuitively.

On the backend, Python orchestrates the seamless communication between the frontend and the GPT-3.5 Turbo model from OpenAI. This model, renowned for its vast dataset and language understanding capabilities, plays a pivotal role in generating responses that are not only accurate but also empathetic, enhancing the user's sense of engagement and support.

The deployment on AWS through Docker containers underscores the system's scalability, reliability, and security. Leveraging AWS's infrastructure ensures that the chatbot can handle varying loads and maintain high availability, critical for providing uninterrupted mental health support.

Overall, this meticulously designed architecture maximizes the strengths of each component to create a chatbot system that offers personalized, effective, and empathetic mental health support, thereby marking a significant milestone in the application of AI for mental wellness.

1. **REFERENCES**
2. Adamopoulou E, Moussiades L. An overview of chatbot technology. Proceedings of the 16th International Conference on Artificial Intelligence Applications and Innovations; AIAI '20; June 5-7, 2020; Neos Marmaras, Greece. 2020.
3. Khan S, Rabbani MR. Artificial intelligence and NLP -based chatbot for Islamic banking and finance. Int J Inf Retr Res. 2021.
4. Cui L, Huang S, Wei F, Tan C, Duan C, Zhou M. SuperAgent: a customer service chatbot for e-commerce websites. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics-System Demonstrations; ALC '17; July 30-August 4, 2017; Vancouver, Canada. 2017.
5. Winkler R, Söllner M. Unleashing the potential of chatbots in education: a state-of-the-art analysis. Proceedings of the 78th Academy of Management Annual Meeting; AOM '18; August 10-14, 2018; Chicago, IL, USA. 2018.
6. Bhirud N, Tataale S, Randive S, Nahar S. A literature review on chatbots in healthcare domain. Int J Sci Res. 2019 Jul;8(7):225–31.
7. Lee YC, Yamashita N, Huang Y. Designing a chatbot as a mediator for promoting deep self-disclosure to a real mental health professional. Proc ACM Hum Comput Interact. 2020 May 29.
8. Oh YJ, Zhang J, Fang M, Fukuoka Y. A systematic review of artificial intelligence chatbots for promoting physical activity, healthy diet, and weight loss. Int J Behav Nutr Phys Act. 2021 Dec 11.
9. Lee M, Ackermans S, van As N, Chang H, Lucas E, IJsselsteijn W. Caring for Vincent: a chatbot for self-compassion. Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems; CHI '19; May 4-9, 2019; Glasgow, Scotland, UK. 2019.
10. Abd-Alrazaq AA, Alajlani M, Alalwan AA, Bewick BM, Gardner P, Househ M. An overview of the features of chatbots in mental health: a scoping review. Int J Med Inform. 2019 Dec.