MindSpark

Project Supervisor

Dr. Syed Khaldoon Khurshid

Project Member

Ammad Aslam (Project Lead)	2021-CS-67
Mutaiba Mohsin	2021-CS-63
Fatima Muskan	2021-CS-78
Uswa Arif	2021-CS-77

Department of Computer Science

University of Engineering and Technology, Lahore

Pakistan

Abstract

MindSpark is an AI-powered diagnostic assistant designed to enhance the efficiency and accuracy of psychiatric evaluations. By integrating advanced Natural Language Processing (NLP) techniques with DSM-5 guidelines, MindSpark enables psychiatrists to analyze patient symptoms in real-time, whether provided through speech or text. The system automates the documentation process, generates diagnostic suggestions, and provides recovery recommendations, thereby reducing the reliance on manual note-taking.

MindSpark features a user-friendly interface that supports secure data handling in compliance with medical privacy standards like HIPAA. It visualizes symptom similarities with DSM-5 disorders, allowing psychiatrists to make informed decisions during patient sessions. The platform is cloud-based, ensuring scalability, accessibility, and seamless integration with existing clinical workflows.

By combining cutting-edge AI with clinical expertise, MindSpark aims to empower mental health professionals, improve patient care, and set a new standard for AI-driven solutions in mental health diagnostics.

Declaration

I declare that the Project entitled “MindSpark” submitted by G-26 has been carried out under the supervision of Dr. Khaldoon Khurshid. This document has been prepared under his guidance and has not been submitted elsewhere.

We will be solely responsible if any kind of plagiarism is found.:

Signed:
Date:

Acknowledgments

We extend our deepest gratitude to our project supervisor, **Dr. Syed Khaldoon Khurshid**, for his invaluable guidance, support, and encouragement throughout this project. His expertise and mentorship were instrumental in the successful completion of this work.

We would also like to acknowledge **Sir Waseem**, whose teachings in the Artificial Intelligence course provided us with a strong foundation in AI concepts and techniques, many of which were applied in this project.

Our sincere thanks go to the Department of Computer Science at the University of Engineering and Technology, Lahore, for providing the resources, facilities, and platform necessary for this project.

Finally, we are grateful to everyone who contributed to this project, directly or indirectly, through their guidance, support, and encouragement.

Thank you all for your support and belief in our work.

Contents

List of Tables

List of Figures

1 Introduction

1.1 Purpose

An AI-powered diagnostic support tool to assist psychiatrists during patient sessions. By leveraging advanced AI techniques, the DSM-5 guidelines and comparing it with the Benchmark (GPT-session), the system will analyze patient behaviors, speech patterns, and symptoms observed in real-time, helping psychiatrists document and interpret clinical findings more efficiently. This tool will assist in the diagnostic process by automatically capturing key observations, generating potential diagnoses, and offering recovery recommendations, thereby reducing the reliance on manual note-taking and improving diagnostic accuracy. The project seeks to enhance the quality and efficiency of mental health assessments, empowering psychiatrists with AI-driven insights and support during patient evaluations.

1.2 Scope

- Real-time documentation and analysis of patient behavior, speech patterns, and symptoms using AI and NLP techniques.
- Integration of DSM-5 guidelines for generating diagnostic suggestions and recovery plans based on clinical observations.
- Assistance in reducing manual note-taking by automatically capturing and organizing session data.
- Providing psychiatrists with a user-friendly interface to review, modify, and approve diagnostic recommendations.
- Secure handling and storage of patient data in compliance with medical standards and privacy regulations.
- Ability to generate comprehensive diagnostic reports and summaries at the end of each session.
- Feature to maintain patient history for future reference and longitudinal analysis.

1.3 Definitions, Acronyms, and Abbreviations

- **NLP:** Natural Language Processing – AI technology that processes and understands human language, enabling the analysis of patient speech during psychiatry sessions.
- **DSM-5:** Diagnostic and Statistical Manual of Mental Disorders, 5th Edition – The standard classification of mental disorders used by psychiatrists for diagnostic purposes.
- **ML:** Machine Learning – A subset of AI that allows systems to learn from data, crucial for recognizing patterns in patient behavior and symptoms.
- **Compact LLM:** Compact Large Language Model – A scaled-down version of large language models designed for real-time processing in resource-constrained environments like psychiatry sessions.
- **Speech-to-Text (STT)**: Technology that converts spoken language into written text, facilitating automatic transcription of psychiatry sessions.
- **Psychiatric Assessment:** The process of collecting and analyzing patient behavioral, cognitive, and emotional data to inform diagnosis.
- **HIPAA:** Health Insurance Portability and Accountability Act – Regulations ensuring the confidentiality of patient data, relevant for handling sensitive information in psychiatry.

2 System Requirement Specification

2.1 Product Perspective

The system operates within the context of mental health care, specifically assisting psychiatrists during patient sessions. It integrates with existing clinical workflows by providing real-time diagnostic support based on patient input and the DSM-5 guidelines. The system depends on reliable internet connectivity to function optimally and requires access to audio and text input from patients.

2.2 Product Functions

- **User Registration:** Allow psychiatrists to create secure accounts for accessing the system.
- **Anonymous User Access:** Enable patients to provide input without the need for registration, preserving privacy.
- **Diagnosis Generation:** Analyze patient inputs in real-time and suggest possible diagnoses based on DSM-5 criteria.
- **Report Viewing and Generation:** Create comprehensive reports summarizing session details, findings, and suggested diagnoses.
- **Session History Management:** Allow psychiatrists to access past reports and patient history to inform ongoing treatment.
- **User Authentication and Security:** Ensure secure login for registered users and safeguard sensitive patient data.
- **Cloud Deployment:** Utilize a cloud service for hosting the application, ensuring scalability and accessibility.

2.3 User Classes and Characteristics

The system caters to the following user classes:

- **Psychiatrists (Registered Users):** Healthcare professionals who will use the system to manage patient sessions, generate diagnoses, and view reports. They require secure access and comprehensive functionality.
- **Psychiatrists (Anonymous Users):** Individuals seeking mental health support who can interact with the system without creating an account. They have limited access and input capabilities.
- **Support Staff/Admin (optional):** Assistants who may assist psychiatrists in managing patient records or navigating the system, requiring limited access to patient data.

2.4 Operating Environment

The system will run as a web application hosted on a cloud platform (e.g., AWS, Google Cloud, or Azure) to ensure high availability and scalability. It will be accessible via modern web browsers on desktops and mobile devices, enabling psychiatrists to conduct sessions from various locations while maintaining patient confidentiality.

2.5 Design and Implementation Constraints

Key constraints for the system include:

- **Security Requirements:** Compliance with health data protection regulations (e.g., HIPAA) to ensure patient data confidentiality and security.
- **Compliance with DSM-5:** All diagnostic suggestions and functionalities must align with the DSM-5 guidelines to ensure clinical relevance and accuracy.

- **Performance Limits:** The system should deliver real-time responses and maintain low latency, especially during live sessions.
- **Resource Constraints:** The Compact LLM must operate efficiently on devices with varying processing capabilities, particularly in clinical settings with limited resources.
- **Cloud Dependency:** The system's operation is contingent on stable cloud services, requiring reliable internet connectivity.

2.6 Assumptions and Dependencies

The system operates under the following assumptions:

- Users will have reliable internet access for optimal system performance.
- Psychiatrists will have basic proficiency in using web-based applications and mobile apps.
- The system may rely on third-party APIs for functionalities such as speech recognition or data analytics.
- Users will provide accurate and honest information during their interactions with the system to ensure effective diagnostics.

3 Functional Requirements

3.1 User Registration and Authentication

- The system must allow psychiatrists to create secure accounts with unique usernames and passwords.
- The system must enable users to reset passwords and recover accounts through email verification.
- The system must provide anonymous access for psychiatrists or patients, allowing them to provide input without registration.

3.2 Patient Input Management

- The system must enable patients to input their symptoms through text or speech.
- The system must provide options for patients to upload additional files or documents relevant to their mental health.
- The system must include a speech-to-text feature to convert spoken patient input into text format.

3.3 Diagnosis Generation

- The system must analyze patient inputs based on the DSM-5 guidelines and generate potential diagnoses.
- The system must provide detailed explanations of the generated diagnoses, including relevant symptoms and criteria.
- The system must allow psychiatrists to review and edit diagnosis suggestions before finalizing them.

3.4 User Interface

- The system must provide an intuitive and user-friendly interface for both psychiatrists and patients.
- The system must allow for easy navigation between different functionalities, such as input management, diagnosis generation, and report viewing.
- The system must be responsive and accessible on both desktop and mobile devices.

3.5 Report Generation and Management

- The system must automatically generate comprehensive reports summarizing the patient's input, findings, and suggested diagnoses.
- The system must allow psychiatrists to view, download, and print reports in PDF or other formats.
- The system must maintain a history of past reports for each patient, accessible by the psychiatrist.

3.6 Data Security and Privacy

- The system must ensure that all patient data is encrypted both in transit and at rest to protect sensitive information.
- The system must implement role-based access control to restrict access to patient data based on user roles (e.g., psychiatrists, administrators).
- The system must comply with relevant data protection regulations (e.g., HIPAA) to safeguard patient privacy.

3.7 Cloud Deployment

- The system must be designed to deploy on a cloud infrastructure to ensure high availability, scalability, and reliability.
- The system should facilitate automated backups and disaster recovery processes within the cloud environment.

4 Non Functional Requirements

4.1 Performance Requirements

- Response Time: The system must provide responses to user inputs within 3 seconds to ensure a smooth user experience during patient sessions.
- Throughput: The system should support at least 50 concurrent users without degradation in performance, accommodating multiple psychiatrists and patients accessing the system simultaneously.
- Scalability: The system must be designed to scale horizontally, allowing for the addition of resources (e.g., servers) as user demand increases.

4.2 Security Requirements

- Data Encryption: All sensitive patient data must be encrypted both in transit (using HTTPS) and at rest (using industry-standard encryption algorithms).
- Authentication: The system must implement strong authentication mechanisms, including multi-factor authentication for psychiatrists to ensure secure access.
- Access Control: Role-based access control must be enforced to restrict access to patient data based on user roles (e.g., psychiatrists, support team/Admin).

4.3 Usability Requirements

- User Interface Design: The system must have an user interface that is easy to navigate, ensuring that users with varying levels of technical proficiency can effectively use the system.

- Help and Documentation: The system must provide comprehensive user documentation and help resources, including tutorials and FAQs, to assist users in understanding the system's functionalities.

4.4 Reliability Requirements

- Availability: The system must have a 99.9 percent uptime, ensuring it is available for use by psychiatrists and patients most of the time, even during maintenance periods.
- Fault Tolerance: The system must be designed to handle unexpected errors gracefully, providing users with meaningful error messages and ensuring that data is not lost during failures.
- Backup and Recovery: The system must implement regular automated backups of patient data and allow for quick recovery in the event of data loss or system failure.

4.5 Compliance Requirements

- Regulatory Compliance: The system must comply with relevant regulations such as HIPAA (Health Insurance Portability and Accountability Act) to protect patient confidentiality and privacy.
- Data Retention Policies: The system must adhere to data retention policies, ensuring that patient data is stored for the required duration and disposed of securely when no longer needed.

4.6 Environmental Requirements

- Network Requirements: The system must require a minimum internet bandwidth of 5 Mbps for optimal performance, ensuring seamless communication between the cloud server and client devices.

5 System Architecture Design

5.1 Use Case Diagram

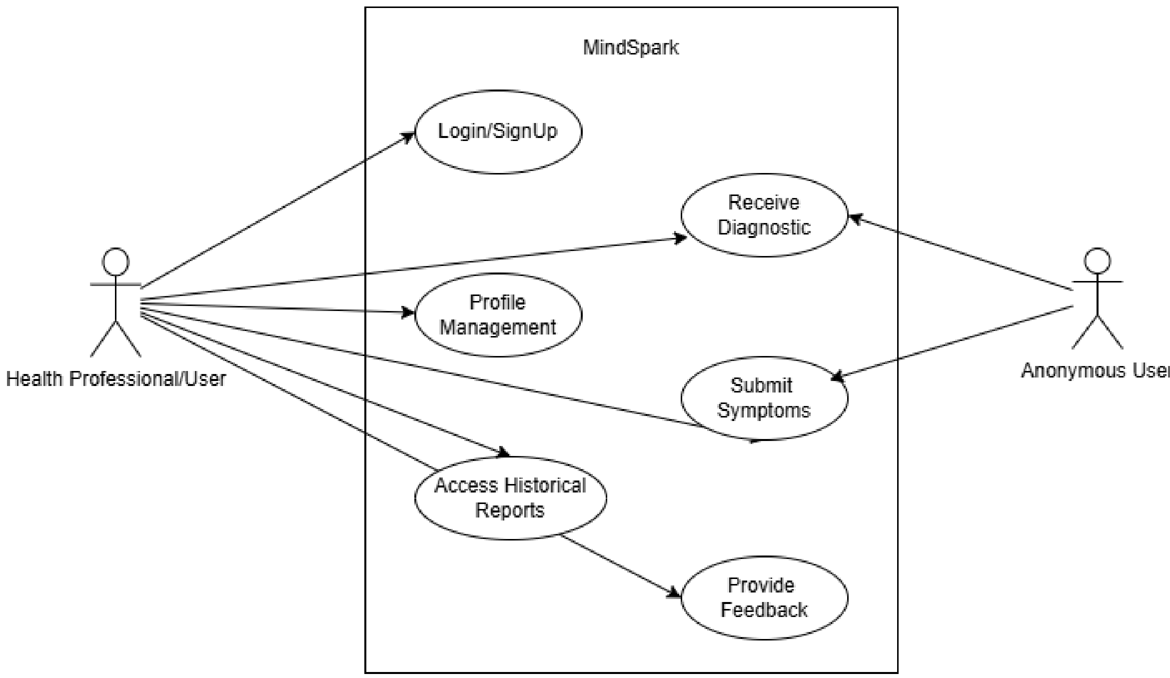


Figure 1: Use Case Diagram

The diagram illustrates the Use Case Diagram for our system involving three main actors: the Anonymous User, Mental Health Professional, and Admin. The system allows Anonymous Users to submit their symptoms and receive diagnostic reports, with the option to convert speech to text using an NLP and Speech-to-Text API. Mental Health Professionals can access historical reports, provide expert feedback, and log in or sign up to the system. The system leverages Generative AI to generate diagnoses based on user-submitted symptoms. Finally, the Admin is responsible for managing patient data within the system. This diagram highlights the interactions between users and system features, with external integrations such as AI and APIs enhancing its functionality.

5.2 Class Diagram

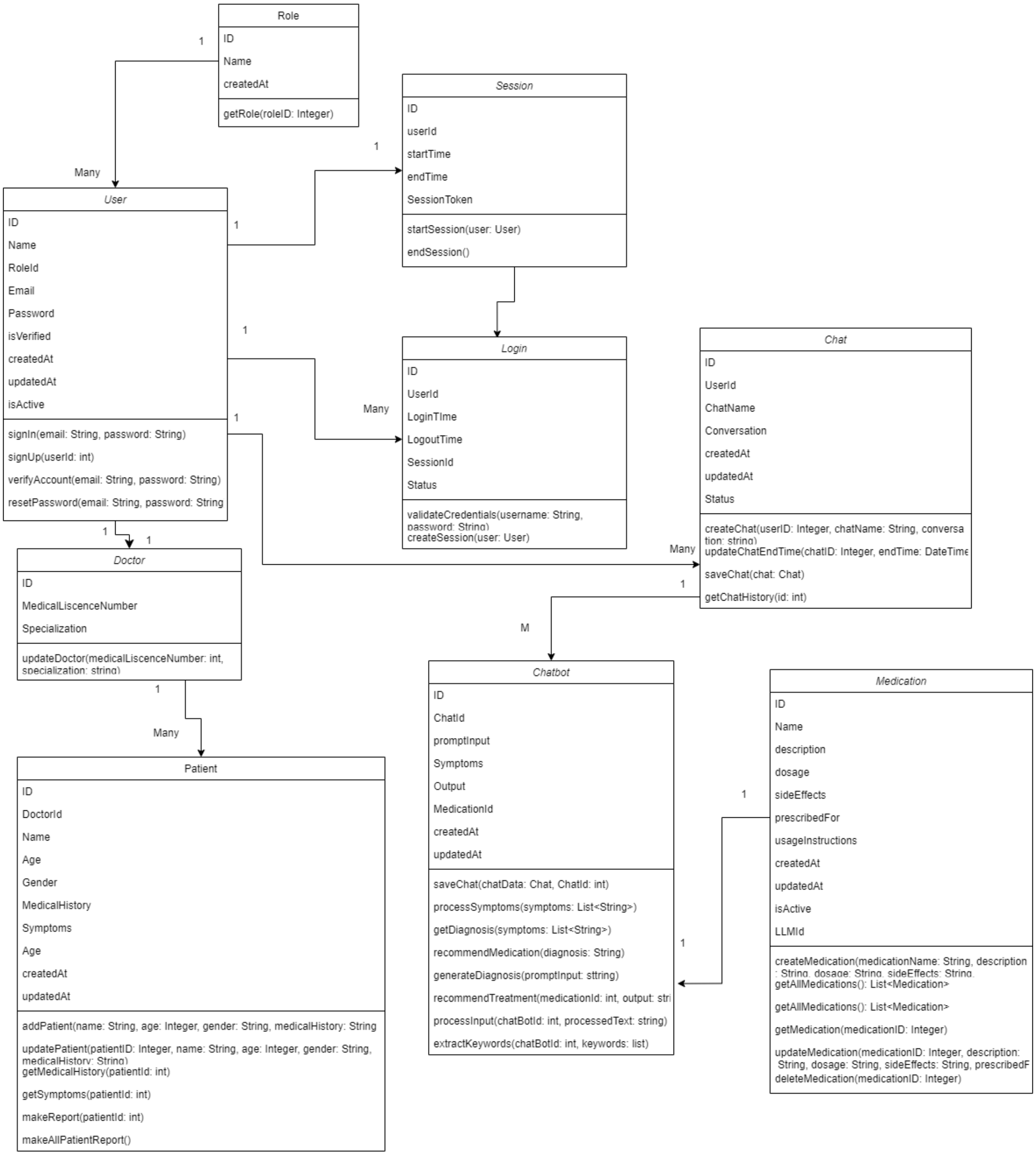


Figure 2: Class Diagram

5.3 Activity Diagram

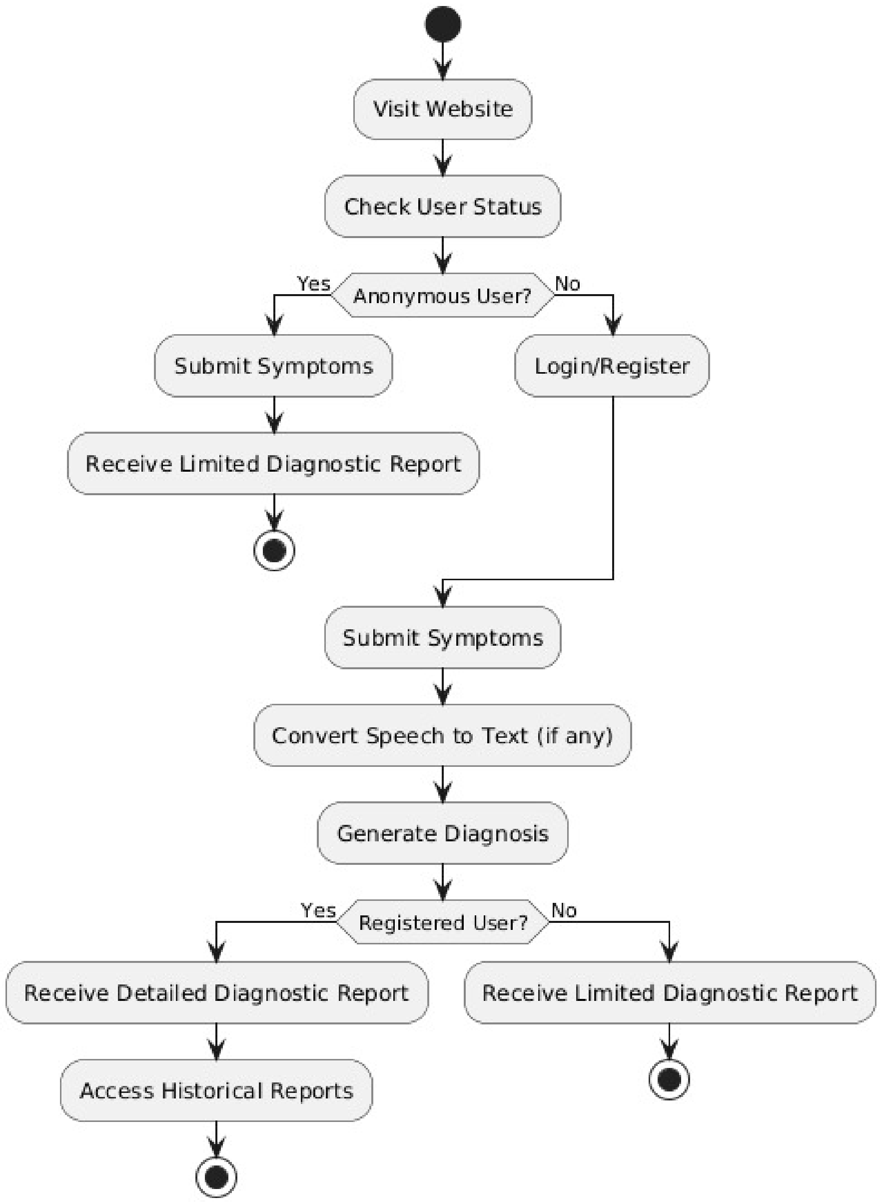


Figure 3: Activity Diagram

Diagram illustrates the flow of activities for users interacting with the system to submit symptoms and receive a diagnosis. The process begins when the user visits the website, where the system checks their user status, determining if they are anonymous or logged in. If the user is anonymous, they are directed to either submit their symptoms for a limited diagnostic report or register/login for further access. Once logged in, the user can submit their symptoms, and the system proceeds to generate a more detailed diagnostic report. An additional feature in the diagram is speech-to-text conversion, which allows the system to handle voice inputs if necessary. After the diagnosis is generated, registered users have the advantage of accessing historical reports and detailed feedback, whereas anonymous users receive only a limited report.

5.4 Sequence Diagram

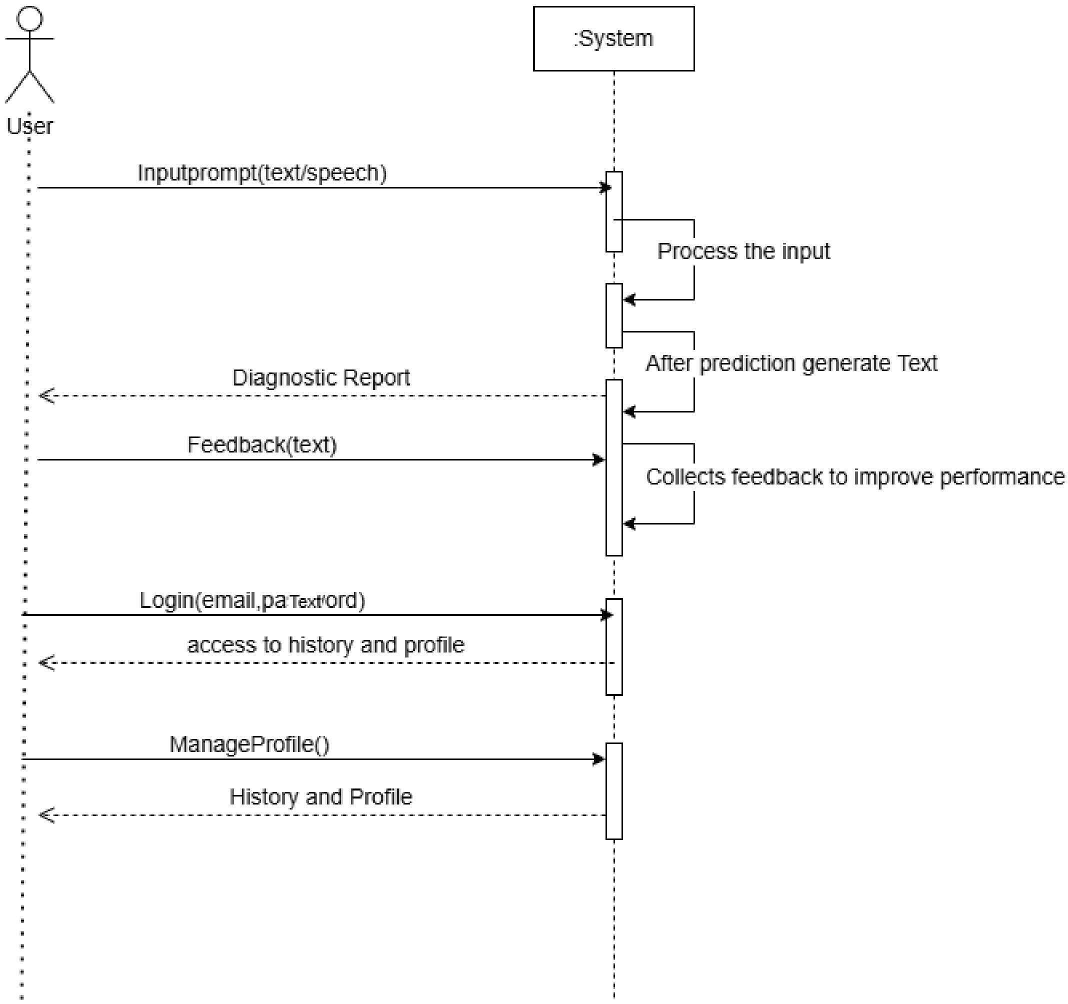


Figure 4: Sequence Diagram

It represents the sequence of interactions between the system's components during the diagnosis process. It begins with the actor (user), who either logs in or remains anonymous and interacts with the system by submitting symptoms. These symptoms are processed through various subsystems, including User Profile Management, Speech Recognition, Data Preprocessing, and LLM (Large Language Model) Prediction. The submitted symptoms are preprocessed and converted from speech to text if needed, after which the data is sent to the LLM for prediction. Based on these predictions, the system generates a diagnostic report, which is displayed to the user. Finally, the system collects feedback from the user regarding the accuracy of the report. This feedback is used to improve the performance of the model, making the system more accurate over time.

5.5 Component Diagram

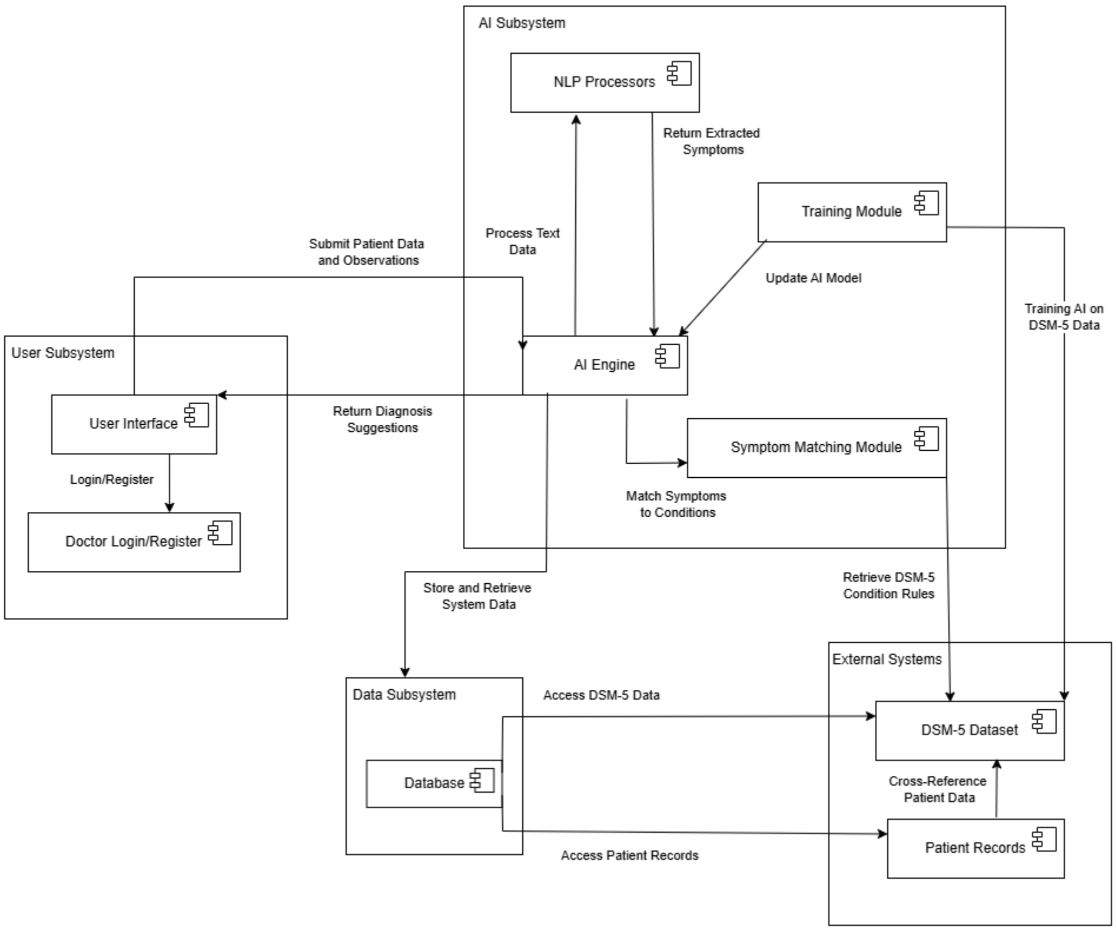


Figure 5: Component Diagram

The diagram shows our system that uses AI to assist in medical diagnosis. It includes four subsystems: User, AI, Data, and External Systems. Doctors submit patient data via the User Subsystem, which the AI processes using NLP to extract symptoms and match them with DSM-5 criteria. The AI is continuously trained to improve accuracy. The Data Subsystem manages stored information, while External Systems provide access to DSM-5 datasets and patient records for cross-referencing.

5.6 Entity Relationship Diagram

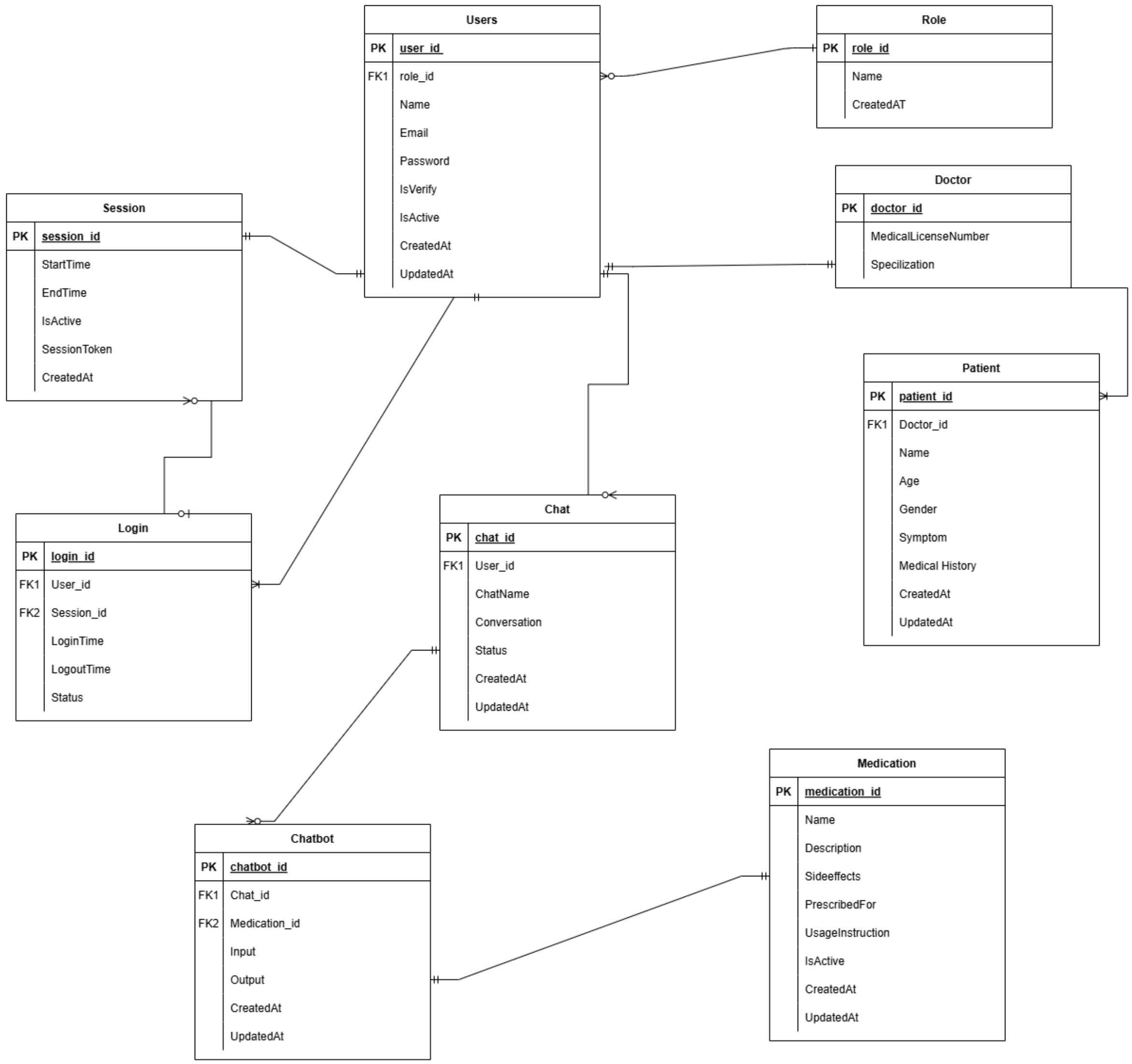


Figure 6: Entity Relationship Diagram

The ERD depicts entities such as ‘User‘, ‘Doctor‘, ‘Patient‘, ‘Session‘, ‘Chat‘, ‘Chatbot‘, ‘Medication‘, and their relationships. Key relationships include:

- ‘User‘ can assume different ‘Roles‘, such as ‘Doctor‘, which inherits properties from the ‘User‘.
- ‘Doctor‘ treats ‘Patients‘ and initiates ‘Chat‘ sessions for consultations.
- ‘Chatbot‘ handles chats and can recommend ‘Medications‘ based on symptoms.

5.7 Context Diagram

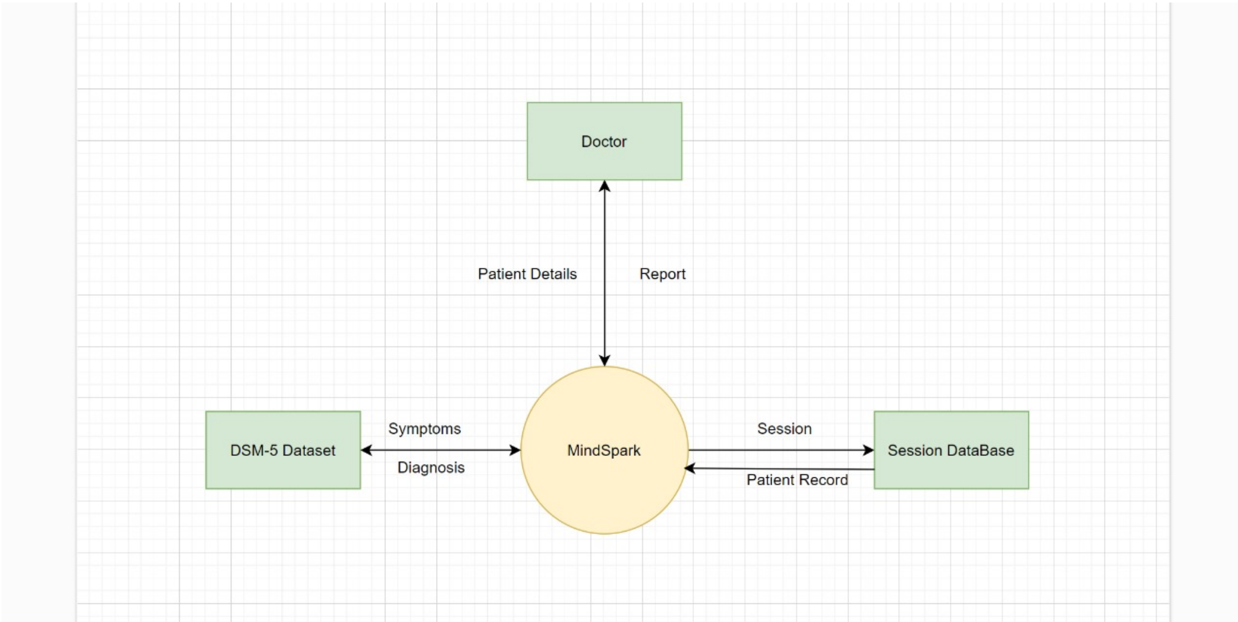


Figure 7: Context Diagram

The context diagram represents the flow of interactions between key components within the "MindSpark" system.

1. Doctor:
   - Inputs patient details into the MindSpark system.
   - Receives a diagnostic report from MindSpark based on the session data and symptom analysis.
2. MindSpark (System itself) :
   - Acts as the central engine, receiving inputs from multiple sources.
   - Receives symptoms from the patient and compares them with the **DSM-5 dataset** to generate a diagnosis.
   - Sends the diagnosis back to the DSM-5 dataset for reference and processing.
   - Handles the **session** details of the patient, recording and storing them in the session Database.
   - Generates and sends patient records and diagnostic reports for doctors to review.
3. DSM-5 Dataset:
   - Provides reference guidelines based on DSM-5 for symptom analysis and diagnosis generation.
4. Session Database:
   - Stores patient session data and records for future reference and diagnostic improvements.

5.8 Data Flow Diagram

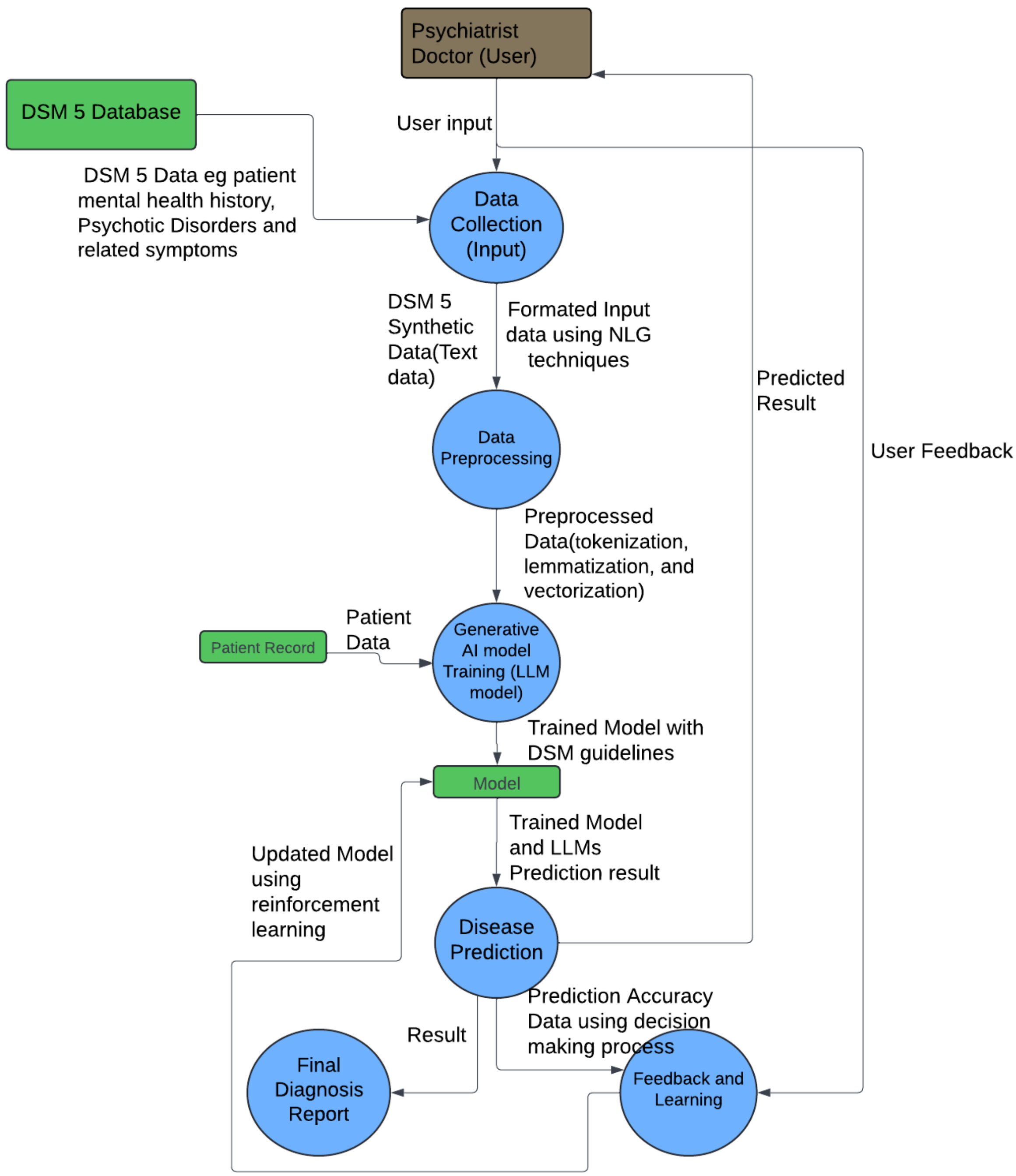


Figure 8: Data Flow Diagram

The DFD shows the flow of data through the system. Data is collected from the psychiatrist (user) and patient records, including input from the DSM-5 database. It undergoes preprocessing, such as tokenization and vectorization, before being used to train an AI model. The trained model predicts potential diagnoses and helps the psychiatrists in sessions, which are refined based on user feedback and reinforcement learning. The system produces a final diagnosis report, continuously updating the model for improved accuracy.

5.9 State Diagram

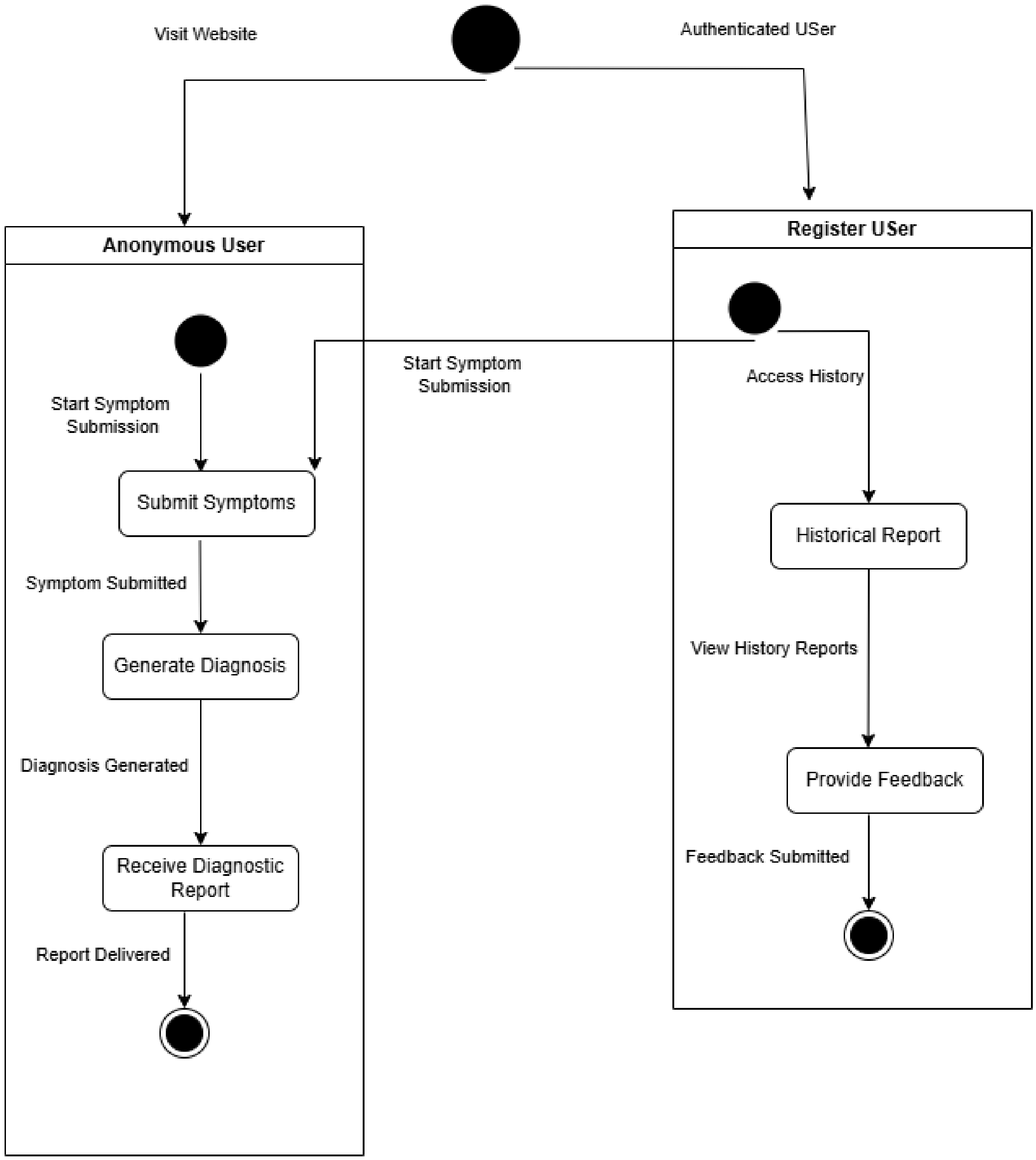


Figure 9: State Diagram

Diagram depicts the interactions between different user roles: Anonymous User, and Registered User. Anonymous users can submit symptoms and receive limited diagnostic reports without history, whereas Registered Users can generate a diagnosis, access detailed reports, diagnostic feedback and view historical reports. This flow represents a system that facilitates symptom submission, diagnostic feedback, and report management, with varying levels of access depending on user registration.

5.10 Deployment Diagram

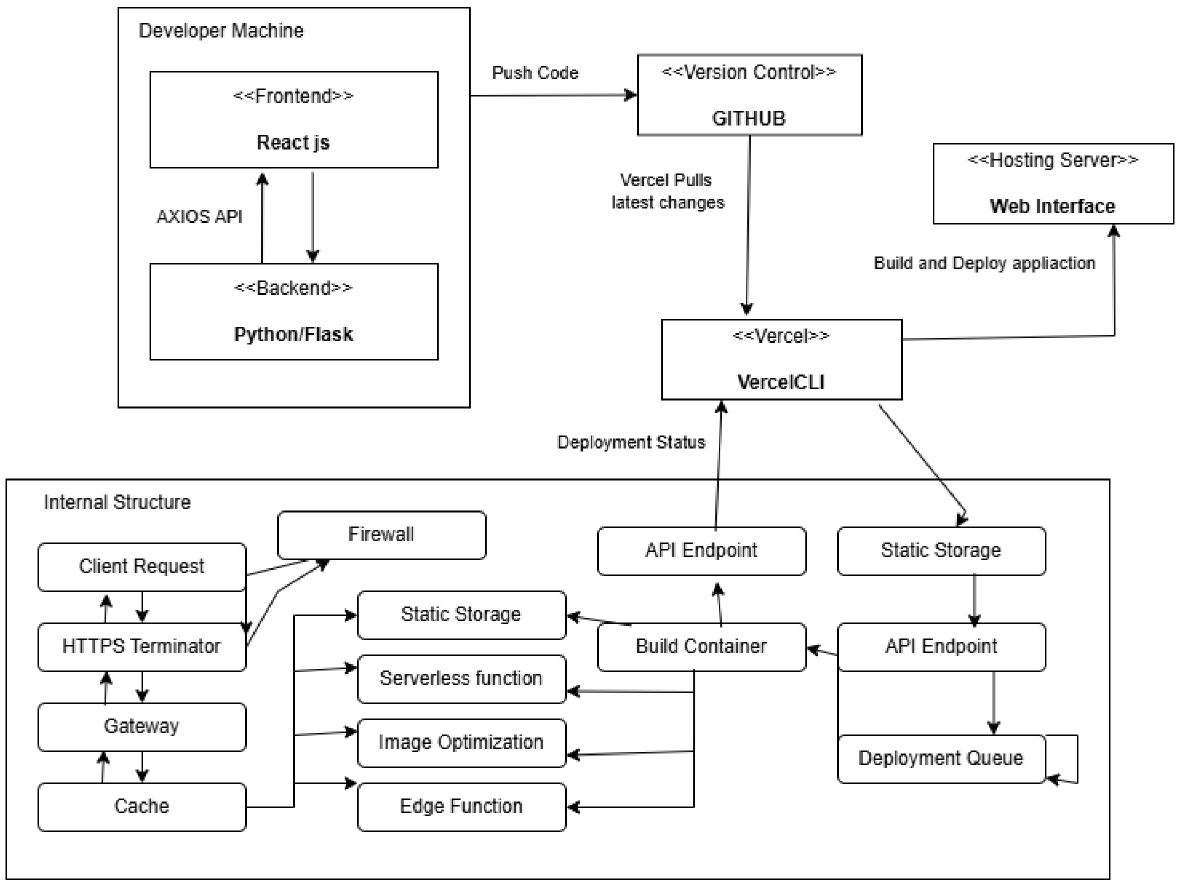


Figure 10: Deployment Diagram

This deployment diagram illustrates the flow of a React.js project hosted on Vercel. It begins with the developer pushing code from their machine to GitHub, where Vercel automatically pulls the latest changes. Vercel's internal infrastructure handles the build process, optimizes assets (e.g., images), and deploys the application using serverless functions, edge functions, and static storage. The final application is hosted on Vercel's web server, ready to serve client requests securely and efficiently.

6 Project Timeline and Resources

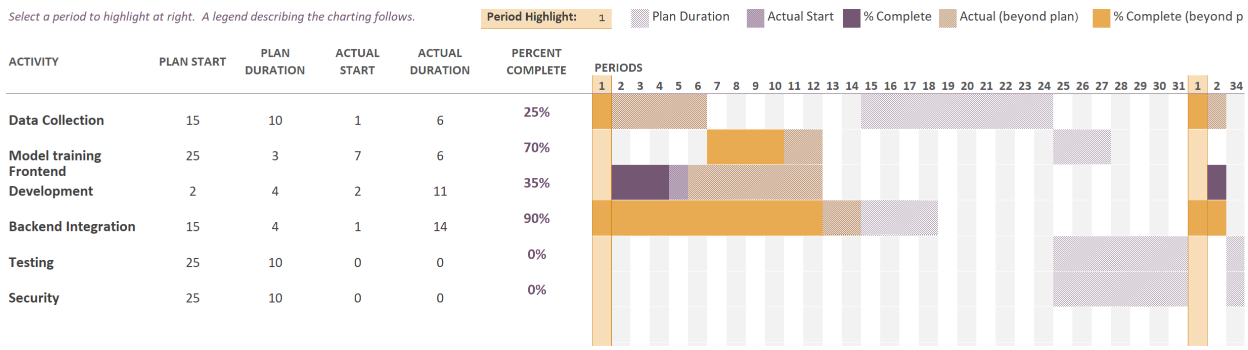
6.1 Team Introduction

The project team consists of two frontend developers and two backend developers:

- **Frontend Developers:** Ammad Aslam (Project Lead) and Uswa Arif. They are responsible for the user interface design, ensuring the system is intuitive and user-friendly, as well as integrating the frontend with the backend.

- **Backend Developers:** Mutaiba Mohsin and Fatima Muskan. They focus on implementing the AI models, handling the backend infrastructure, and ensuring secure and efficient data processing and storage.

6.2 Timeline



ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Data Collection	15	10	1	6	25%
Model training	25	3	7	6	70%
Frontend Development	2	4	2	11	35%
Backend Integration	15	4	1	14	90%
Testing	25	10	0	0	0%
Security	25	10	0	0	0%

6.3 Resource Allocation

6.3.1 Hardware

- Mid-range laptops/PCs for development (e.g., 16GB RAM, Intel Core i7/AMD Ryzen 7)
- Cloud instances with moderate CPU/GPU (e.g., Google Colab, AWS T2/T3 instances)
- External storage (e.g., SSDs or cloud storage like Google Drive) for datasets

6.3.2 Software

- Python (with TensorFlow Lite or ONNX for lightweight model implementation)
- Hugging Face Transformers for fine-tuning smaller models (e.g., DistilGPT or TinyBERT)
- React.js for frontend development
- Node.js for backend API development
- SQLite or Firebase for lightweight database solutions
- DSM-5 parsed data in CSV format for model training
- Jupyter Notebooks for testing and visualization
- Git/GitHub for version control and collaboration

6.4 Task Distribution

- **Backend Developers:**
  - Mutaiba Mohsin (2021-CS-63): Model implementation, dataset handling, and backend testing
  - Fatima Muskan (2021-CS-78): Dataset collection, backend integration, and testing
- **Frontend Developers:**
  - Ammad Aslam (2021-CS-67): UI/UX design for chat interface and frontend implementation
  - Uswa Arif (2021-CS-77): Enhancing the chat page and integrating with the backend

- **All Members:**
  - Data collection (10 days)
  - Security measures and final testing

6.5 Future Steps

1. Expand the dataset to 10k samples for improved accuracy.
2. Fine-tune a lightweight NLP model to generate logical questions and provide insights.
3. Integrate frontend and backend components.
4. Conduct rigorous testing and implement security protocols.

7 System Testing and Validtion

7.1 Use Cases

7.1.1 User Registration and Authentication

Table 1: User Registration and Authentication

User Registration and Authentication
ID: UC-01
Description: Allows account creation, login, and password recovery.
Primary Actor: Psychiatrist
Secondary Actor: System Admin
Preconditions: A valid email address must be available.
Main Flow: 1. The login page is displayed. 2. Credentials are entered and submitted. 3. Credentials are verified. 4. If valid, access is granted.
Postconditions: Authentication is completed successfully.
Alternative Flow: 1. If credentials are incorrect, an error message is displayed. 2. If password is forgotten, a reset request can be initiated. 3. If an invalid email is provided, a prompt is displayed to enter a correct one.

7.1.2 Patient Input Submission

Table 2: Patient Input Submission

Patient Input Submission
ID: UC-02
Description: Allows entry of symptoms through text or speech input.
Primary Actor: Psychiatrist
Secondary Actor: AI System
Preconditions: The symptom submission page must be accessible.
Main Flow: 1. The input section is accessed. 2. Symptoms are provided via text or speech. 3. The system processes and validates the input. 4. Symptoms are saved for further analysis.
Postconditions: Symptoms are successfully recorded.
Alternative Flow: 1. If speech input fails, text input is requested. 2. If no symptoms are provided, a prompt is displayed to enter details.

7.1.3 Diagnosis Generation

Table 3: Diagnosis Generation

Diagnosis Generation
ID: UC-03
Description: Generates potential diagnoses based on submitted symptoms and DSM-5 criteria.
Primary Actor: AI System
Secondary Actor: Psychiatrist
Preconditions: Symptoms must be submitted and stored in the system.
Main Flow: 1. Submitted symptoms are retrieved. 2. AI analysis is performed based on DSM-5 data. 3. Possible diagnoses are identified and listed.
Postconditions: Diagnoses are generated and displayed.
Alternative Flow: 1. If no matching disorder is found, a recommendation for further evaluation is provided.

7.1.4 Report Viewing & Management

Table 4: Report Viewing & Management

Report Viewing & Management
ID: UC-04
Description: Provides access to diagnostic reports with options for review, editing, and downloading.
Primary Actor: Psychiatrist
Secondary Actor: System Database
Preconditions: At least one diagnostic report must exist.
Main Flow: 1. The report management section is accessed. 2. A report is selected for review. 3. The report is displayed with editing and downloading options.
Postconditions: The report is reviewed or downloaded.
Alternative Flow: 1. If no reports exist, a message is displayed stating that no records are available.

7.1.5 Session History Access

Table 5: Session History Access

Session History Access
ID: UC-05
Description: Allows retrieval of past session records and diagnoses.
Primary Actor: Psychiatrist
Secondary Actor: System Database
Preconditions: Access to session history must be available.
Main Flow: 1. The session history page is accessed. 2. Patient records are searched. 3. The system retrieves and displays previous session data.
Postconditions: Past session data is successfully displayed.
Alternative Flow: 1. If no session history is found, a message is displayed.

7.2 Test Cases

7.2.1 User Registration and Authentication

- **Test Case UC-01.1:** Verify that a new user can successfully create an account with valid credentials.

- **Test Case UC-01.2:** Validate that the system rejects account creation with invalid or incomplete information.
- **Test Case UC-01.3:** Test password recovery functionality by verifying that a password reset link is sent and successfully resets the password.

7.2.2 Patient Input Submission

- **Test Case UC-02.1:** Ensure that symptoms can be entered via text input and are correctly saved in the system.
- **Test Case UC-02.2:** Test speech-to-text functionality to verify accurate conversion of spoken symptoms into text.
- **Test Case UC-02.3:** Validate error handling when no symptoms are provided during input.

7.2.3 Diagnosis Generation

- **Test Case UC-03.1:** Verify that the system accurately generates diagnoses based on submitted symptoms and DSM-5 criteria.
- **Test Case UC-03.2:** Test the system's response to rare or complex symptoms to ensure accurate diagnosis suggestions.
- **Test Case UC-03.3:** Validate the system's recommendation for further evaluation when no matching disorder is found.

7.2.4 Report Viewing & Management

- **Test Case UC-04.1:** Ensure that psychiatrists can access and view diagnostic reports for accuracy and completeness.
- **Test Case UC-04.2:** Test editing functionality to validate that changes to reports are saved correctly.
- **Test Case UC-04.3:** Verify downloading options to ensure reports are available in the desired format (e.g., PDF).

7.2.5 Session History Access

- **Test Case UC-05.1:** Validate the system's ability to retrieve and display past session records and diagnoses.
- **Test Case UC-05.2:** Test search functionality to ensure that specific patient records can be easily located.
- **Test Case UC-05.3:** Verify error handling when no session history is found for a patient.

8 System Implementation and Deployment

This section describes the implementation of the AI-powered psychiatric assistant model.

8.1 Data Extraction and Preprocessing

The DSM-5 manual was extracted into a CSV file, containing disorder criteria and symptoms. A sample of the extracted data is shown in Table 6.

Disorder	Criteria	Treatment Recommendations
Hyperactivity Disorder	Anxiety, attention, avoidance, behavior, behavioral, behaviors	Behavioral therapy, stimulant medications (e.g., Methylphenidate), structured routine
Delusional Disorder	Behavior, delusional, delusions, depression, disorganized	Antipsychotics (e.g., Risperidone), Cognitive Behavioral Therapy (CBT)

Table 6: Sample extracted DSM-5 data

8.2 Conversation Data

A CSV file containing symptom-specific diagnostic questions was used to train the model for AI-generated question retrieval. A sample of the extracted data is shown in Table 7.

Symptom	Diagnostic Questions
Depression	Do you often feel sad or empty?, Have you lost interest in activities you used to enjoy?, Do you feel fatigued even after resting?, Have you experienced changes in your sleep patterns?, Do you feel hopeless about the future?
Anxiety	Do you frequently worry about things even when there's no clear reason?, Do you experience restlessness or difficulty relaxing?, Do you often feel a sense of impending danger or panic?, Have you noticed increased heart rate or sweating during stressful situations?, Do you struggle with concentrating due to constant worrying?

Table 7: Sample extracted symptom-question data for AI-generated question retrieval

8.3 Model Modules

The AI-Powered Psychiatric Assistant consists of the following key components:

- **Question Retrieval:** This module selects the most relevant diagnostic questions based on the user's symptoms. It ensures that the questions align with psychiatric guidelines for accurate assessment. The retrieval process adapts dynamically to different patient inputs. By selecting precise questions, it improves diagnostic efficiency. This helps in structured and systematic patient evaluation.
- **Empathetic Question Rephrasing:** To enhance communication, this module reformulates clinical questions in a more natural way. It ensures that diagnostic queries sound less rigid and more conversational. The rephrased questions maintain medical accuracy while making patients feel at ease. This reduces hesitation in sharing personal experiences. A more engaging approach leads to better diagnostic insights.
- **Disorder Detection:** This module analyzes patient symptoms and compares them against known diagnostic criteria. It identifies patterns and suggests potential disorders based on similarity with existing conditions. By ranking possible disorders, it provides structured insights for further evaluation. This assists healthcare professionals in narrowing down diagnoses. The process ensures a data-driven approach to mental health assessment.

- **Chat Summarization:** To assist doctors in reviewing cases, this module condenses patient conversations into key points. It extracts relevant details such as reported symptoms, emotional expressions, and follow-up questions. The summaries help in tracking patient progress over time. By reducing information overload, it allows for faster and more effective decision-making. This improves efficiency in clinical workflows.
- **Treatment Recommendations:** Once a disorder is detected, this module provides suitable treatment options. It suggests therapeutic approaches, possible medications, and lifestyle changes based on clinical guidelines. The recommendations are tailored to the patient's condition for personalized care. By presenting clear options, it helps in informed decision-making. This ensures a holistic and structured approach to treatment planning.

8.4 Backend Technologies

- **Question Retrieval:**
  - **FAISS (Facebook AI Similarity Search):** Efficiently retrieves the most relevant diagnostic question from the dataset.
  - **MiniLM:** Embedding model for optimizing question search.
- **Empathetic Question Rephrasing:**
  - **DeepSeek API:** Enhances retrieved diagnostic questions by rewording them in a more conversational and empathetic manner.
  - **Groq API:** Assists in refining the phrasing of questions to improve patient engagement.
- **Disorder Detection:**
  - **MPNet:** Computes similarity between patient symptoms and disorder descriptions in `DisorderTreatment.csv`.
- **Chat Summarization:**
  - **T5 Transformer Model:** Summarizes long conversations for quick review.
- **Treatment Recommendations:**
  - **Groq API:** Extracts and generates relevant treatment suggestions based on diagnosed disorders.

8.5 Approaches Used

Several approaches were explored for optimizing performance. Each approach was evaluated based on efficiency, accuracy, and conversational adaptability.

8.5.1 Simple RAG Approach

This approach employed a basic Retrieval-Augmented Generation (RAG) method to fetch diagnostic questions and generate responses. It utilized a high-speed search mechanism to retrieve stored medical data efficiently. FAISS was implemented to index and search symptom-related questions, allowing for quick retrieval based on similarity. MiniLM was used to generate embeddings for patient queries, ensuring relevant matches with pre-existing questions. Disorder detection was handled using MPNet, which analyzed symptom descriptions and predicted potential disorders based on similarity to stored medical cases. The T5 model was used for summarization, providing concise overviews of patient interactions. However, while effective in structured retrieval, this approach struggled with dynamically rephrasing questions and generating more natural, contextual responses.

8.5.2 DeepSeek-Based Approach

The DeepSeek-based approach leveraged the DeepSeek API for advanced question retrieval and natural language understanding. FAISS indexed the database of symptom-related questions, and when a symptom query was made, DeepSeek selected the most relevant question. Instead of presenting the retrieved question directly, DeepSeek dynamically rephrased it to make it sound more conversational and empathetic. This helped improve patient engagement by avoiding rigid, robotic phrasing. Additionally, DeepSeek handled response generation, providing context-aware and human-like replies. FAISS + MiniLM were used as a supplementary retrieval system to refine search efficiency. While this approach improved adaptability in conversations, it had some limitations, such as dependency on the API and less control over internal fine-tuning.

8.5.3 Hybrid RAG Approach (Final Selection)

The final approach combined structured retrieval with dynamic response generation, balancing efficiency and conversational flexibility. FAISS retrieved relevant diagnostic questions based on symptom input, while Groq API rephrased these retrieved questions to enhance naturalness and patient engagement. For disorder detection, MPNet was utilized to compare patient symptoms with known diagnostic criteria, improving predictive accuracy. The T5 model played a key role in summarization, ensuring that patient interactions and medical discussions were documented concisely.This hybrid approach effectively integrated multiple specialized components, ensuring smooth, context-aware conversations while maintaining medical accuracy. Due to its ability to retrieve, rephrase, detect disorders, and generate structured reports, this approach was selected as the final implementation.

8.6 Model Comparison

Criteria	**Simple RAG Approach**	**DeepSeek-Based Approach**	**Hybrid RAG Approach (Final Selection)**
Question Retrieval	FAISS + MiniLM for retrieving predefined questions	FAISS + DeepSeek API for retrieval and contextual selection	FAISS + MiniLM for retrieval, enhanced by Groq API for natural phrasing
Question Rephrasing	No dynamic rephrasing; presents retrieved questions as-is	DeepSeek API dynamically rephrases retrieved questions	Groq API rephrases retrieved questions to improve engagement
Disorder Detection	MPNet for matching symptoms to stored disorder descriptions	DeepSeek API handles symptom interpretation and response generation	MPNet detects disorders based on symptom similarity analysis
Response Generation	Limited to predefined responses; lacks conversational flexibility	DeepSeek API generates natural, human-like responses	Hybrid model integrates retrieval and generative techniques for balanced responses
Chat Summarization	T5 summarizes patient interactions	Not explicitly handled; responses are direct but not summarized	T5 model generates structured summaries of patient interactions
Treatment Recommendations	Basic retrieval of predefined treatment suggestions	DeepSeek API provides AI-generated treatment responses	Groq API generates structured treatment recommendations based on disorder detection
Strengths	Fast and efficient for structured retrieval	More natural conversation flow and human-like responses	Combines structured retrieval with dynamic AI-generated responses for balanced performance
Limitations	Lacks natural conversation flow; responses feel robotic	Dependent on DeepSeek API, limiting fine-tuning control	More complex implementation but achieves optimal balance of efficiency and adaptability

Table 8: Comparison of Different Approaches

By combining retrieval-based and generative methods, the Hybrid RAG approach offered an optimal balance of speed, accuracy, and natural conversation flow.

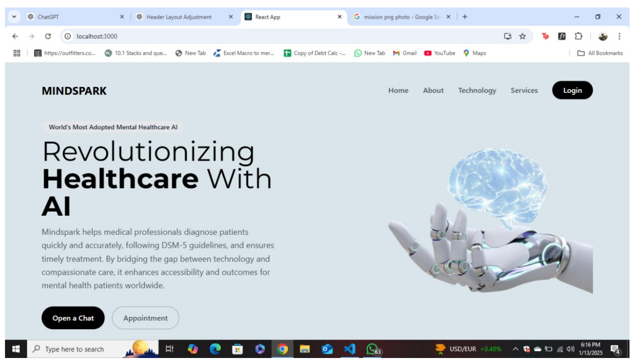
8.7 Frontend Technologies

The frontend of the "Mind Spark" project is built using modern web development technologies aimed at delivering an intuitive and responsive user experience. The core technologies used are as follows:
React: For developing the cross-platform mobile-web application. With React's component-based architecture, different sections of the application—such as the chatbot, sign-up, and sign-in pages—are easily managed and updated independently, ensuring smooth interactions for users.
Axios: For making HTTP requests between the frontend and backend. Axios also facilitates secure user authentication on the Sign Up and Sign In pages by sending user credentials and managing token-based authentication.
Tailwind CSS: For styling the user interface with a utility-first approach.
Material UI (MUI): To provide pre-built, customizable React components that align with modern design principles.

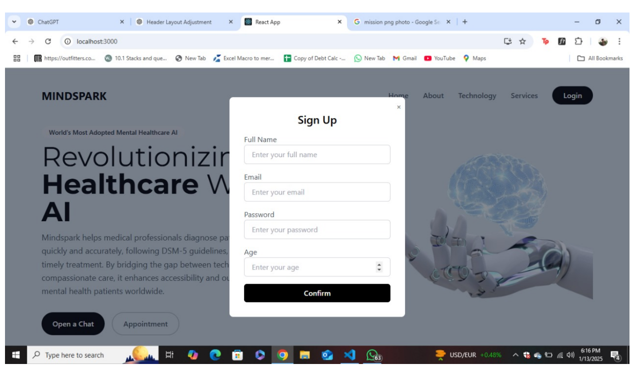
Pages

The following are the key pages developed using these frontend technologies:

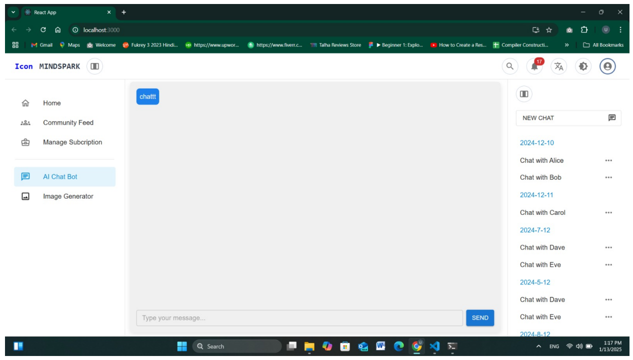
- **Landing Page:** The Landing Page introduces users to the "Mind Spark" platform, featuring clear navigation options and a visually engaging design. Users are provided with an overview of the platform's services, testimonials, and calls-to-action to either sign up or sign in.



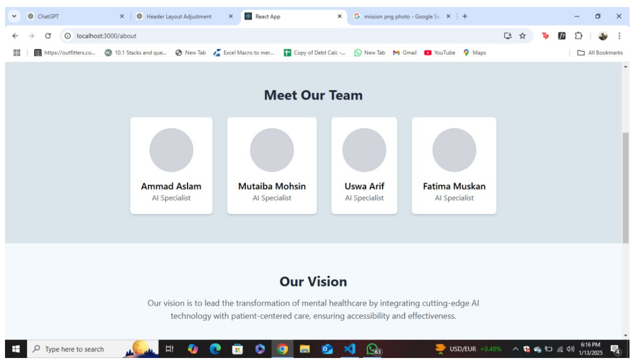
- **Sign Up and Sign In Pages:** These pages allow users to create accounts or log in to access the chatbot and mental health support services. Axios handles the API requests to the backend for user registration and authentication, ensuring secure transmission of data. These pages include validation and error handling to improve the user experience.



- **Chatbot Page:** The Chatbot Page is the central feature of the "Mind Spark" platform. It allows users to interact with the AI chatbot, which provides mental health support by answering queries and offering advice. The user input is sent to the backend via Axios, where the AI model processes the data and returns responses that are displayed in the chat window.



- **About Us Page:** The About Us Page provides detailed information about the team behind "Mind Spark" and the platform's mission. This page also highlights the importance of mental health and how "Mind Spark" aims to provide support through its AI-powered platform.



- **Services Page:** The Services Page outlines the various services offered by "Mind Spark," including mental health assessments, chatbot interactions, and resources for users.

8.8 Deployment

The AI-powered psychiatric assistant was deployed using a cloud-based architecture to ensure scalability, efficiency, and accessibility.

- **Model Deployment:**
  - **Hugging Face Model Hub:** The core models, including MiniLM, MPNet, and T5, were hosted on Hugging Face for easy access and efficient inference.
  - **Hugging Face Inference API:** Enabled seamless interaction with hosted models, reducing deployment complexity.
  - **FastAPI:** Used for serving API requests, integrating retrieval, rephrasing, disorder detection, and summarization into a unified system.
- **Frontend Deployment:**
  - **Vercel:** Deployed the frontend, ensuring high availability, fast performance, and easy updates.
  - **Git:** Managed source code versioning and continuous integration for frontend updates.

8.9 Challenges Encountered and Solutions Applied

During the development of the PsychTrait application, multiple challenges arose. Below are the key obstacles and the solutions applied across different approaches.

8.9.1 Data Scarcity

Challenge: Limited availability of public datasets due to patient privacy.
Solution:

- **Simple RAG**: Relied on FAISS-based retrieval over existing structured datasets.
- **DeepSeek-based**: Leveraged DeepSeek API's broader knowledge base for question generation to mitigate dataset limitations.
- **Hybrid RAG**: Combined structured retrieval (FAISS) with DeepSeek-assisted contextual refinement, improving data coverage.

8.9.2 Complexity of Medical Text

Challenge: Psychiatric texts contain complex language, medical jargon, and subtle nuances, making them difficult for AI models to interpret.
Solution:

- **Simple RAG**: T5-based summarization helped condense and simplify text.
- **DeepSeek-based**: Used DeepSeek API's advanced NLP capabilities for natural language understanding.
- **Hybrid RAG**: Combined structured summarization (T5) with context-aware language processing.

8.9.3 Model Performance and Fine-tuning

Challenge: Ensuring models generalize well without overfitting or underfitting.
Solution:

- **Simple RAG**: Employed FAISS and MPNet for retrieval without heavy model fine-tuning.
- **DeepSeek-based**: Relied on DeepSeek API's pre-trained capabilities, reducing fine-tuning requirements.
- **Hybrid RAG**: Applied fine-tuning techniques (cross-validation, early stopping) on MiniLM for improved retrieval efficiency.

8.9.4 Symptom Matching and Disorder Identification

Challenge: Accurately matching symptoms to DSM-5 disorders is difficult due to overlapping or ambiguous symptom descriptions.
Solution:

- **Simple RAG**: Used MPNet-based similarity matching.
- **DeepSeek-based**: DeepSeek API directly handled disorder detection via generative reasoning.
- **Hybrid RAG**: Combined MPNet similarity matching with assisted disorder identification.

8.9.5 Privacy and Security Concerns

Challenge: Handling sensitive patient data while ensuring compliance with regulations like HIPAA.
Solution:

- **All Approaches**: Enforced encryption, anonymization, and secure API endpoints to protect patient data.

8.9.6 Interpretability and Transparency

Challenge: AI models, especially LLMs, are often black-box systems, making it difficult to explain predictions.
Solution:

- **Simple RAG**: Provided rule-based explanations from structured disorder databases.
- **DeepSeek-based**: Used DeepSeek API's inherent interpretability features.
- **Hybrid RAG**: Combined structured disorder explanations with reasoning.

8.9.7 Real-time Performance and User Interaction

Challenge: Ensuring low-latency responses to maintain smooth user experience.
Solution:

- **Simple RAG**: Optimized FAISS retrieval for fast question lookup.
- **DeepSeek-based**: Relied on DeepSeek API's efficient inference.
- **Hybrid RAG**: Cached frequently used responses and optimized API calls for speed.

8.9.8 Bias in AI Models

Challenge: AI models can inherit biases from training data, leading to unfair outcomes.
Solution:

- **Simple RAG**: Curated a balanced disorder-symptom dataset.
- **DeepSeek-based**: Applied prompt engineering techniques to mitigate bias.
- **Hybrid RAG**: Combined structured retrieval with Groq API fairness-aware learning.

8.9.9 Model Accuracy

Challenge: Ensuring high prediction accuracy for psychiatric diagnoses.
Solution:

- **Simple RAG**: Improved FAISS indexing for more relevant question retrieval.
- **DeepSeek-based**: Leveraged DeepSeek API's powerful generative capabilities.
- **Hybrid RAG**: Combined FAISS, MPNet, and Groq API for optimal accuracy.

9 Conclusion

This project is an important step in using AI to help psychiatrists during patient sessions. By using AI models based on DSM-5 guidelines, the system supports doctors by recognizing symptoms, suggesting possible diagnoses, and creating detailed reports. This reduces the need for manual note-taking and helps ensure important information is not missed. The system also uses speech-to-text and natural language processing to easily handle both spoken and written inputs from patients. A major benefit of the project is its cloud-based design, which makes it scalable, secure, and accessible from anywhere. This will also allow future integration with Electronic Health Record (EHR) systems, offering a complete solution for mental health professionals. The platform is built to be user-friendly, so doctors with different levels of tech experience can use it effectively. When finished, this system will provide psychiatrists with a reliable tool that improves diagnosis, speeds up their workflow, and leads to better patient care. Future improvements will focus on enhancing the AI, making the system even easier to use, and ensuring it meets all healthcare data security regulations. This project has the potential to make a real difference in mental health care and pave the way for more AI-driven solutions in the field.

10 Appendices

10.1 Appendix A: Glossary

- DSM-5 (Diagnostic and Statistical Manual of Mental Disorders, Fifth Edition):* A manual used by healthcare professionals to diagnose mental disorders.
- EHR (Electronic Health Record):* Digital version of a patient's paper chart that contains the medical and treatment history of patients.
- NLP (Natural Language Processing):* A field of AI that focuses on the interaction between computers and humans through natural language.
- Speech-to-Text:* Technology that converts spoken language into written text.

10.2 Appendix B: References

1. American Psychiatric Association. (2013). Diagnostic and Statistical Manual of Mental Disorders (5th ed.).
2. Duan, H., and Deng, Z. (2020). "Applications of Artificial Intelligence in Mental Health Diagnosis." Journal of Health Informatics Research, 6(2), 112-123.
3. Topol, E. J. (2019). "Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again." Basic Books.

10.3 Appendix C: Related Work

- Babylon Health: A digital health service that offers AI-powered consultations based on personal medical history and common medical knowledge.
- Ada Health: An AI-driven health companion app that helps users understand their symptoms and suggests possible conditions.