CHATBOT SYSTEM

BY

Keval Patel - 21162171022
Shubham Patel - 21162171024
Srijit Das – 21162171010

SRS FORMAT

01

Introduction

02

Overall Description

03

Requirement Specifications

04

Diagrams

INTRODUCTION

- Purpose
- Scope
- Glossary
- Overview

PURPOSE

- A chatbot is a computer program that is designed to simulate conversation with human users, especially over the Internet.
- Chatbots are often used in customer service to answer frequently asked questions, or to help customers complete a task.
- Chatbots can also be used for entertainment, such as creating chatbots that can tell jokes or play games with users.
- Some chatbots are designed to be very human-like in their conversation, while others are more straightforward and simply provide information.
- Chatbots are usually accessed through a chat interface, such as a messaging app, a website, or a mobile app..

SCOPE

- Some chatbots are designed to handle a specific task, such as answering questions about a particular topic or helping users make a purchase.
- Other chatbots are more general-purpose and can handle a wide range of tasks and conversations.
- The scope of a chatbot is ultimately determined by the goals of its creators and the capabilities of the underlying technology.

GLOSSARY

1. Artificial intelligence (AI) - The branch of computer science that aims to create intelligent machines that can think and act like humans.

2. Natural language processing (NLP) - The ability of a computer to understand, interpret, and generate human language.

3. Chatbot - A computer program designed to simulate conversation with human users, especially over the Internet.

4. Chat interface - A user interface that allows users to communicate with a chatbot, typically through a messaging app, website, or mobile app.

5. Dialog flow - The sequence of interactions that take place between a user and a chatbot during a conversation.

GLOSSARY

6. Entity - A specific piece of information that a chatbot can recognize and extract from a user's message.

7. Knowledge base - A database of information that a chatbot can use to answer questions or provide information to users.

8. Machine learning - A type of artificial intelligence that enables computers to learn and adapt to new data without being explicitly programmed.

OVERVIEW

- A chatbot is a computer program designed to simulate conversation with human users, especially over the Internet.
- Chatbots can be used in a variety of contexts, such as customer service, marketing, or entertainment.
- They can be found on websites, messaging apps, and even as part of virtual assistants on smartphones.

OVERVIEW

- There are two main types of chatbots: rule-based and artificial intelligence-based.

  1. Rule-based chatbots are programmed to follow a specific set of rules to respond to user input. These chatbots can handle simple queries and provide basic information, but they are not able to understand or respond to more complex questions.

  2.Overall, chatbots can be a useful tool for businesses or organizations looking to improve customer service, generate leads, or engage with users in a more interactive way.

- Artificial intelligence-based chatbots, on the other hand, use machine learning algorithms to understand and respond to user input. These chatbots can understand and interpret more complex queries and can generate responses that are more natural and human-like.

OVERALL DESCRIPTION

- Problem Statement
- Existing System
- Proposed System
- Product Functions
- User’s Characteristics
- Constraints
- Assumption And Dependencies

PROBLEM STATMENT

- Many companies today are still struggling with engaging their audience, providing 24/7 customer service, and streamlining their sales processes.
- As a result, businesses are losing potential customers and revenue.
- A chatbot could be an effective solution for these issues by providing around-the-clock customer service, lead generation and qualification, and automating sales processes.
- The goal of this project is to develop a chatbot that can understand natural language input, answer questions and perform tasks.
- And also providing accurate and personalized responses to customer inquiries, increasing customer satisfaction and sales revenue.

EXISTING SYSTEM

- The existing system consists of either not up-to-the-mark chatbots or workers with paid jobs for this position.
- This leads to higher resource cost over just subscribing to a chatbot system for over the surface interaction with customers.

PROPOSED SYSTEM

- We propose the use our chatbot system for every basic interactions that can be integrated with any business.
- Any and all inquiries regarding prices, refundability, availability(of say, a 'Product' for example) or any other cursory topics can be easily integrated into and handled by the system.
- For all the 'Business-Specific' questions which may only be known to a certain position of employees and above, the task can be transferred over to an employee.

PRODUCT FUNCTIONS

- Natural Language Processing (NLP) to understand and respond to user input in a human-like manner.
- Integration with external data sources and APIs to provide relevant information to users.
- Ability to handle multiple languages and dialects.
- User engagement and conversation flow management to guide users through a task or conversation.
- Ability to handle a wide range of user inquiries, from simple questions to more complex interactions.
- Customizable responses and the ability to handle different types of user data.
- Advanced analytics and reporting to track performance and improve the chatbot's effectiveness over time.
- Self-learning capability to improve the chatbot's understanding of user input over time.

USER'S CHARACTERISTICS

- User's Requirements
- User's Educational Level
- User's Technical Expertise

USER'S REQUIRMENTS

- The ability to handle a high volume of customer inquiries, providing 24/7 availability and reducing the need for human customer service representatives.
- Integration with existing systems and databases, such as customer information and order history, to provide personalized and accurate responses to customers.
- The ability to handle multiple languages and dialects to serve a global customer base.
- The ability to handle high traffic and multiple users simultaneously with inquiries including frequently asked questions and more complex issues.
- Compliance with data protection and security regulations.
- The ability to guide customers through the purchasing process, upselling or cross-selling products, and increasing the conversion rates.

USER'S EDUCATIONAL LEVEL

- The chatbot system should be designed to be user-friendly and easy to navigate for users of all educational levels.
- The interface should be intuitive and the language used should be simple and easy to understand, avoiding complex technical jargon or terminology.
- The chatbot should be able to provide step-by-step instructions and guidance for completing tasks, such as making a purchase or troubleshooting an issue, to ensure that all users can successfully interact with the system.
- The chatbot should be designed to be flexible and adaptable to the users' needs, and should be able to adjust its responses based on the user's educational level, where possible.

USER'S TECHNICAL EXPERTISE

- We would not assume any particular level of technical expertise for our users.
- Additionally, the chatbot should have a self-help feature, such as a FAQ section, to provide solutions to common problems without the need for human assistance.

CONSTRAINTS

- Technical constraints such as hardware and software limitations, which can affect the chatbot's performance and scalability.
- Time and budget constraints that may limit the scope and complexity of the chatbot's capabilities.
- Data privacy and security constraints that may limit the types of information that the chatbot can collect and store.
- Legal and regulatory constraints that may limit the chatbot's ability to operate in certain jurisdictions or industries.
- Language and dialect constraints that may limit the chatbot's ability to understand and respond to user input in certain languages or dialects.
- Accessibility constraints that may limit the chatbot's ability to serve users with disabilities or accessibility needs.
- Performance constraints such as response time and accuracy that may affect the chatbot's usability and user satisfaction

ASSUMPTION AND DEPENDENCIES

- Dependence on the availability and reliability of internet and network connections for the chatbot to function.
- Dependence on the availability and accuracy of external data sources, such as databases and APIs, which the chatbot relies on to provide relevant information to users.
- Dependence on the availability of the necessary personnel, such as data scientists, to maintain and update the chatbot.
- Assumptions about the device and platform compatibility, such as browsers, mobile and desktop operating systems, and screen sizes, which the chatbot needs to support.
- Assumptions about the user's technical expertise and familiarity with the chatbot's interface and functionality.
- Assumptions about the user's language and dialect, which the chatbot needs to support.

REQUIREMENT SPECIFICATION

- Functional Requirements
- Non Functional Requirements

FUNCTIONAL REQUIREMENTS

- Performance Requirements
- Design Constraints
- Hardware Requirements
- Software Requirements
- Other Requirements

PERFORMANCE REQUIREMENTS

- **Response time:** The chatbot should be able to respond to user inquiries within an acceptable time frame, typically a few seconds to a few minutes, depending on the complexity of the inquiry.
- **Accuracy:** The chatbot should be able to provide accurate and relevant responses to user inquiries, with a high degree of confidence.
- **Scalability:** The chatbot should be able to handle a high volume of user inquiries and support a large number of users simultaneously.
- **Security:** The chatbot should be able to protect user data and ensure compliance with data protection and security regulations.
- **Customizability:** The chatbot should be able to adapt and tailor its responses based on the user's input, context, and history.

DESIGN CONSTRAINTS

- User interface constraints, such as the availability of different modes of interaction (text, voice, touch) and the availability of different input methods (keyboard, touch-screen, speech).
- Accessibility constraints that may limit the chatbot's ability to serve users with disabilities or accessibility needs.
- Architecture constraints may lead to incomplete reach as should have been projected.

HARDWARE REQUIREMENTS

- **CPU:** The chatbot should have sufficient processing power to handle natural language processing (NLP) and other complex tasks.
- **Memory:** The chatbot should have sufficient memory to store and access the knowledge base and other data.
- **Storage:** The chatbot should have sufficient storage capacity to store the knowledge base, historical data, and other information.
- **Connectivity:** The chatbot should have sufficient network connectivity to communicate with external systems and data sources.
- **Power consumption:** The chatbot's hardware should have low power consumption to minimize the cost of operation.
- **Durability:** The chatbot's hardware should be durable to operate in different environments.
- **Portability:** The chatbot's hardware should be portable if the chatbot needs to be deployed in different locations.

SOFTWARE REQUIREMENTS

- **Natural Language Processing (NLP) capability:** The chatbot should be able to understand and respond to user input in a human-like manner.
- **Integration with external data sources and APIs:** The chatbot should be able to integrate with external systems and data sources to provide relevant information to users.
- **Multi-language support:** The chatbot should be able to handle multiple languages and dialects.
- **Conversation flow management and Customizable responses:** The chatbot should be able to guide users through a task or conversation and should be able to handle different types of user data and provide customizable responses.
- **Self-learning capability:** The chatbot should be able to improve its understanding of user input over time.
- **Compatibility:** The chatbot should be compatible with different devices and platforms such as browsers, mobile, and desktop operating systems, and screen sizes.

OTHER REQUIREMENTS

- **Data management:**
  - Users: Company Employees, DBAdmins
  - Result: The chatbot should have the capability to manage and store user data in a secure and compliant manner.

- **Human fallback:**
  - Users: End-User Customers
  - Result: The chatbot should have the capability to escalate a conversation to a human agent if it is unable to handle the user's request.

- **Compliance:**
  - Users: Company Employees
  - Result: The chatbot should comply with relevant industry regulations and standards, such as GDPR, HIPAA, and SOC2.

OTHER REQUIREMENTS

- **Maintenance and updates:**
  - Users: Developers
  - Result: The chatbot should have a mechanism for regular maintenance and updates to improve its performance and add new features.

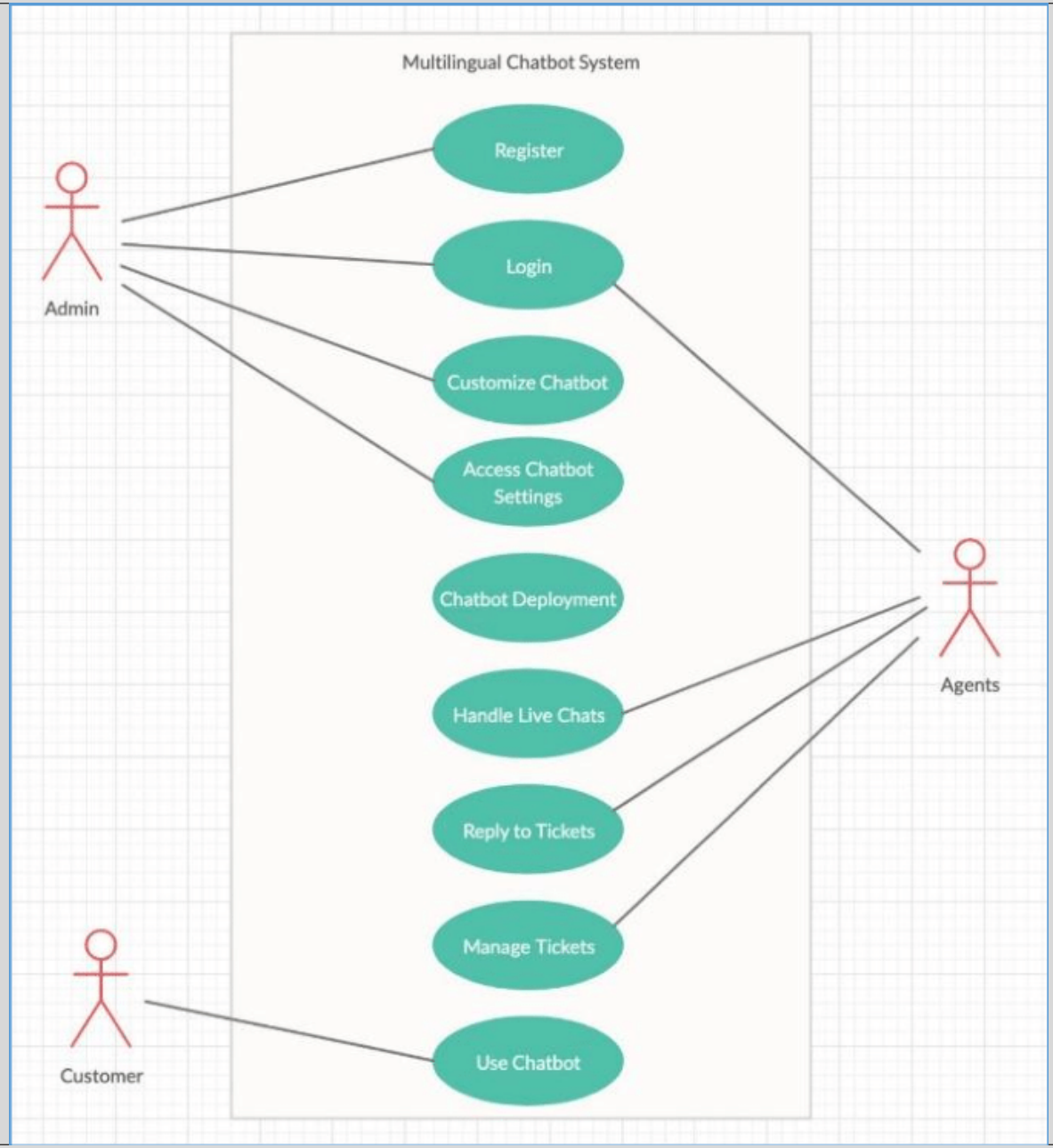
- **Monitoring and reporting:**
  - Users: Company Employees
  - Result: The chatbot should have the capability to monitor and report on its performance, including metrics such as response time and accuracy.

NON FUNCTIONAL REQUIREMENTS

- **Scalability:** The chatbot should be able to handle a high volume of user inquiries and support a large number of users simultaneously.
- **Maintainability:** The chatbot should be designed to be easy to maintain and update, with minimal downtime or disruption to users.
- **Portability:** The chatbot should be able to run on different hardware and software platforms.
- **Extensibility:** The chatbot should be designed to allow for future expansion and growth of its capabilities.
- **Testability:** The chatbot should be designed to be easily testable, with clear and measurable performance metrics.

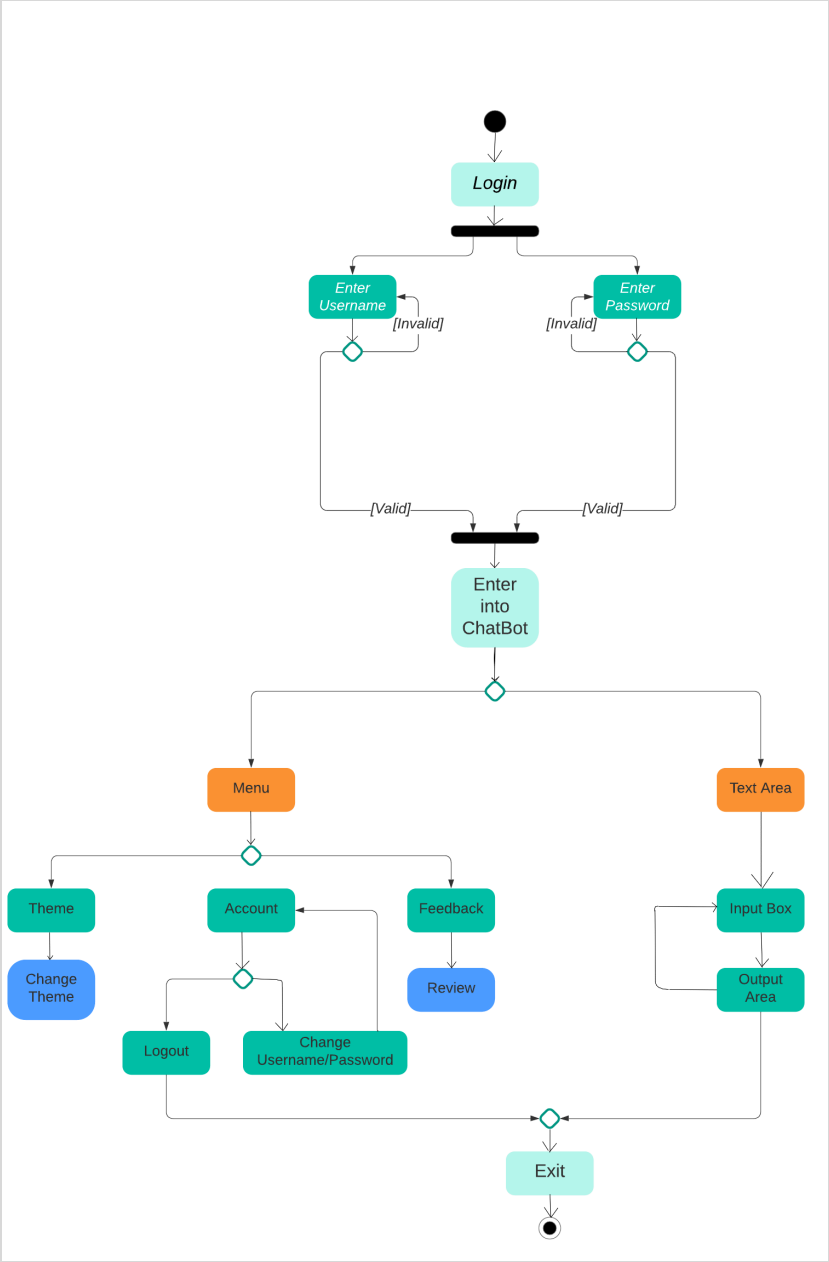
DIAGRAMS

- Use Case Diagram



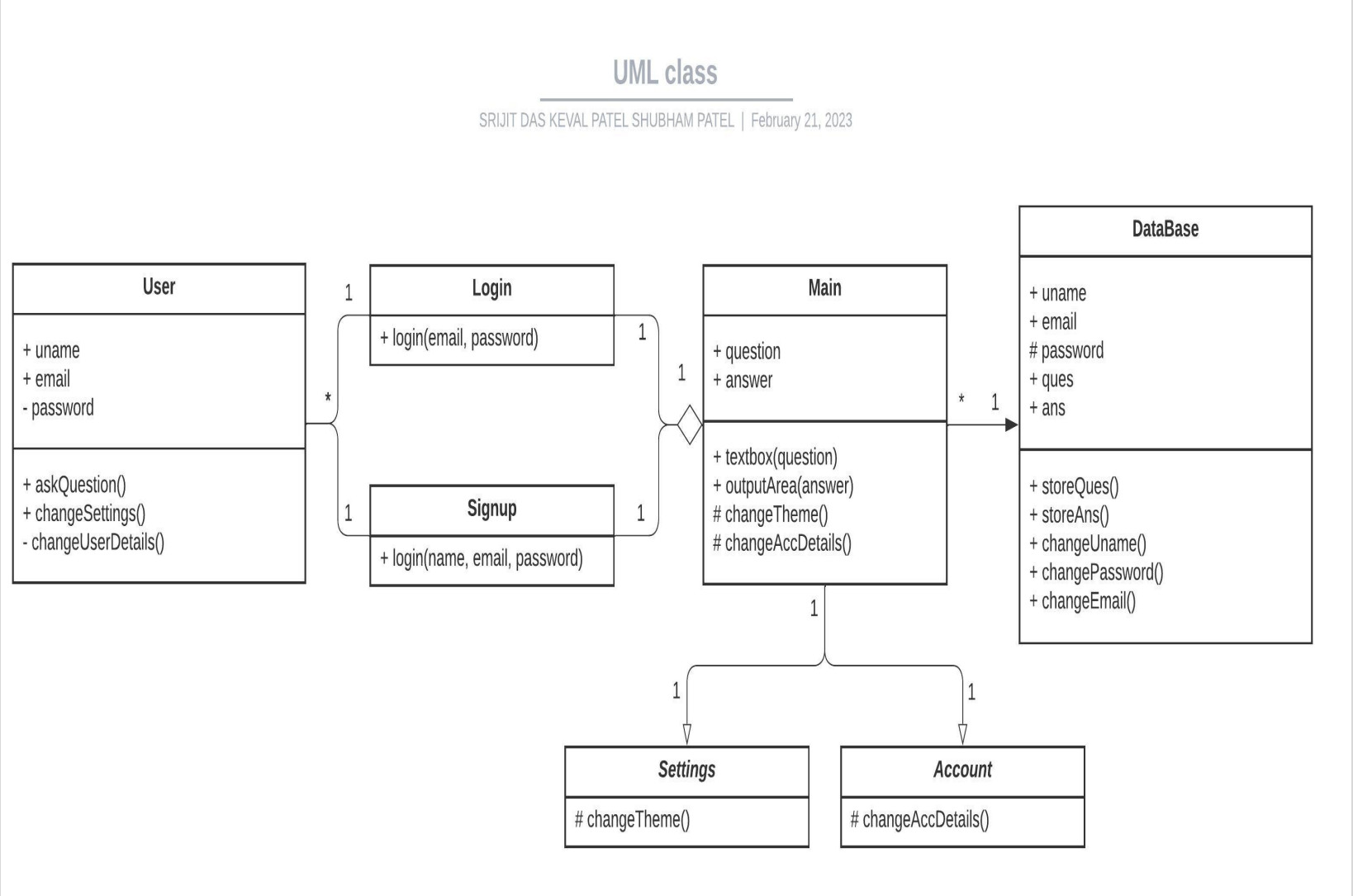
DIAGRAMS

- Activity Diagram



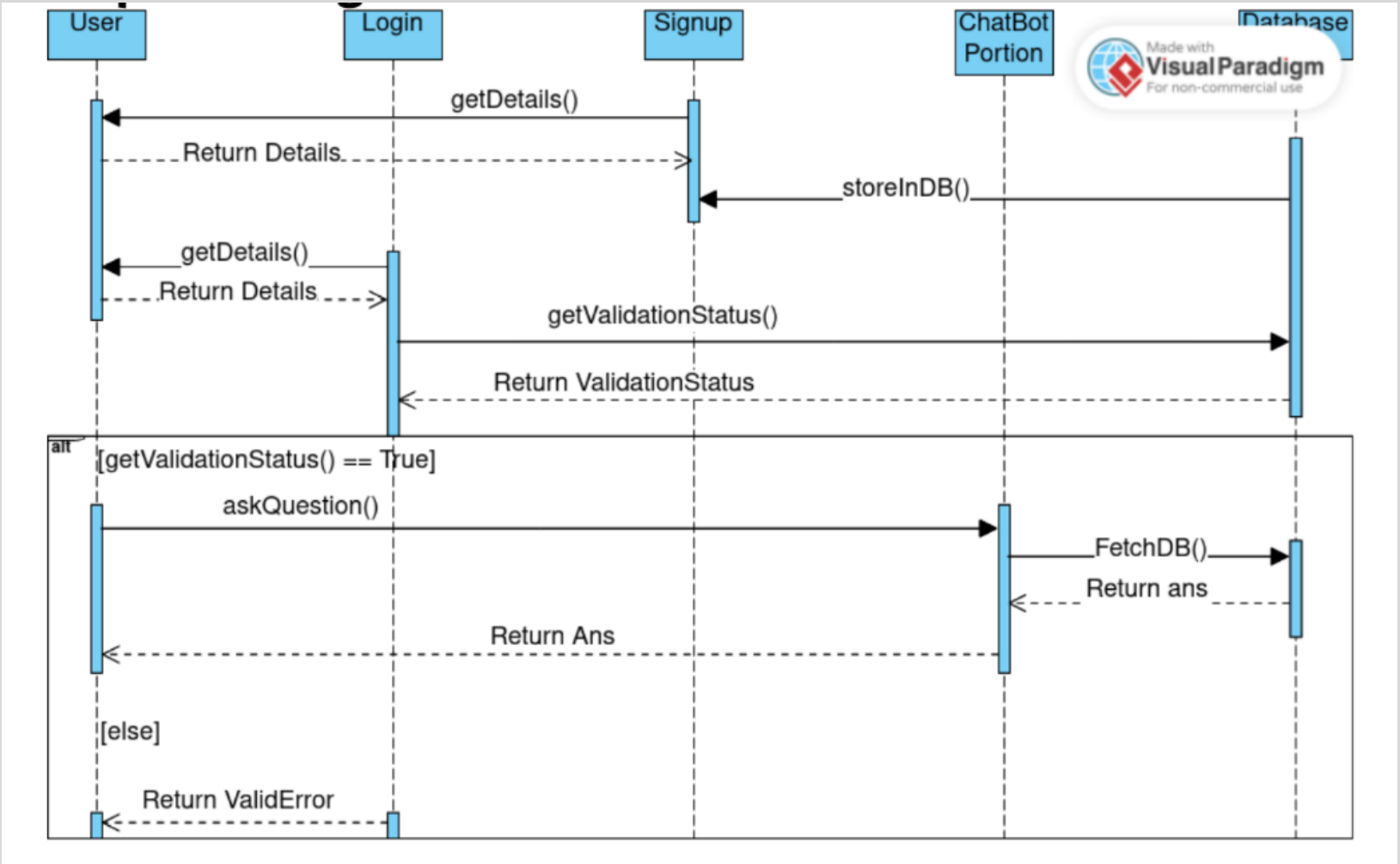
DIAGRAMS

- Class Diagram



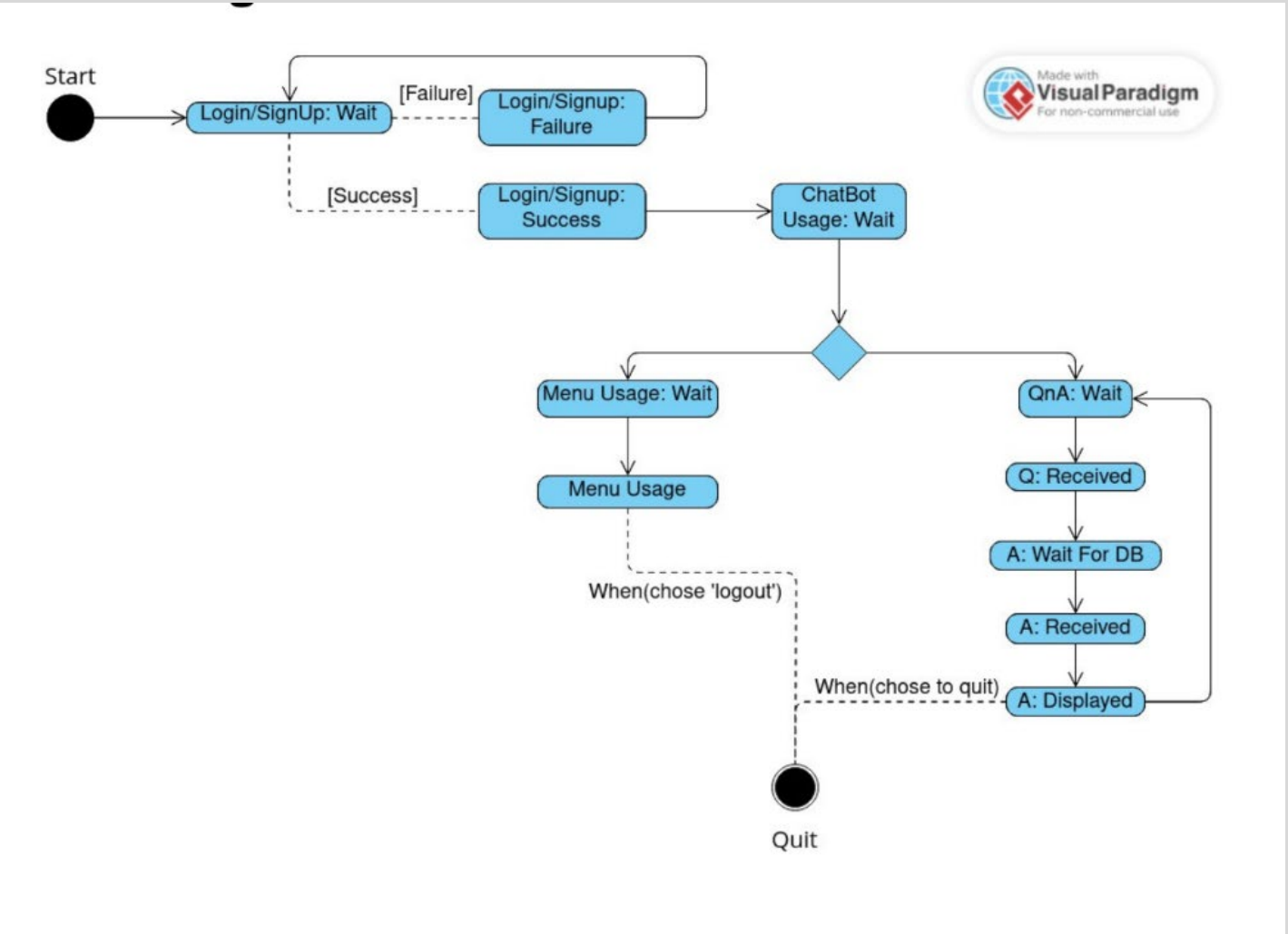
DIAGRAMS

- Sequence Diagram

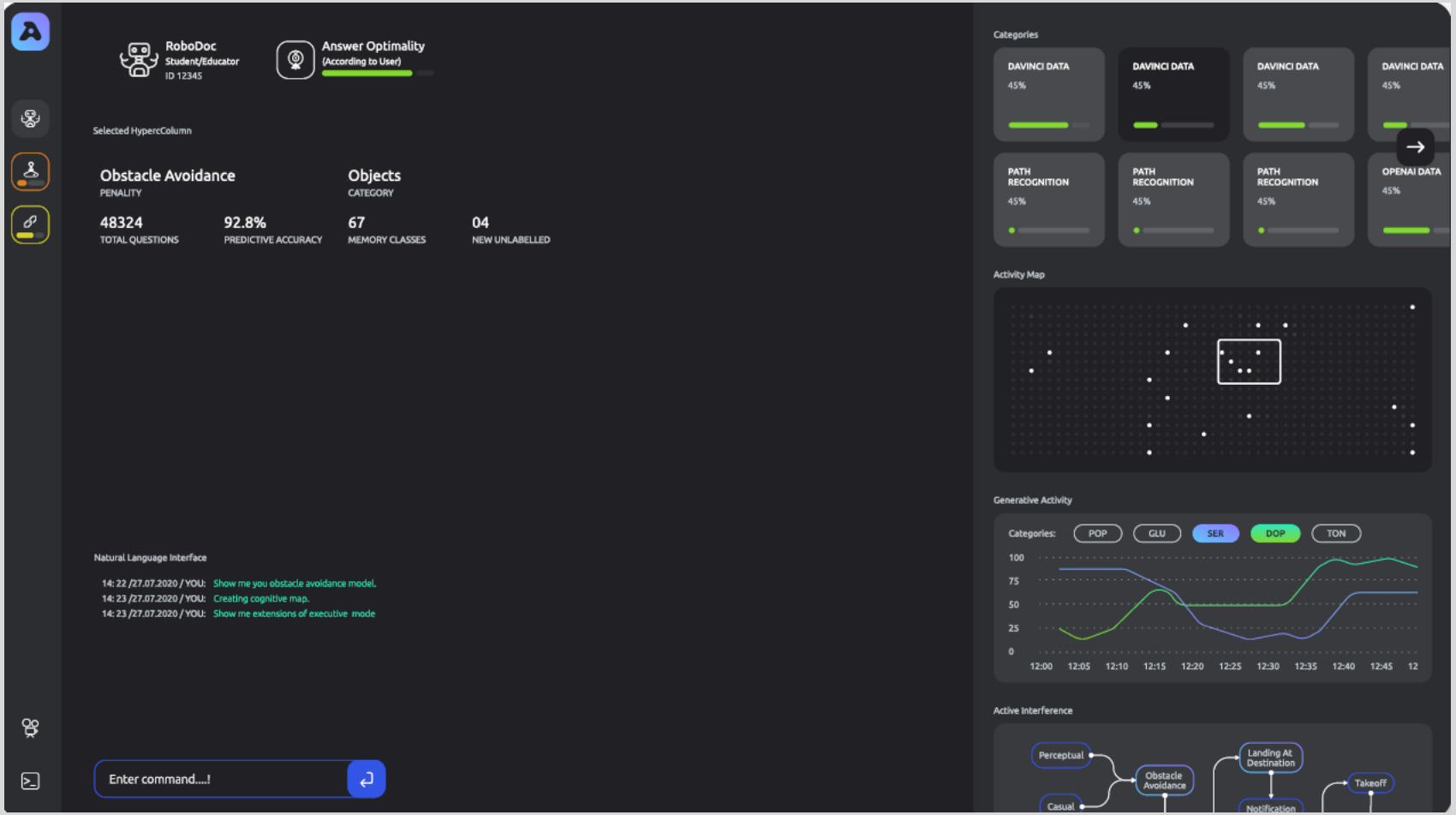


DIAGRAMS

- State Diagram



WireFrame


RoboDoc
Student/Educator
ID 12345
Answer Optimality
(According to User)
Selected HypercColumn
Obstacle Avoidance
PENALITY
Objects
CATEGORY
48324
TOTAL QUESTIONS
92.8%
PREDICTIVE ACCURACY
67
MEMORY CLASSES
04
NEW UNLABELLED
Natural Language Interface
14: 22 /27.07.2020 / YOU: Show me you obstacle avoidance model.
14: 23 /27.07.2020 / YOU: Creating cognitive map.
14: 23 /27.07.2020 / YOU: Show me extensions of executive mode
Enter command....!
Categories
DAVINCI DATA
45%
PATH RECOGNITION
OPENAI DATA
Activity Map
Generative Activity
Categories:
POP
GLU
SER
DOP
TON
100
75
50
25
0
12:00
12:05
12:10
12:15
12:20
12:25
12:30
12:35
12:40
12:45
Active Interference
Perceptual
Casual
Obstacle Avoidance
Landing At Destination
Notification
Takeoff

Code

project.py > ...

```python
import tkinter as tk
import tkinter.messagebox as tkm
import customtkinter as ctk
import os
import openai
import sqlite3
from datetime import datetime
import pytz

# Connect to openAI
API_KEY = "***"
os.environ['OPENAI_Key'] = API_KEY
openai.api_key = os.environ['OPENAI_Key']

# set app theme
ctk.set_appearance_mode("Dark")
ctk.set_default_color_theme("blue")

# Connect to the database
conn = sqlite3.connect('chatbot.db')
c = conn.cursor()
# Set timezone to India (GMT+5:30)
timezone = pytz.timezone('Asia/Kolkata')
# Create the table to store chat history if it doesn't exist
```

Code

```
c.execute('''
    CREATE TABLE IF NOT EXISTS chat_history (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        query TEXT NOT NULL,
        response TEXT NOT NULL,
        created_at TEXT DEFAULT (datetime('now', 'utc'))
    )
''')
conn.commit()


# Main app class
class App(ctk.CTk):

    def __init__(self):
        super().__init__()

        # configure window
        self.title("ChatBot")
        self.geometry(f"{1600}x{900}")

        # configure grid layout (4x4)
        self.grid_columnconfigure(1, weight=1)
        self.grid_columnconfigure((2, 3), weight=0)
        self.grid_rowconfigure(0, weight=1)
```

Code

```
51          # create sidebar frame with widgets
52          self.sidebar_frame = ctk.CTkFrame(self, width=300, corner_radius=10)
53          self.sidebar_frame.grid(row=0, column=0, rowspan=4, sticky="nsew")
54          self.sidebar_frame.grid_rowconfigure(4, weight=1)
55          self.logo_label = ctk.CTkLabel(self.sidebar_frame,
56                                         text="ChatBot",
57                                         font=ctk.CTkFont(size=24,
58                                                          weight="bold"))
59          self.logo_label.grid(row=0, column=0, padx=20, pady=(20, 10))
60          self.sidebar_button_1 = ctk.CTkButton(
61              self.sidebar_frame,
62              font=ctk.CTkFont(size=16, weight="normal"),
63              height=50,
64              command=lambda: self.clear_history(),
65              text="Clear History")
66          self.sidebar_button_1.grid(row=1, column=0, padx=20, pady=10)
67          self.appearance_mode_label = ctk.CTkLabel(self.sidebar_frame,
68                                                    text="Appearance Mode:",
69                                                    anchor="w")
70          self.appearance_mode_label.grid(row=5, column=0, padx=20, pady=(10, 0))
71          self.appearance_mode_optionemenu = ctk.CTkOptionMenu(
72              self.sidebar_frame,
73              values=["Light", "Dark", "System"],
74              command=self.change_appearance_mode_event)
```

Code

```
        command=self.change_appearance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6,
                                              column=0,
                                              padx=20,
                                              pady=(10, 10))
        self.scaling_label = ctk.CTkLabel(self.sidebar_frame,
                                          text="UI Scaling:",
                                          anchor="w")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10, 0))
        self.scaling_optionemenu = ctk.CTkOptionMenu(
            self.sidebar_frame,
            values=["80%", "90%", "100%", "110%", "120%"],
            command=self.change_scaling_event)
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(10, 20))

        # History Bar
        self.sidebar_frame = ctk.CTkFrame(self, width=300, corner_radius=10)
        self.sidebar_frame.grid(row=0, column=3, rowspan=4, sticky="nsew")
        self.sidebar_frame.grid_rowconfigure(2, weight=1)
        self.logo_label = ctk.CTkLabel(self.sidebar_frame,
                                       text="History",
                                       font=ctk.CTkFont(size=20,
                                                        weight="normal"))
        self.logo_label.grid(row=0, column=0, padx=60, pady=(20, 0))

```

Code

```
100     self.history_text = ctk.CTkTextbox(
101         self.sidebar_frame,
102         height=ctk.CTk.winfo_screenheight(self),
103         width=800,
104         font=ctk.CTkFont(size=14, weight="bold"))
105     self.history_text.grid(row=1, column=0, padx=10, pady=10)
106
107     # Add content to the Text widget
108     self.entry = ctk.CTkEntry(self, placeholder_text="Ask Here")
109     self.entry.grid(row=3,
110                     column=1,
111                     columnspan=2,
112                     padx=(20, 20),
113                     pady=(20, 20),
114                     sticky="nsew")
115
116     # create main button
117     self.main_button_1 = ctk.CTkButton(master=self,
118                                        width=100,
119                                        fg_color="transparent",
120                                        border_width=2,
121                                        text="Search",
122                                        text_color=("gray10", "#DCE4EE"),
123                                        command=lambda: self.ans())
124
```

Code

```
125         self.main_button_1.grid(row=3,
126                                  column=3,
127                                  columnspan=1,
128                                  padx=(250, 250),
129                                  pady=(20, 20),
130                                  sticky="nsew")
131 
132         # create textbox
133         self.textbox = ctk.CTkTextbox(self,
134                                       width=250,
135                                       font=ctk.CTkFont(size=16))
136 
137         self.textbox.grid(row=0,
138                           column=1,
139                           padx=(20, 20),
140                           pady=(20, 0),
141                           sticky="nsew")
142 
143         # set default values
144         self.appearance_mode_optionemenu.set("Dark")
145         self.scaling_optionemenu.set("100%")
146         self.textbox.insert("0.0", "Answers Here\n\n")
147         self.textbox.configure(state="disabled")
148 
```

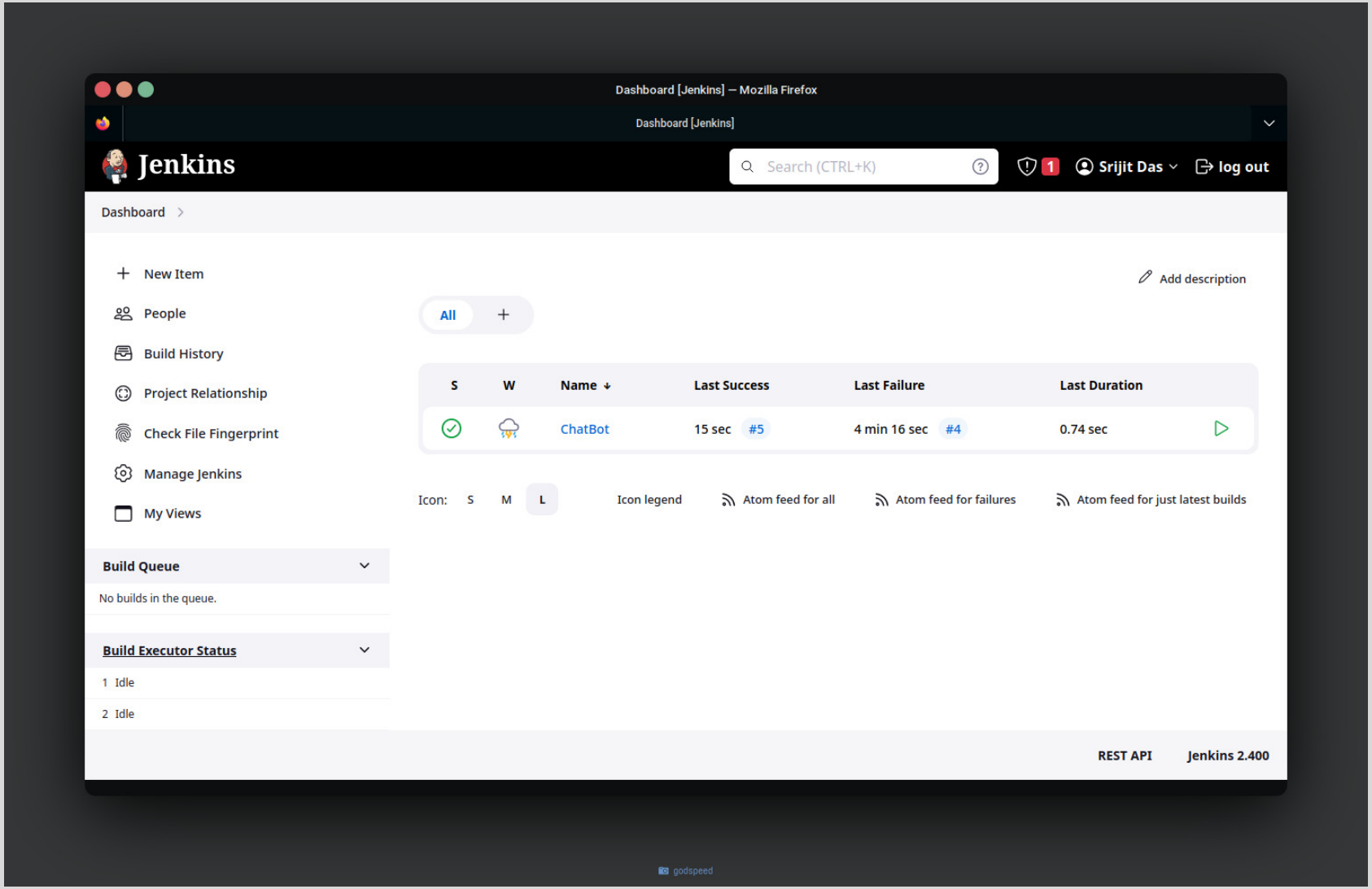
Code

```
149     def change_appearance_mode_event(self, new_appearance_mode: str):
150         ctk.set_appearance_mode(new_appearance_mode)
151         self.sidebar_frame.set_appearance_mode(new_appearance_mode)
152 
153     def change_scaling_event(self, new_scaling: str):
154         new_scaling_float = int(new_scaling.replace("%", "")) / 100
155         ctk.set_widget_scaling(new_scaling_float)
156         self.sidebar_frame.set_widget_scaling(new_scaling_float)
157 
158     def clear_history(self):
159         c.execute("DELETE FROM chat_history")
160         conn.commit()
161         self.history()
162 
163     def ans(self):
164         self.textbox.configure(state="normal")
165         self.textbox.delete("0.0", "end")
166         self.textbox.insert("0.0",
167                             "\n\n" + str(self.get_response(self.entry.get())))
168         self.textbox.configure(state="disabled")
169 
170     def history(self):
171         c.execute("SELECT * FROM chat_history ORDER BY created_at DESC")
172         rows = c.fetchall()
173         print(rows)
174         self.history_text.configure(state="normal")
175         self.history_text.delete("0.0", "end")
```

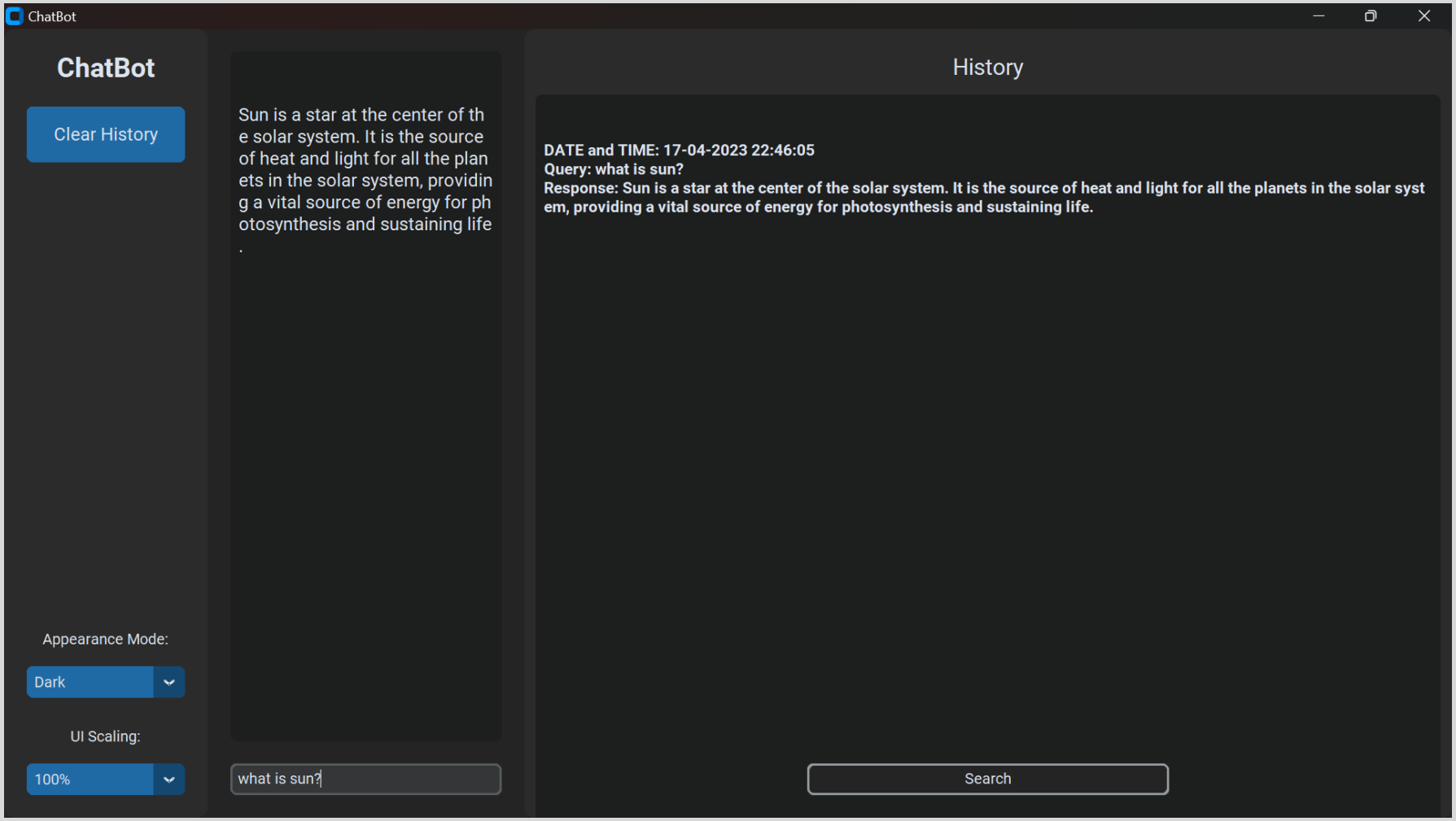
Code

```
175             self.history_text.delete("0.0", "end")
176             for row in rows:
177                 self.history_text.insert(ctk.END, f"\n\nDATE and TIME: {row[3]}\n")
178                 self.history_text.insert(ctk.END, f"Query: {row[1]}\n")
179                 self.history_text.insert(ctk.END, f"Response: {row[2]}\n\n")
180             self.history_text.configure(state="disabled")
181
182         def get_response(self, prompt):
183             response = openai.Completion.create(model='text-davinci-003',
184                                                 prompt=prompt,
185                                                 max_tokens=1000)
186
187             answer = response.choices[0].text.strip()
188
189             # Store the chat history in the database with current time in India
190             current_time = datetime.now(timezone)
191             current_time = current_time.strftime("%d-%m-%Y %H:%M:%S")
192             c.execute(
193                 'INSERT INTO chat_history (query, response, created_at) VALUES (?, ?, ?)',
194                 (self.entry.get(), answer, current_time))
195             conn.commit()
196             self.history()
197             return (answer)
198
199
200     if __name__ == "__main__":
201         app = App()
202         app.history()
203         app.mainloop()
```

Testing


Dashboard [Jenkins] — Mozilla Firefox
Dashboard [Jenkins]
Jenkins
Search (CTRL+K)
1
Srijit Das
log out
Dashboard
New Item
People
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Add description
All
S
W
Name
Last Success
Last Failure
Last Duration
ChatBot
15 sec #5
4 min 16 sec #4
0.74 sec
Icon: S M L
Icon legend
Atom feed for all
Atom feed for failures
Atom feed for just latest builds
REST API
Jenkins 2.400
godspeed

Output


ChatBot
ChatBot
Clear History
Sun is a star at the center of th e solar system. It is the source of heat and light for all the plan ets in the solar system, providin g a vital source of energy for ph otosynthesis and sustaining life .
Appearance Mode:
Dark
UI Scaling:
100%
what is sun?
History
DATE and TIME: 17-04-2023 22:46:05
Query: what is sun?
Response: Sun is a star at the center of the solar system. It is the source of heat and light for all the planets in the solar syst em, providing a vital source of energy for photosynthesis and sustaining life.
Search