RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE

CB23332
SOFTWARE ENGINEERING LAB

Laboratory Record Note Book

Name : . . . . . .

Year / Branch / Section : . . . . . .

Register No. : . . . . . .

Semester : . . . . . .

Academic Year : . . . . . .

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

RAJALAKSHMI NAGAR, THANDALAM – 602-105

BONAFIDE CERTIFICATE

NAME:_____**REGISTER NO.:** _____

ACADEMIC YEAR: 2024-25 **SEMESTER:** III **BRANCH:**_____B.E/B.Tech

This Certification is the bonafide record of work done by the above student in the

CB23332-SOFTWARE ENGINEERING - Laboratory during the year 2024 – 2025.

Signature of Faculty -in – Charge

Submitted for the Practical Examination held on _____

Internal Examiner

External Examiner

INDEX

S. No.	Name of the Experiment	Expt. Date	Faculty Sign
1.	Preparing Problem Statement		
2.	Software Requirement Specification (SRS)		
3.	Entity-Relational Diagram		
4.	Data Flow Diagram		
5.	Use Case Diagram		
6.	Activity Diagram		
7.	State Chart Diagram		
8.	Sequence Diagram		
9.	Collaboration Diagramt		
10.	Class Diagram		

EX NO:1	WRITE THE COMPLETE PROBLEM STATEMENT
DATE:	

AIM:

To prepare PROBLEM STATEMENT for conversational image recognition chatbot

ALGORITHM:

1. **Identify the Problem**
    - Define the central issue that the project aims to address. For the conversational image recognition chatbot, focus on issues users face with image interpretation accuracy, especially in complex scenarios.
2. **Draft an Initial Problem Statement**
    - Formulate a concise statement describing the problem without suggesting a specific solution. Ensure this statement is broad enough to provide room for exploring multiple solutions.
3. **Gather User Feedback and Background Information**
    - Collect insights from user feedback, surveys, and system logs to validate the problem. Describe the impact of the problem on user experience and overall system reliability.
4. **Outline the Background**
    - Summarize key observations that highlight why the issue exists, providing context such as recent test results or usage data that demonstrate the scale of the problem.
5. **Explain the Relevance**
    - Detail why solving this problem is critical for the project's success. Emphasize how the problem affects users and the importance of achieving improved image recognition and response accuracy.
6. **Set Clear Objectives**
    - Define specific goals for the project, such as enhancing image recognition capabilities, optimizing natural language processing, refining response generation, and implementing feedback mechanisms.
7. **Determine Desired Outcomes**
    - Outline the desired result, such as reducing user dissatisfaction rates to below 15%. Establish criteria for evaluating the project's success, including user feedback and performance metrics.
8. **Prepare the Final Problem Statement**
    - Combine all the elements to create a detailed problem statement that includes the problem, background, relevance, objectives, and expected outcomes.

Input:

- Problem Statement Prepared by Customer:

The customer provides a preliminary problem statement highlighting challenges users face with the chatbot's image recognition and response accuracy. This statement may include user feedback data, specific pain points, and broad expectations for improvement in user satisfaction and response relevance.

-Overview of Existing System:

A high-level description of the current chatbot, including its capabilities and limitations. This may cover the chatbot's existing image recognition techniques, response generation algorithms, and any known issues based on previous testing or user reports.

- Broad Expectations from the New System:

- Improved accuracy in recognizing and interpreting complex images.
- Enhanced natural language processing to understand user queries more effectively.
- Increased user satisfaction and engagement through relevant and accurate responses.
- A structured feedback mechanism to gather user input for ongoing improvement.

- Requirements Elicitation:

- Requirements Gathering: Involves discussions with the customer to identify critical functionalities and areas of improvement in image recognition and response generation.

- Customer and Existing System Processes: Engaging with customers and reviewing the existing system processes to understand gaps and potential enhancements, aligning with both technical and user expectations.

Problem Statement:

During recent usability tests, 40% of users reported dissatisfaction with the accuracy of responses provided by the conversational image recognition chatbot when querying image content. The chatbot struggles with accurately identifying objects and scenes in complex or unclear images, which diminishes user trust and satisfaction. The goal is to reduce user-reported dissatisfaction to below 15% by improving image recognition accuracy and the relevance of responses

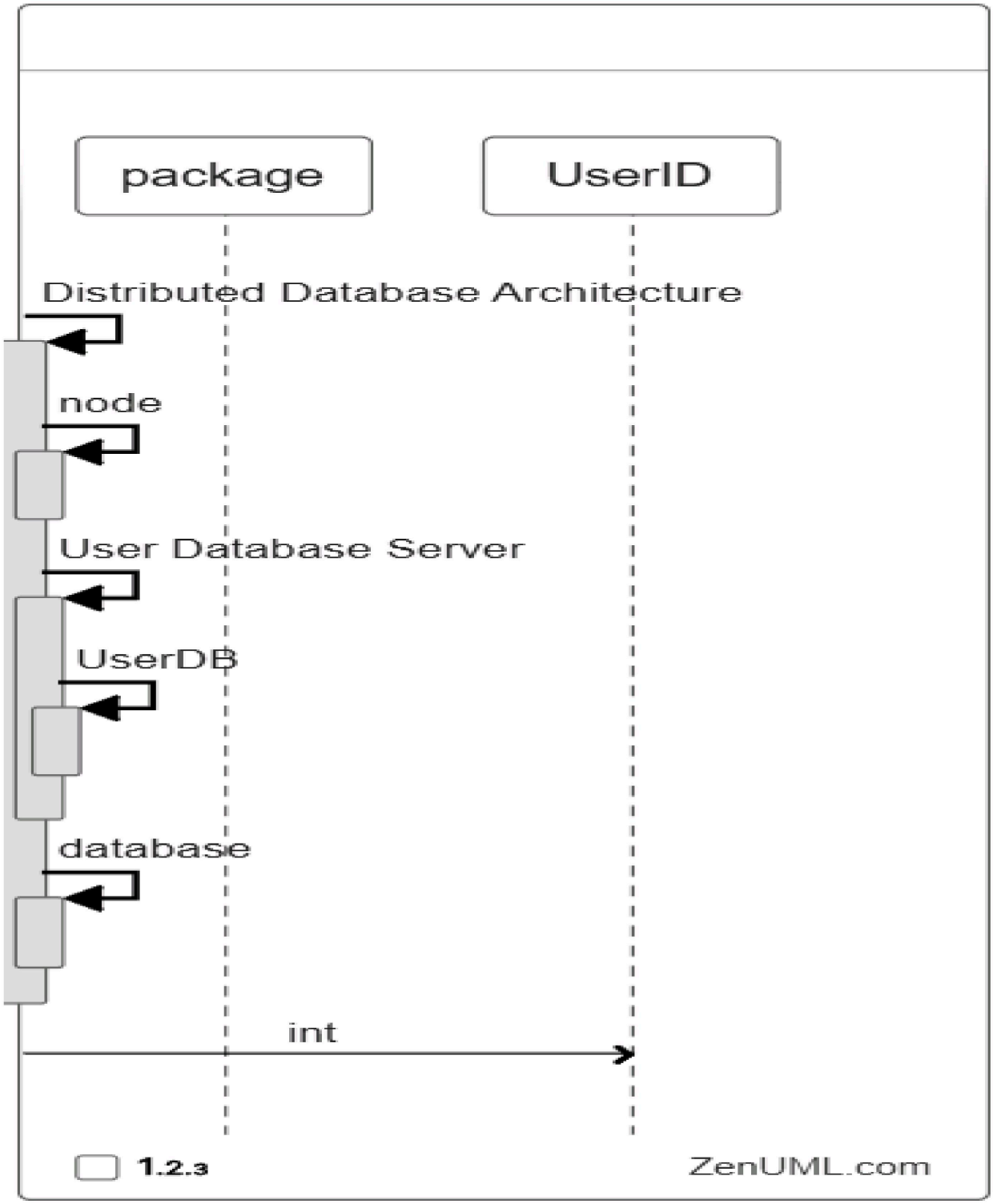
Background Relevance:

The conversational image recognition chatbot, designed to assist users in retrieving detailed information about visual content through natural language queries, faces a significant challenge with user satisfaction. Users often receive inaccurate or incomplete responses for complex image queries, leading to frustration and reduced trust in the tool. This issue has been highlighted in recent user feedback, showing a dissatisfaction rate of 40%. To achieve meaningful improvement, the chatbot requires enhanced recognition capabilities, more accurate natural language understanding, and a streamlined response generation process. Effective image recognition and response generation are essential to the chatbot's functionality and user experience. Accurate image-based answers encourage user engagement and establish trust, while inaccurate responses can lead to user frustration and diminished usage. By addressing these challenges, the chatbot can improve its overall reliability, user satisfaction, and engagement levels, reinforcing its value as an image interpretation tool.

Objectives:

1. **Improving Image Recognition Capabilities**
    - Upgrading the image recognition engine to improve its ability to analyze and interpret complex image content.
    - Implementing machine learning models that can accurately identify a wider range of objects, scenes, and contexts in various image types.
2. **Enhancing Natural Language Processing (NLP)**
    - Optimizing the query processor to better interpret user queries and extract relevant details.
    - Training the NLP models to understand contextual nuances, so responses can better match user expectations for varied query types.
3. **Optimizing Response Generation**
    - Developing a more intelligent response generation system to provide clear, accurate answers based on the recognized image content and user query.
    - Implementing multi-response generation capabilities, offering users alternative interpretations or additional details.
4. **Refining User Feedback Mechanisms**
    - Creating a robust feedback loop that captures user dissatisfaction and reasons for inaccuracies, helping continuously improve system performance.
    - Incorporating user feedback directly into the image recognition and response systems for real-time improvements.
5. **Monitoring and Evaluation**
    - Regularly assessing improvements through user surveys, satisfaction scores, and key performance indicators (KPIs) related to accuracy and response relevance.
    - Using metrics such as satisfaction rate, error rate in response generation, and image recognition success rate to track progress.
6. **Sustaining Long-term Improvement**
    - Establishing an ongoing maintenance framework to continuously update models and algorithms in response to evolving image recognition challenges and user needs.

Result:


package
UserID
Distributed Database Architecture
node
User Database Server
UserDB
database
int
1.2.3
ZenUML.com

EX NO:2	WRITE THE SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT
DATE:	

AIM:

To do requirement analysis and develop Software Requirement Specification Sheet(SRS) for conversational image recognition chatbot.

ALGORITHM:

SRS shall address are the following:

a) **Functionality.** What is the software supposed to do?

b) **External interfaces.** How does the software interact with people, the system‘s hardware, other hardware, and other software?

c) **Performance.** What is the speed, availability, response time, recovery time of various software functions, etc.?

d) **Attributes.** What is the portability, correctness, maintainability, security, etc. considerations?

e) **Design constraints imposed on an implementation.** Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.?

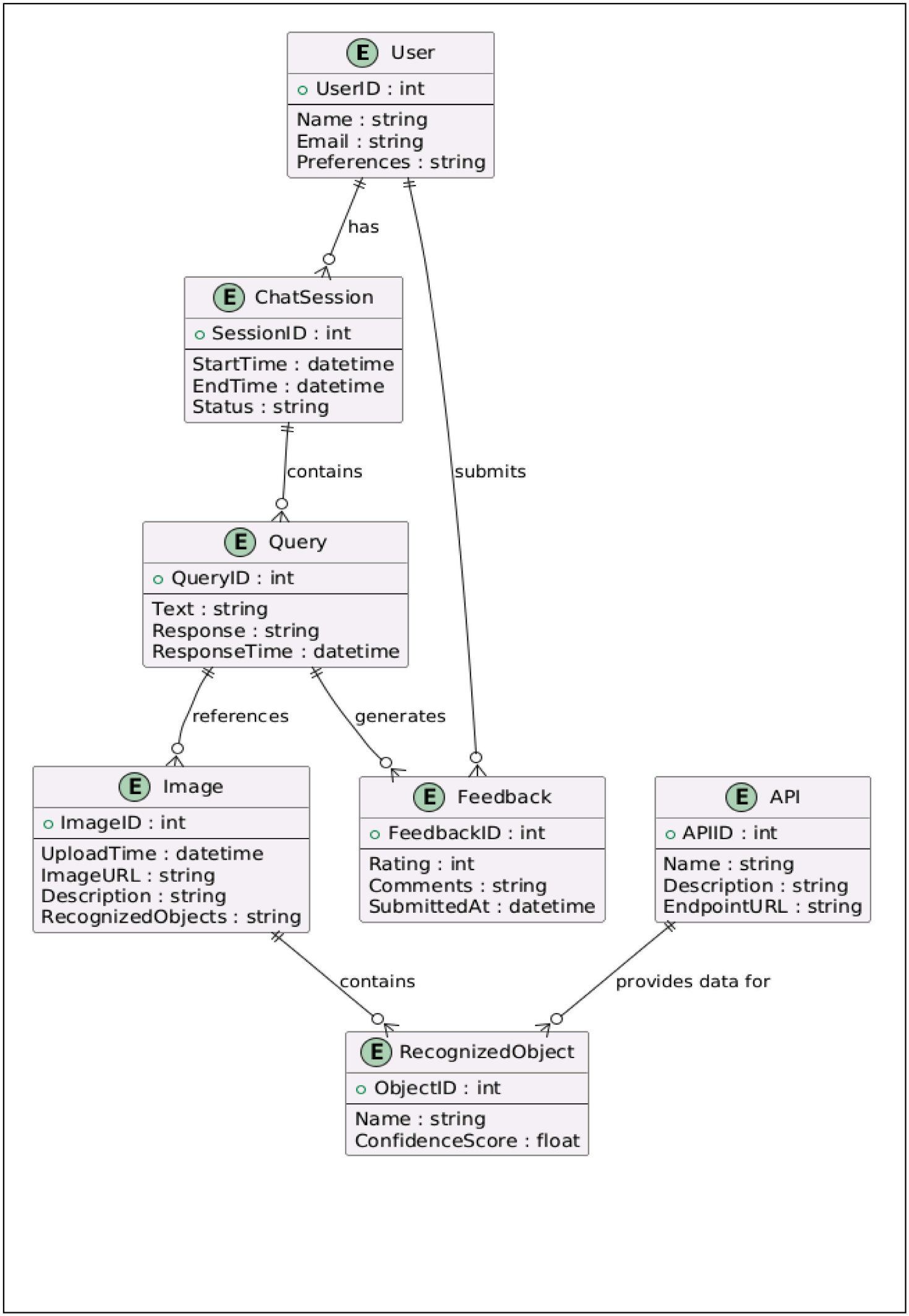
1. Introduction

1.1 Purpose

Define the goal of the conversational image recognition chatbot, which is to provide users with real-time image recognition coupled with a conversational interface for interactive and personalized responses.

1.2 Document Conventions

- Use standard terminology for image recognition, AI, and chatbot functionalities. Key terms such as "image recognition," "NLP (Natural Language Processing)," "confidence level," and "user feedback" are defined for clarity.


User
UserID : int
Name : string
Email : string
Preferences : string
has
ChatSession
SessionID : int
StartTime : datetime
EndTime : datetime
Status : string
contains
submits
Query
QueryID : int
Text : string
Response : string
ResponseTime : datetime
references
generates
Image
ImageID : int
UploadTime : datetime
ImageURL : string
Description : string
RecognizedObjects : string
Feedback
FeedbackID : int
Rating : int
Comments : string
SubmittedAt : datetime
API
APIID : int
Name : string
Description : string
EndpointURL : string
contains
provides data for
RecognizedObject
ObjectID : int
Name : string
ConfidenceScore : float

1.3. Intended Audience and Reading Suggestions

- Targeted for developers, project managers, UX designers, and end-users interested in understanding the software's functionality and technical requirements. Readers with expertise in AI, machine learning, and chatbot development will find relevant sections in technical detail.

1.4. Project Scope

- The chatbot will provide users with a platform to upload images and receive insights and object identification through conversational responses. It will support general image recognition tasks (e.g., identifying objects, scenes) and offer follow-up answers to user queries, enhancing user engagement and providing real-time information.

1.5. References

- Cite research papers, image recognition API documentation (such as Google Vision, AWS Recognition), and conversational AI frameworks (such as Dialogflow, Rasa) used in the project.

2. Overall Description

2.1 Product Perspective

This chatbot is designed as a standalone application but can also integrate into popular messaging platforms (e.g., WhatsApp, Messenger). It relies on cloud-based image recognition and NLP services to interpret images and respond conversationally.

2.2 Product Features

- Image recognition to identify objects, text, scenes, and people.
- Contextual conversation based on image content and user queries.
- Feedback loop for users to rate recognition accuracy.
- Integration with popular messaging services.

2.3 User Class and Characteristics

The chatbot is aimed at users of all ages and tech familiarity who want real-time image analysis and conversational assistance. Potential users include general consumers, students, and visually impaired individuals who require image descriptions.

2.4 Operating Environment

The chatbot will operate on cloud platforms such as AWS or Azure, accessible via mobile and desktop web browsers, and popular messaging applications with internet connectivity.

2.5 Design and Implementation Constraints

Development must consider data privacy laws (GDPR) for image data, latency in real-time responses, and security standards to protect user information.

2.6 Assumption Dependencies

It’s assumed that users will have internet access and basic knowledge of messaging interfaces. Additionally, the chatbot’s performance will depend on third-party image recognition and NLP services.

3. Specific Requirements

Description and Priority

- **Image Recognition and Description**: Recognize and describe images in real-time with a high level of accuracy.
- **Conversational AI**: Answer user questions about recognized objects, scenes, or text within the image.
- **Personalization**: Track user preferences and use conversational memory for tailored responses.
- **Feedback Collection**: Collect user feedback on response accuracy to improve the model over time.

4. External Interface Requirements

4.1 User Interfaces

- Simple and intuitive interface for uploading images and engaging in text-based chat. Include buttons for uploading images, a feedback mechanism, and text input for queries.

4.2 Hardware Interfaces

- Device compatibility with cameras for real-time image capture and upload Compatible with mobile and desktop devices.

4.3 Software Interfaces

- Integrate with image recognition APIs (e.g., Google Vision), NLP frameworks (e.g., Dialogflow), and user authentication systems (OAuth 2.0).

4.4 Communication Interfaces

- Use HTTPS for secure image and data transmission, supporting real-time message delivery across different messaging platforms.

5. Additional Requirements

5.1 Performance Requirements

The chatbot should process and respond to text queries within 2 seconds and image-based queries within 5 seconds. It must handle up to 1,000 concurrent users without performance degradation.

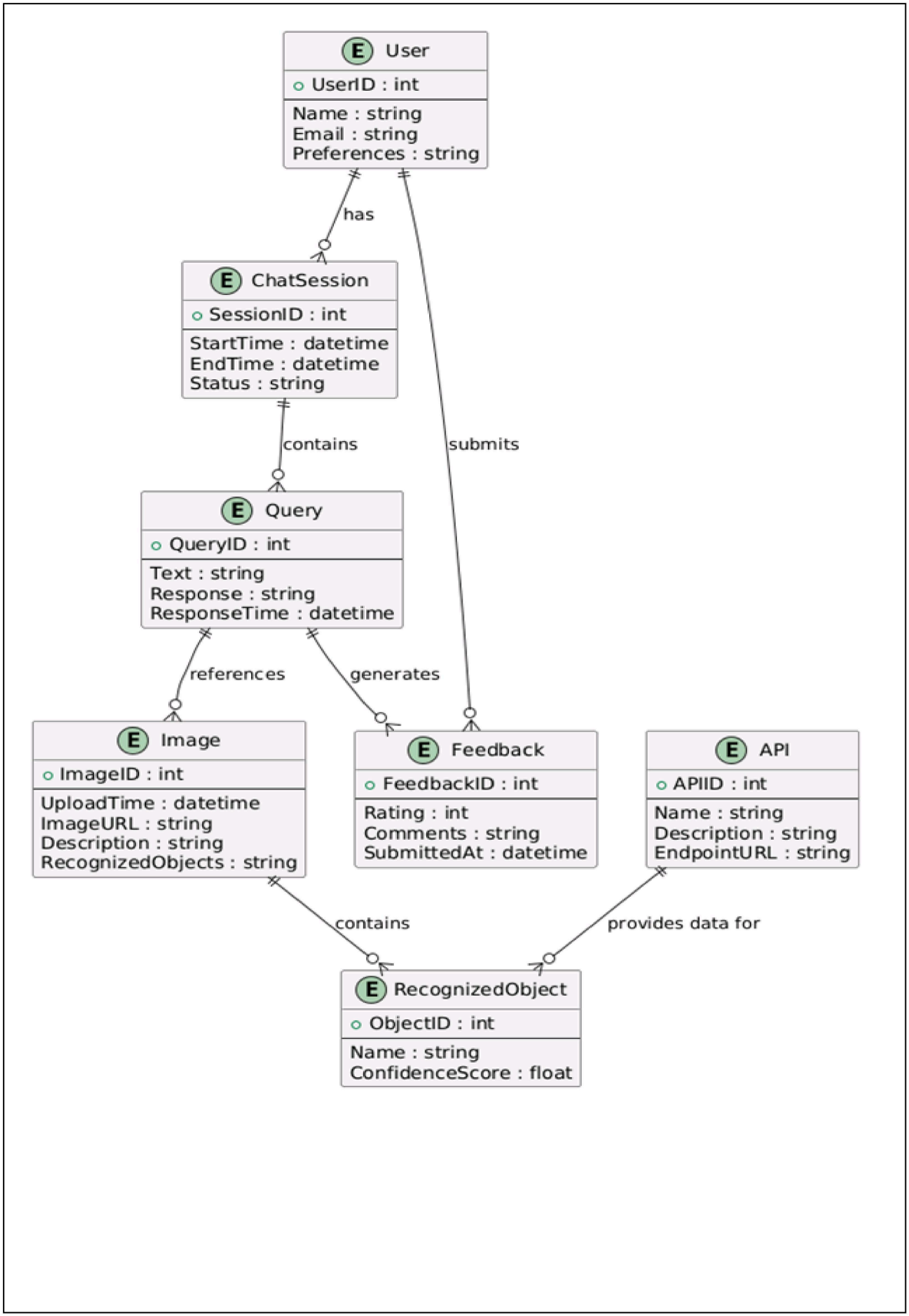
5.2 Safety Requirements

Prevent users from sharing sensitive or inappropriate content through automated content filtering mechanisms and ensure safe conversational interactions.

5.3 Security Requirements

Implement data encryption for all user images and messages, with secure user authentication, and comply with data privacy laws, ensuring data storage and processing follow legal standards.

Result:


User
UserID : int
Name : string
Email : string
Preferences : string
has
ChatSession
SessionID : int
StartTime : datetime
EndTime : datetime
Status : string
contains
submits
Query
QueryID : int
Text : string
Response : string
ResponseTime : datetime
references
generates
Image
ImageID : int
UploadTime : datetime
ImageURL : string
Description : string
RecognizedObjects : string
Feedback
FeedbackID : int
Rating : int
Comments : string
SubmittedAt : datetime
API
APIID : int
Name : string
Description : string
EndpointURL : string
contains
provides data for
RecognizedObject
ObjectID : int
Name : string
ConfidenceScore : float

EX NO:3	DRAW THE ENTITY RELATIONSHIP DIAGRAM
DATE:	

AIM:

To Draw the Entity Relationship Diagram for conversational image recognition chatbot

ALGORITHM:

Step 1: Mapping of Regular Entity Types

Step 2: Mapping of Weak Entity Types

Step 3: Mapping of Binary 1:1 Relation Types

Step 4: Mapping of Binary 1:N Relationship Types.

Step 5: Mapping of Binary M:N Relationship Types.

Step 6: Mapping of Multivalued attributes.

INPUT:

Entities

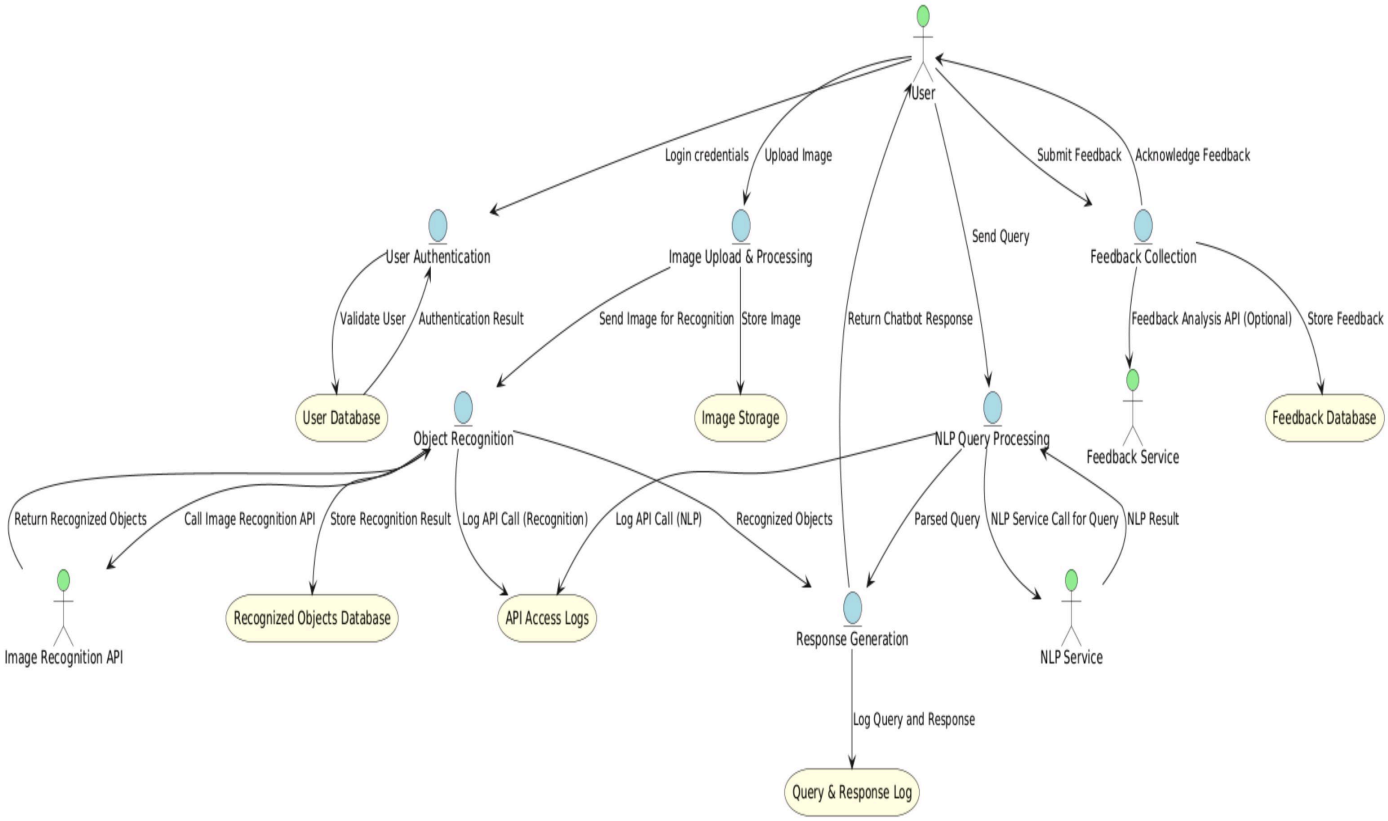
Entity Relationship Matrix

Primary Keys

Attributes

Mapping of Attributes with Entities

Result:


User
Login credentials
Upload Image
Submit Feedback
Acknowledge Feedback
User Authentication
Image Upload & Processing
Send Query
Feedback Collection
Validate User
Authentication Result
Send Image for Recognition
Store Image
Return Chatbot Response
Feedback Analysis API (Optional)
Store Feedback
User Database
Object Recognition
Image Storage
NLP Query Processing
Feedback Service
Feedback Database
Return Recognized Objects
Call Image Recognition API
Store Recognition Result
Log API Call (Recognition)
Log API Call (NLP)
Recognized Objects
Parsed Query
NLP Service Call for Query
NLP Result
Image Recognition API
Recognized Objects Database
API Access Logs
Response Generation
NLP Service
Log Query and Response
Query & Response Log

EX NO:4	DRAW THE DATA FLOW DIAGRAMS AT LEVEL 0 AND LEVEL 1
DATE:	

AIM:

To Draw the Data Flow Diagram for any project and List the Modules in the Application.

ALGORITHM:

1. Open the Visual Paradigm to draw DFD (Ex.Lucidchart)
2. Select a data flow diagram template
3. Name the data flow diagram
4. Add an external entity that starts the process
5. Add a Process to the DFD
6. Add a data store to the diagram
7. Continue to add items to the DFD
8. Add data flow to the DFD
9. Name the data flow
10. Customize the DFD with colours and fonts
11. Add a title and share your data flow diagram

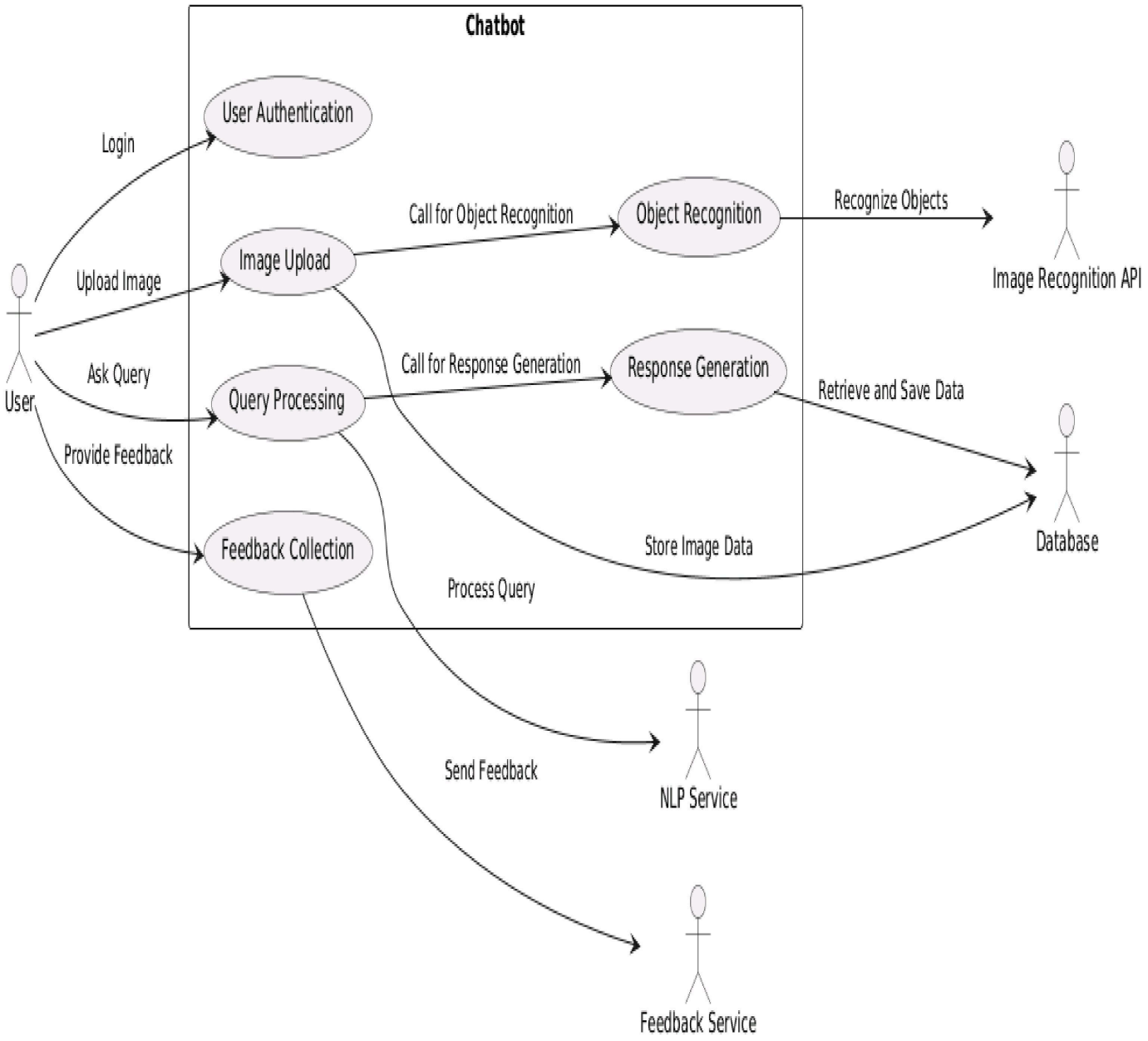
INPUT:

Processes

Datastores

External Entities

Result:


Chatbot
User Authentication
Login
Image Upload
Upload Image
Query Processing
Ask Query
Feedback Collection
Provide Feedback
User
Object Recognition
Call for Object Recognition
Recognize Objects
Image Recognition API
Response Generation
Call for Response Generation
Retrieve and Save Data
Database
Store Image Data
Process Query
NLP Service
Send Feedback
Feedback Service

EX NO:5	DRAW USE CASE DIAGRAM
DATE:	

AIM:

To Draw the Use Case Diagram for any project

ALGORITHM:

Step 1: Identify Actors

Step 2: Identify Use Cases

Step 3: Connect Actors and Use Cases

Step 4: Add System Boundary

Step 5: Define Relationships

Step 6: Review and Refine

Step 7: Validate

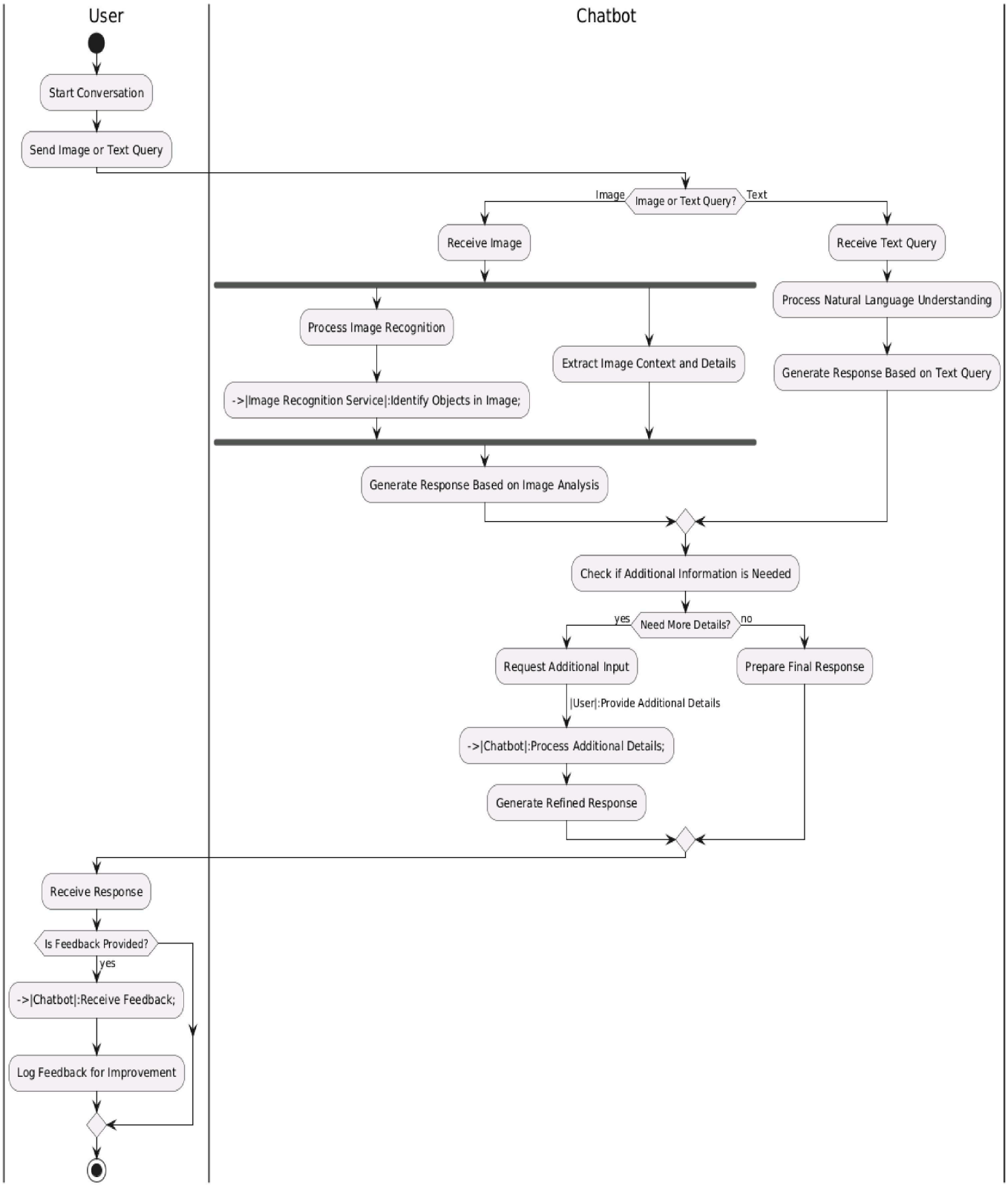
INPUTS:

Actors

Use Cases

Relations

Result:


User
Chatbot
Start Conversation
Send Image or Text Query
Image or Text Query?
Image
Text
Receive Image
Receive Text Query
Process Image Recognition
Extract Image Context and Details
Process Natural Language Understanding
Generate Response Based on Text Query
->|Image Recognition Service|:Identify Objects in Image;
Generate Response Based on Image Analysis
Check if Additional Information is Needed
Need More Details?
yes
no
Request Additional Input
Prepare Final Response
|User|:Provide Additional Details
->|Chatbot|:Process Additional Details;
Generate Refined Response
Receive Response
Is Feedback Provided?
yes
->|Chatbot|:Receive Feedback;
Log Feedback for Improvement

EX NO:6	DRAW ACTIVITY DIAGRAM OF ALL USE CASES.
DATE:	

AIM:

To Draw the activity Diagram for any project

ALGORITHM:

Step 1: Identify the Initial State and Final States

Step 2: Identify the Intermediate Activities Needed

Step 3: Identify the Conditions or Constraints

Step 4: Draw the Diagram with Appropriate Notations

INPUTS:

Activities

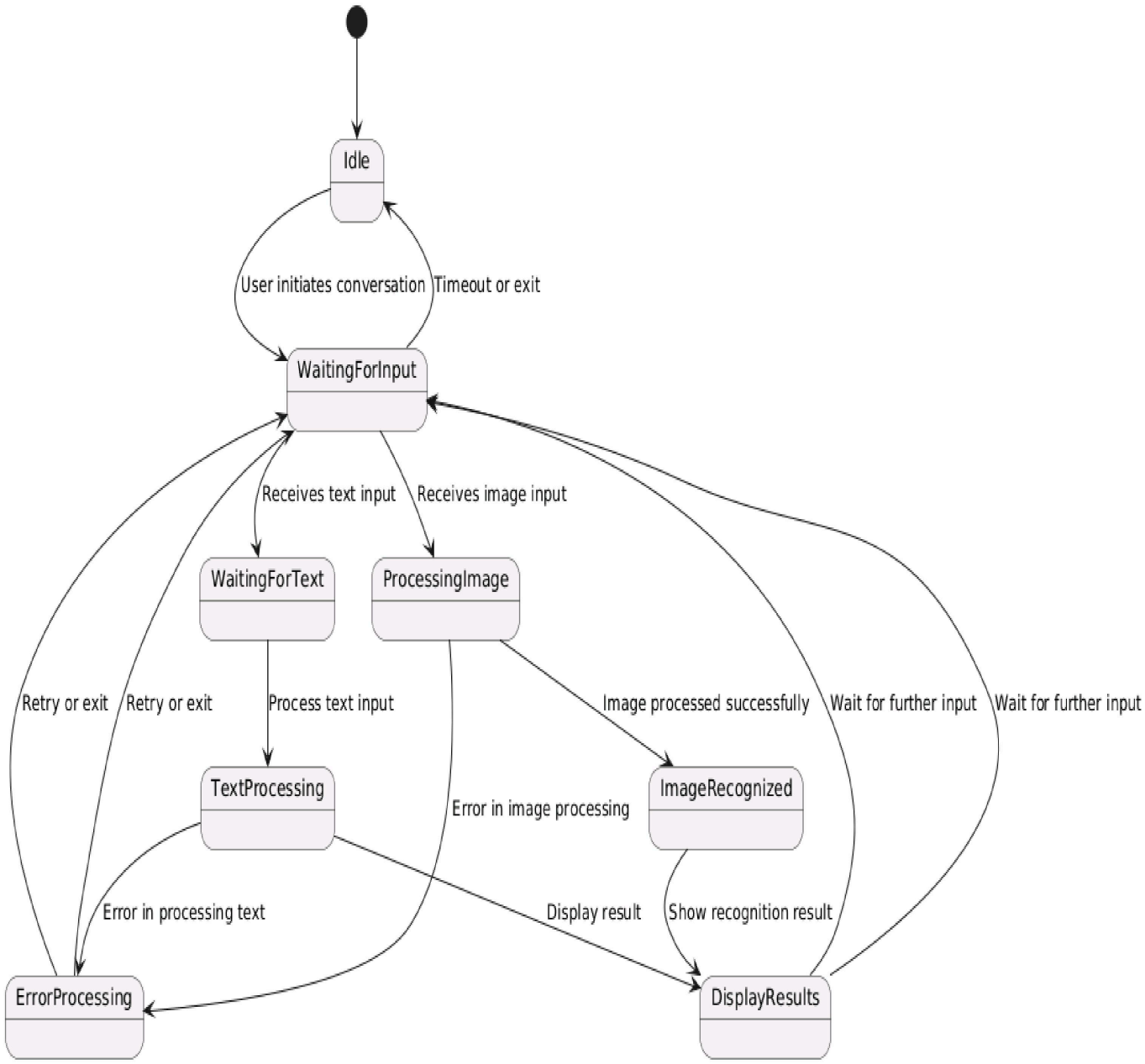
Decision Points

Guards

Parallel Activities

Conditions

Result:


Idle
User initiates conversation
Timeout or exit
WaitingForInput
Receives text input
Receives image input
WaitingForText
ProcessingImage
Retry or exit
Retry or exit
Process text input
Image processed successfully
Wait for further input
Wait for further input
TextProcessing
Error in image processing
ImageRecognized
Error in processing text
Display result
Show recognition result
ErrorProcessing
DisplayResults

EX NO:7	DRAW STATE CHART DIAGRAM OF ALL USE CASES.
DATE:	

AIM:

To Draw the State Chart Diagram for any project

ALGORITHM:

STEP-1: Identify the important objects to be analysed.

STEP-2: Identify the states.

STEP-3: Identify the events.

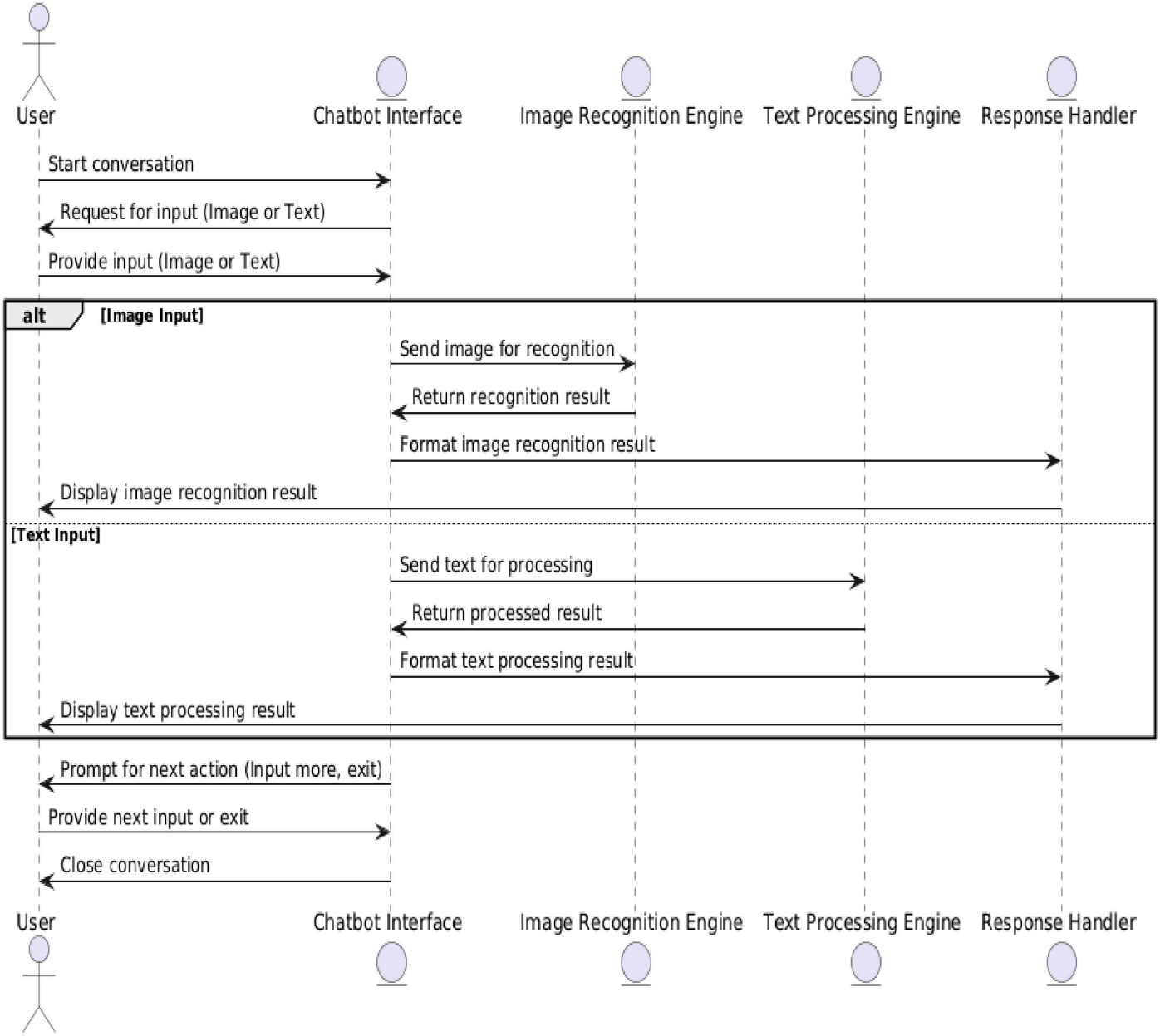
INPUTS:

Objects

States

Events

Result:


User
Chatbot Interface
Image Recognition Engine
Text Processing Engine
Response Handler
Start conversation
Request for input (Image or Text)
Provide input (Image or Text)
alt
[Image Input]
Send image for recognition
Return recognition result
Format image recognition result
Display image recognition result
[Text Input]
Send text for processing
Return processed result
Format text processing result
Display text processing result
Prompt for next action (Input more, exit)
Provide next input or exit
Close conversation
User
Chatbot Interface
Image Recognition Engine
Text Processing Engine
Response Handler

EX NO:8	DRAW SEQUENCE DIAGRAM OF ALL USE CASES.
DATE:	

AIM: To Draw the Sequence Diagram for any project

ALGORITHM:

1. Identify the Scenario
2. List the Participants
3. Define Lifelines
4. Arrange Lifelines
5. Add Activation Bars
6. Draw Messages
7. Include Return Messages
8. Indicate Timing and Order
9. Include Conditions and Loops
10. Consider Parallel Execution
11. Review and Refine
12. Add Annotations and Comments
13. Document Assumptions and Constraints
14. Use a Tool to create a neat sequence diagram

INPUTS:

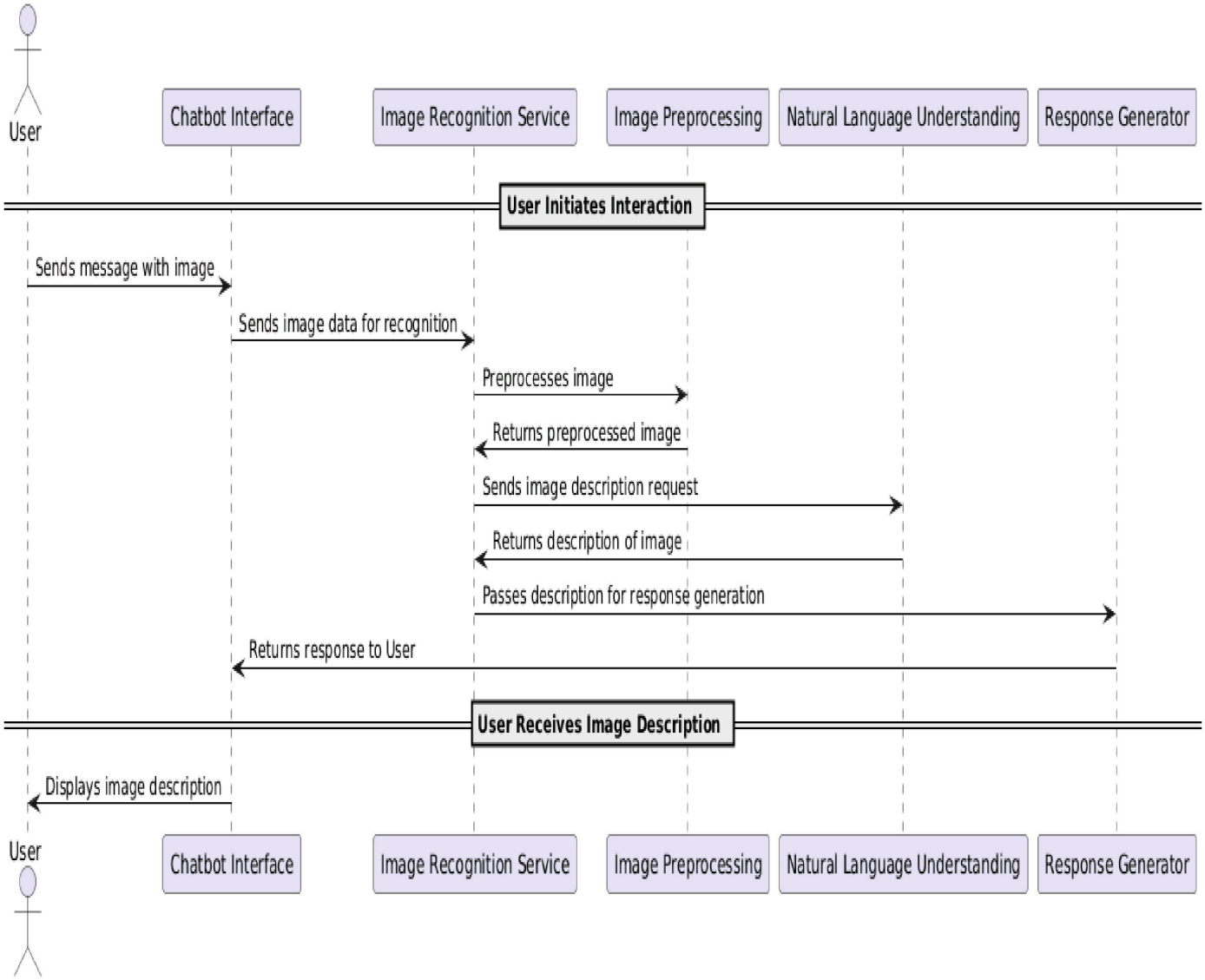
Objects taking part in the interaction.

Message flows among the objects.

The sequence in which the messages are flowing.

Object organization.

Result:


User
Chatbot Interface
Image Recognition Service
Image Preprocessing
Natural Language Understanding
Response Generator
User Initiates Interaction
Sends message with image
Sends image data for recognition
Preprocesses image
Returns preprocessed image
Sends image description request
Returns description of image
Passes description for response generation
Returns response to User
User Receives Image Description
Displays image description
User
Chatbot Interface
Image Recognition Service
Image Preprocessing
Natural Language Understanding
Response Generator

EX NO:9	DRAW COLLABORATION DIAGRAM OF ALL USE CASES
DATE:	

AIM:

To Draw the Collaboration Diagram for any project

ALGORITHM:

Step 1: Identify Objects/Participants

Step 2: Define Interactions

Step 3: Add Messages

Step 4: Consider Relationships

Step 5: Document the collaboration diagram along with any relevant explanations or annotations.

INPUTS:

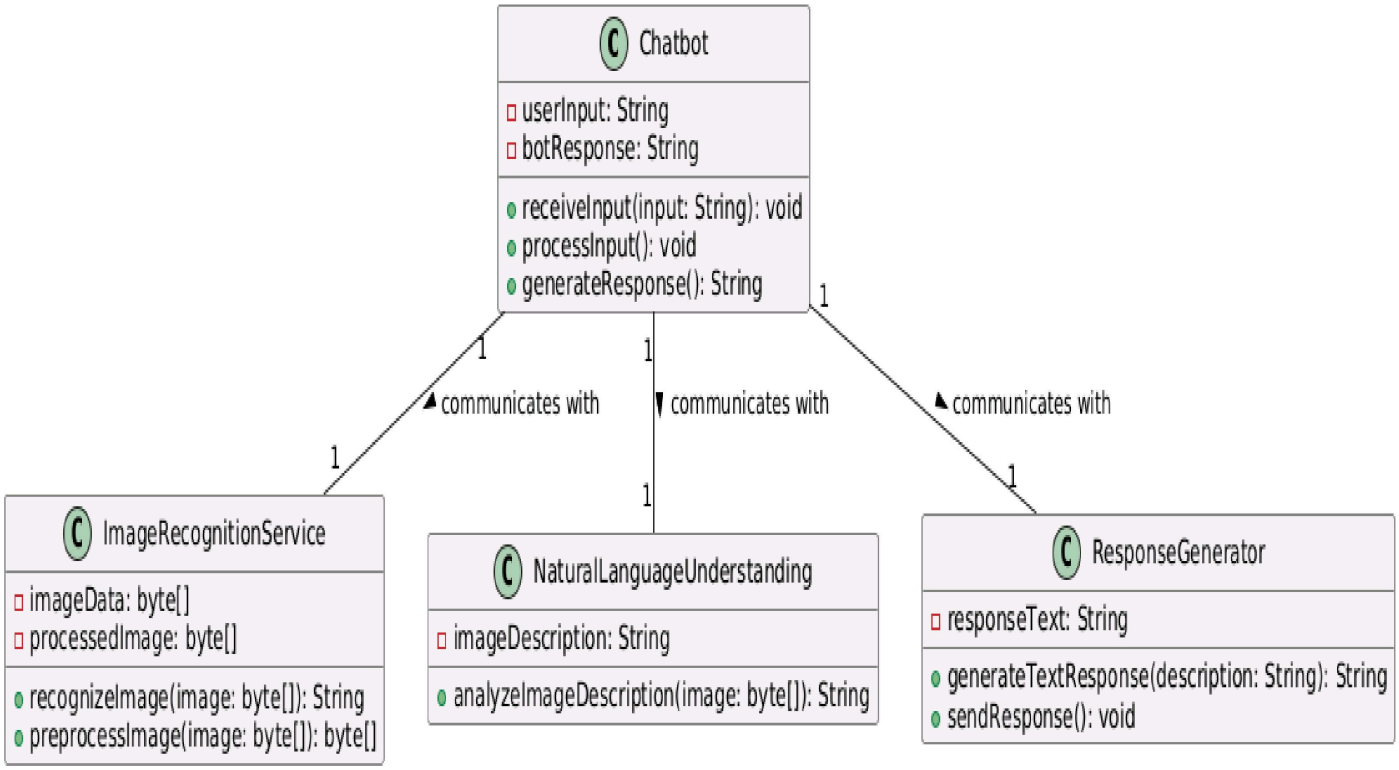
Objects taking part in the interaction.

Message flows among the objects.

The sequence in which the messages are flowing.

Object organization.

Result:


Chatbot
userInput: String
botResponse: String
receiveInput(input: String): void
processInput(): void
generateResponse(): String
1
1
communicates with
1
1
communicates with
1
1
communicates with
1
ImageRecognitionService
imageData: byte[]
processedImage: byte[]
recognizeImage(image: byte[]): String
preprocessImage(image: byte[]): byte[]
NaturalLanguageUnderstanding
imageDescription: String
analyzeImageDescription(image: byte[]): String
ResponseGenerator
responseText: String
generateTextResponse(description: String): String
sendResponse(): void

EX NO:10	ASSIGN OBJECTS IN SEQUENCE DIAGRAM TO CLASSES AND MAKE CLASS DIAGRAM.
DATE:	

AIM:

To Draw the Class Diagram for any project

ALGORITHM:

1. Identify Classes
2. List Attributes and Methods
3. Identify Relationships
4. Create Class Boxes
5. Add Attributes and Methods
6. Draw Relationships
7. Label Relationships
8. Review and Refine
9. Use Tools for Digital Drawing

INPUTS:

1. Class Name
2. Attributes
3. Methods
4. Visibility Notation

RESULT:

EX NO:11	MINI PROJECT-CONVERSATIONAL IMAGE RECOGNITION CHATBOT
DATE:	

AIM:

To develop a conversation image recognition chatbot using Streamlit and MySQL that allows staff members to efficiently manage parking slots, register vehicles, track parking activity, and notify vehicle owners via email. The focus is to streamline parking operations, enhance user convenience, and ensure reliability and simplicity in managing parking facilities.

ALGORITHM:

1. Database Connection Initialization
2. Streamlit Interface Setup
3. Operation Selection
- Process image
- Query logs
4. Database Query Execution
5. Email Notification (for Optional)
6. Feedback Display
7. Application Termination

Conversational Image Recognition Chatbot

Upload an image, ask a question, and get an intelligent response!

Upload an Image

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Browse files

No image uploaded.

Ask something about the image or start a conversation:

Send

Project: Conversational Image Recognition Chatbot
Author: Your Name

PROGRAM:

```
import streamlit as st
import tensorflow as tf
from tensorflow.keras.applications import mobilenet_v2
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input, decode_predictions
from PIL import Image
import numpy as np
import openai

# Set OpenAI API Key
openai.api_key = "your_openai_api_key"

# Load pre-trained MobileNetV2 model
model = mobilenet_v2.MobileNetV2(weights="imagenet")

# Function to analyze the image
def recognize_image(image):
    try:
        # Resize and preprocess the image
        img = image.resize((224, 224))
        img_array = np.array(img)
        img_array = preprocess_input(np.expand_dims(img_array, axis=0))

        # Predict using MobileNetV2
        predictions = model.predict(img_array)
        decoded_predictions = decode_predictions(predictions, top=3)[0]

        # Format the results
        results = [{"object": obj[1], "confidence": float(obj[2])} for obj in decoded_predictions]
        return results
    except Exception as e:
        return [{"error": str(e)}]

# Function to get GPT-generated response
```

```
def generate_response(user_input, image_info=None):
    try:
        prompt = (
            f"You are a helpful AI assistant. {f'The image analysis shows: {image_info}.' if image_info else ''} "
            f"User said: {user_input}. Respond conversationally."
        )
        response = openai.Completion.create(
            engine="text-davinci-003",
            prompt=prompt,
            max_tokens=100,
        )
        return response.choices[0].text.strip()
    except Exception as e:
        return str(e)

# Streamlit App
st.title("Conversational Image Recognition Chatbot")
st.write("Upload an image, ask a question, and get an intelligent response!")

# Image upload
uploaded_file = st.file_uploader("Upload an Image", type=["jpg", "png", "jpeg"])

if uploaded_file:
    # Display the uploaded image
    st.image(uploaded_file, caption="Uploaded Image", use_column_width=True)
    image = Image.open(uploaded_file)

    # Process the image
    st.write("Analyzing the image...")
    image_info = recognize_image(image)
    st.write("Image Analysis Results:", image_info)
else:
    image_info = None
```

```
    st.write("No image uploaded.")


# Chat input
user_input = st.text_input("Ask something about the image or start a conversation:")


if st.button("Send"):
    if user_input:
        # Generate a response
        response = generate_response(user_input, image_info)
        st.write("Chatbot Response:")
        st.write(response)
    else:
        st.write("Please enter a message to chat!")


# Footer
st.markdown(
    """
    ---
    *Project:* Conversational Image Recognition Chatbot
    *Author:* Your Name
    """
)
```

Conclusion:

The Conversational Image recognition chatbot developed using Streamlit and SQLite offers a user-friendly interface for students, teachers, and administrators to efficiently manage academic performance data. This system centralizes key functionalities such as adding, updating, and viewing student marks, tracking performance trends, and generating notifications for parents about student progress.