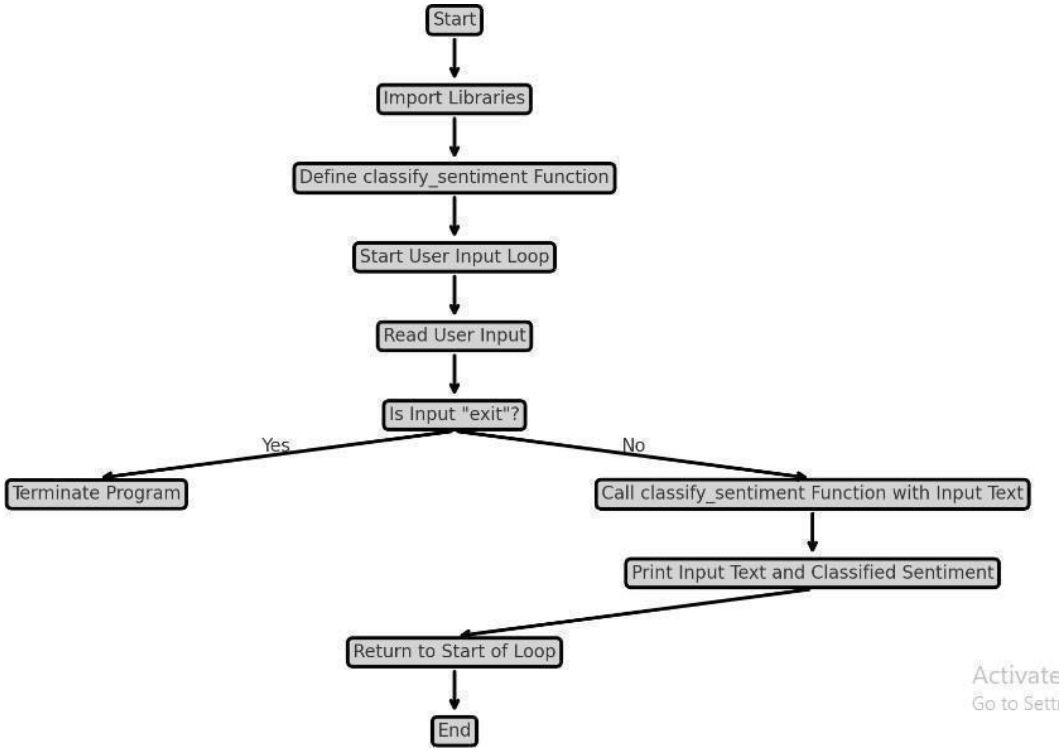
ALGORITHM

1. **Import Necessary Libraries:**
   - Import the TextBlob class from the TextBlob library.
2. **Define the Sentiment Classification Function:**
   - Create a function classify_sentiment (text) that:
     - Takes a string input (text).
     - Creates a TextBlob object from the input text.
     - Retrieves the polarity score from the TextBlob object.
     - Classifies the sentiment based on the polarity score:
       - Returns "positive" if the polarity is greater than 0.
       - Returns "negative" if the polarity is less than 0.
       - Returns "neutral" if the polarity is equal to 0.
3. **Read and Process User Input:**
   - Implement a loop to continuously read input from the user.
   - If the user inputs "exit", break the loop and terminate the program.
   - For each input text, classify the sentiment using the classify_sentiment function.
   - Print the input text and its classified sentiment.

FLOWCHART:



SOURCE CODE

BUILDING SENTIMENT ANALYSIS USING PYTHON PROGRAMMING

This project utilizes pre-trained sentiment analysis models to classify text as positive, negative, or neutral. The sentiment analysis is performed using the TextBlob library, which leverages Natural Language Processing (NLP) techniques to determine the sentiment polarity of the input text.

PROGRAM:

```
from textblob import TextBlob

def classify_sentiment(text):

    # Create a TextBlob object

    blob = TextBlob(text)

    # Get the polarity score

    polarity = blob.sentiment.polarity

    # Classify sentiment based on polarity

    if polarity > 0:

        return "positive"

    elif polarity < 0:

        return "negative"

    else:

        return "neutral"

# Read the input dynamically from the user

while True:

    text = input("Enter text for sentiment analysis (or type 'exit' to quit): ")
```

```
if text.lower() == 'exit'
    break
sentiment = classify_sentiment(text)
print(f"Text:{text}")
print(f"Sentiment: {sentiment}\n")
```

CODE EXPLANATION:

- Define a list of text samples for sentiment analysis. These samples represent a variety of sentiments, including positive, negative, and neutral.
- Implement a function, classify_sentiment, that takes a text input and uses TextBlob to analyze its sentiment.
- The function creates a Textblob object for the given text and retrieves its polarity score, which ranges from -1.0 (very negative) to 1.0 (very positive).
- Based on the polarity score, the function classifies the text as "positive" (polarity > 0), "negative" (polarity < 0), or "neutral" (polarity == 0).
- Perform sentiment analysis on each text sample using the classify_sentiment function.
- Print the original text and its corresponding sentiment classification.

OUTPUT:

Text: I regret buying that expensive gadget.
Sentiment: negative
Text: I love spending time with my family and friends.
Sentiment: positive
Text: The book is on the table.
Sentiment: neutral
Text: The weather is amazing today!
Sentiment: positive
Text: He is wearing a blue shirt.
Sentiment: neutral