A Project Report On

MAD LIBS GENERATOR

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

By

Ms . KARRI JYOTHSNA CHANDANA

Regd. Number: 21B91A04A2

IIDT BLACKBUCKS ChatGPT SHORT TERM INTERNSHIP

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

S. R. K. R. ENGINEERING COLLEGE (A)

(Affiliated to JNTU, KAKINADA) BHIMAVARAM-534204 (2024-2025)

S. R. K. R. ENGINEERING COLLEGE BHIMAVARAM

CERTIFICATE

This is to certify that this is a bonafide work on “MAD LIBS GENERATOR” which has been developed and submitted by Ms. Karri Jyothsna Chandana (21B91A04A2) at the end of third year second semester at IIDT Blackbucks in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering.

TABLE OF CONTENTS

ABSTRACT

Mad Libs is an entertaining word game that involves replacing specific parts of a story or sentence with blank spaces and prompting players to fill in the blanks with various types of words. The goal of this project is to create a userfriendly and interactive game that utilizes **deep learning** concepts that involves generating text with **neural networks**. This can be implemented using the **"transformers"** library with a GUI built using **Tkinter** .
The core functionality of the Mad Libs generator includes:
• Story Templates: Stores story templates with blanks designated by parts of speech (e.g., noun, verb, adjective).
• User Interaction: Prompts users to provide words based on specified parts of speech.
• Story Generation: Integrates user-provided words into the story template to create a unique and often humorous story.
Benefits:
• Entertainment: Provides a fun, interactive way to play with language, fostering laughter and enjoyment.
• Education: Helps users learn about parts of speech and sentence structure in a light-hearted manner.
• Creativity: Encourages creative thinking by prompting users to come up with unexpected words for the blanks.
Sample input and output:
Story Template: Once upon a time, there was a [noun] who was very [adjective]. Every day, it would [verb] to the [place] where it would see a [animal]. They became best friends and had many adventures together.
User Interaction:
1. Enter a noun: dragon
2. Enter an adjective: brave
3. Enter a verb: fly
4. Enter a place: forest
5. Enter an animal: unicorn
Generated Story:

Once upon a time, there was a dragon who was very brave. Every day, it would fly to the forest where it would see a unicorn. They became best friends and had many adventures together

SOFTWARE REQUIREMENTS

1. Text Editor: Jupyter notebook
2. Internet Browser: Google Chrome/Mozilla Firefox/Internet Browser
3. Libraries : Tkinter
4. Technology used : Deep learning

TECHNICAL STACK

- **Libraries used :**

  - **Tkinter (Tk interface) :**

    Tkinter is the standard graphical user interface (GUI) library for Python, providing a fast and easy way to create simple and interactive applications. Derived from the Tk GUI toolkit, Tkinter offers a variety of widgets such as buttons, labels, text boxes, and more, which can be used to design functional and visually appealing interfaces. It is included with the standard Python distribution, making it widely accessible and easy to use without the need for additional installations. Tkinter's simplicity and flexibility make it an excellent choice for beginners and those looking to quickly prototype applications. It uses a combination of event-driven programming and object-oriented concepts, allowing developers to respond to user actions like clicks and key presses with ease. The library supports different layout managers like `pack`, `grid`, and `place`, which help in organizing widgets within the application window. Despite its simplicity, Tkinter is powerful enough to handle more complex GUI tasks, and its cross-platform nature ensures that applications run consistently on Windows, macOS, and Linux. For more advanced needs, Tkinter can be extended with additional libraries and modules, offering a comprehensive solution for Python-based GUI development.

- **Technology used :**

  - **Deep Learning :**

    Deep learning is a subset of machine learning focused on neural networks with many layers, known as deep neural networks. These networks are inspired by the human brain and are designed to recognize patterns and make decisions with minimal human intervention. A deep neural network consists of an input layer, multiple hidden layers, and an output layer. Each neuron in a layer is connected to neurons in the subsequent layer, and these connections have weights that are adjusted during training to minimize error. The training process involves using large datasets and significant computational power, often leveraging GPUs for efficient processing. Key techniques in deep learning include convolutional neural networks (CNNs) for image recognition, recurrent neural networks (RNNs) for sequential data like text and speech, and generative adversarial networks (GANs) for creating new data samples.

ARCHITECTURE

```
A[Start] --> B{Choose a story template}
B --> C{Display story with blanks (nouns, verbs, etc.)}
C --> D{User enters word for first blank}
D --> E{Is this the last blank?}
    E(Yes) --> F{Insert all user words into story}
    E(No) --> C
F --> G{Display completed Mad Lib story}
G --> H{Play again?}
    H(Yes) --> B
    H(No) --> I[End]
```

This flowchart outlines the core gameplay loop:

1. **Start:** The program begins.
2. **Choose a story template:** The user selects a story template from a collection offered by the program.
3. **Display story with blanks:** The program displays the chosen story template, highlighting the blanks with corresponding parts of speech (noun, verb, adjective, etc.).
4. **User enters word for first blank:** The user provides a word that fits the grammatical category of the first blank.
5. **Is this the last blank?:** The program checks if all the blanks in the story have been filled.
    - **Yes:** If it's the last blank, proceed to step 6.
    - **No:** If there are more blanks, go back to step 3 to display the updated story with the filled blank and prompt the user for the next word.
6. **Insert all user words into story:** The program integrates all the user-provided words into the story template, creating the final Mad Lib story.
7. **Display completed Mad Lib story:** The program presents the completed Mad Lib story for the user's amusement.
8. **Play again?:** The program asks the user if they'd like to play another round.
    - **Yes:** If the user wants to play again, go back to step 2 to choose a new story template.
    - **No:** If the user doesn't want to play again, the program ends.
9. **End:** The program terminates.

DESCRIPTION

Data Collection and Preprocessing

- **Collect Mad Libs Stories:** Gather a dataset of various Mad Libs stories with their corresponding parts of speech.
- **Tokenization:** Break down the text into individual words or tokens.
- **Part-of-Speech Tagging:** Assign a part-of-speech tag (noun, verb, adjective, etc.) to each token.
- **Data Cleaning:** Remove noise, inconsistencies, and irrelevant data.

Model Training

- **Choose Deep Learning Model:** Select an appropriate model (e.g., Recurrent Neural Network, Transformer) based on the complexity of the task.
- **Model Architecture:** Define the architecture of the chosen model, including layers, neurons, and activation functions.
- **Training Process:** Feed the preprocessed data into the model and train it to predict the next word or part of speech based on the previous context.
- **Model Evaluation:** Assess the model's performance using metrics like accuracy, perplexity, or BLEU score.

Mad Libs Generation

- **User Input:** Prompt the user to provide words based on the generated prompts.
- **Story Generation:** Use the trained model to generate a Mad Libs story based on the user's input.
- **Part-of-Speech Replacement:** Replace the predicted parts of speech with the user's input words.
- **Story Output:** Display the completed Mad Libs story to the user.

Additional Considerations

- **User Interface:** Design an intuitive user interface for input and output.
- **Story Variety:** Implement mechanisms to generate different types of Mad Libs stories (e.g., humorous, adventurous).
- **Difficulty Levels:** Offer options for different difficulty levels based on the complexity of the generated stories.
- **Interactive Features:** Incorporate interactive elements like multiple choice or fill-in-the-blank prompts.

CODE

Step 1: Install Tkinter

Source Code Snippet:

```
pip install tkinter
```

The GUI toolkit created for Python is called Tkinter. It is the quickest and simplest method for creating apps with graphical user interfaces.

Step 2: Initialize the window and create buttons

Source Code Snippet:

```
Screen11 = Tk()
Screen11.title(" Mad Libs Generator")
Screen11.geometry('400x400')
Screen11.config(bg="pink")
Label(Screen11, text=' Mad Libs Generator').place(x=100, y=20)
Story111Button = Button(Screen11, text='A memorable day', font=("Calibri New Roman", 13),command=lambda: Story11(Screen11),bg='Brown')
Story111Button.place(x=140, y=90)
Story222Button = Button(Screen11, text='Career', font=("Calibri New Roman", 13),command=lambda: Story22(Screen11), bg='Brown')
Story222Button.place(x=150, y=150)
Screen11.update()
Screen11.mainloop()
```

Explanation of the above Code:

The two buttons, Story111 and Story222, link users to a pop-up window where narratives will be generated after filling in the blanks.

1. Tk() function: It aids the window's appearance on Screen11.
2. geometry() function: It specifies the Screen11's geometry.
3. title() function: The title is shown in parentheses at the top of the window.
4. config() function: This function changes the window's background colour.
5. The event loop can be run using the mainloop() function.
6. Button()function: It causes buttons to appear on the window.

Step 3: First story function creation

Source Code Snippet:

```
def Story11(win):
    def final(tl: Toplevel, user_name, sports, City, playeruser_name, drinkuser_name, snacks):
        text1 = f""" 
        We decided to play a {sports} game in {City} one day with our friend {user_name}.
But we were unable to play. Consequently, we attended the game to watch our favourite player, {playeruser_name}.
We consumed {drinkuser_name} and some {snacks} as well.
It was great fun for us! We are eager to visit once more and have fun.
"""
        tl.geometry(newGeometry='500x550')
        Label(tl, text='Story:',  wraplength=tl.winfo_width()).place(x=160, y=310)
        Label(tl, text=text1,wraplength=tl.winfo_width()).place(x=0, y=330)
    NewScreen11 = Toplevel(win, bg='yellow')
    NewScreen11.title("A Memorable Day")
    NewScreen11.geometry('500x500')
    Label(NewScreen11, text=' A Memorable Day').place(x=100, y=0)
    Label(NewScreen11, text='User_name:').place(x=0, y=35)
    Label(NewScreen11, text='Enter a game:').place(x=0, y=70)
    Label(NewScreen11, text='Enter a city:').place(x=0, y=110)
    Label(NewScreen11, text='Enter the user_name of a player:').place(x=0, y=150)
    Label(NewScreen11, text='Enter the user_name of a drink:').place(x=0, y=190)
    Label(NewScreen11, text='Enter the user_name of a snack:').place(x=0, y=230)
    User_name = Entry(NewScreen11, width=11)
    User_name.place(x=200, y=35)
    game = Entry(NewScreen11, width=11)
    game.place(x=200, y=70)
    city = Entry(NewScreen11, width=11)
    city.place(x=200, y=105)
    player = Entry(NewScreen11, width=11)
    player.place(x=200, y=150)
```

```
drink = Entry(NewScreen11, width=11)
drink.place(x=200, y=190)
snack = Entry(NewScreen11, width=11)
snack.place(x=200, y=230)
SubmitButton1 = Button(NewScreen11, text="Submit", background="Brown", font=('Calibri', 12), command=lambda:final(NewScreen11, User_name.get(), game.get(), city.get(), player.get(), drink.get(), snack.get()))
SubmitButton1.place(x=150, y=270)
NewScreen11.mainloop()
```

Explanation of the above Code:

Story11() function will accept the desired input provided by the consumer and present it on Screen11. The story is in text 1.

1. label() function: This prints text1 in square brackets on the window.
2. entry() function: Text area is provided by entry() function.
3. get() function: It aids in obtaining the variable's value.
4. place () function: This places a text1 and textarea in a specific location on Screen11.

Step 4: Second story function Creation

Source Code Snippet:

```
def Story22(win):
 def final(tl: Toplevel, path, pronoun, emotional, emotion, adverb):
        text1 = f"""
 When I was younger, I wanted to be a {path}, but as I got older, I became interested in the {pronoun} and decided to become an engineer. Then I started working at a job where I was not {emotional}.
 I decided to pursue my passion after experiencing {emotion}
 Despite receiving less {adverb} than I did in my prior employment.
 I'm very sensitive. '"
 #let us set the geometry of the project
 tl.geometry(newGeometry='500x550')
 Label(tl, text='Story:',  wraplength=tl.winfo_width()).place(x=160, y=310)
 Label(tl, text=text1,wraplength=tl.winfo_width()).place(x=0, y=330)
 NewScreen11 = Toplevel(win, bg='red')
NewScreen11.title("Career")
NewScreen11.geometry('500x500')
```

```
Label(NewScreen11, text='Career').place(x=150, y=0)
Label(NewScreen11, text='Your Childhood dream:').place(x=0, y=35)
Label(NewScreen11, text='Your current interest:').place(x=0, y=70)
Label(NewScreen11, text='Enter a emotion:').place(x=0, y=110)
Label(NewScreen11, text='Enter a negative emotion:').place(x=0, y=150)
Label(NewScreen11, text='Enter a adverb:').place(x=0, y=190)
Path = Entry(NewScreen11, width=15)
Path.place(x=200, y=35)
Pronoun = Entry(NewScreen11, width=15)
Pronoun.place(x=200, y=70)
Emotional = Entry(NewScreen11, width=15)
Emotional.place(x=200, y=110)
Emotion= Entry(NewScreen11, width=15)
Emotion.place(x=200, y=150)
Adverb = Entry(NewScreen11, width=15)
Adverb.place(x=200, y=190)
SubmitButton1 = Button(NewScreen11, text="Submit", background="Brown", font=('Calibri', 12), command=lambda:final(NewScreen11, Path.get(), Pronoun.get(), Emotional.get(), Emotion.get(), Adverb.get()))
SubmitButton1.place(x=150, y=270)
```

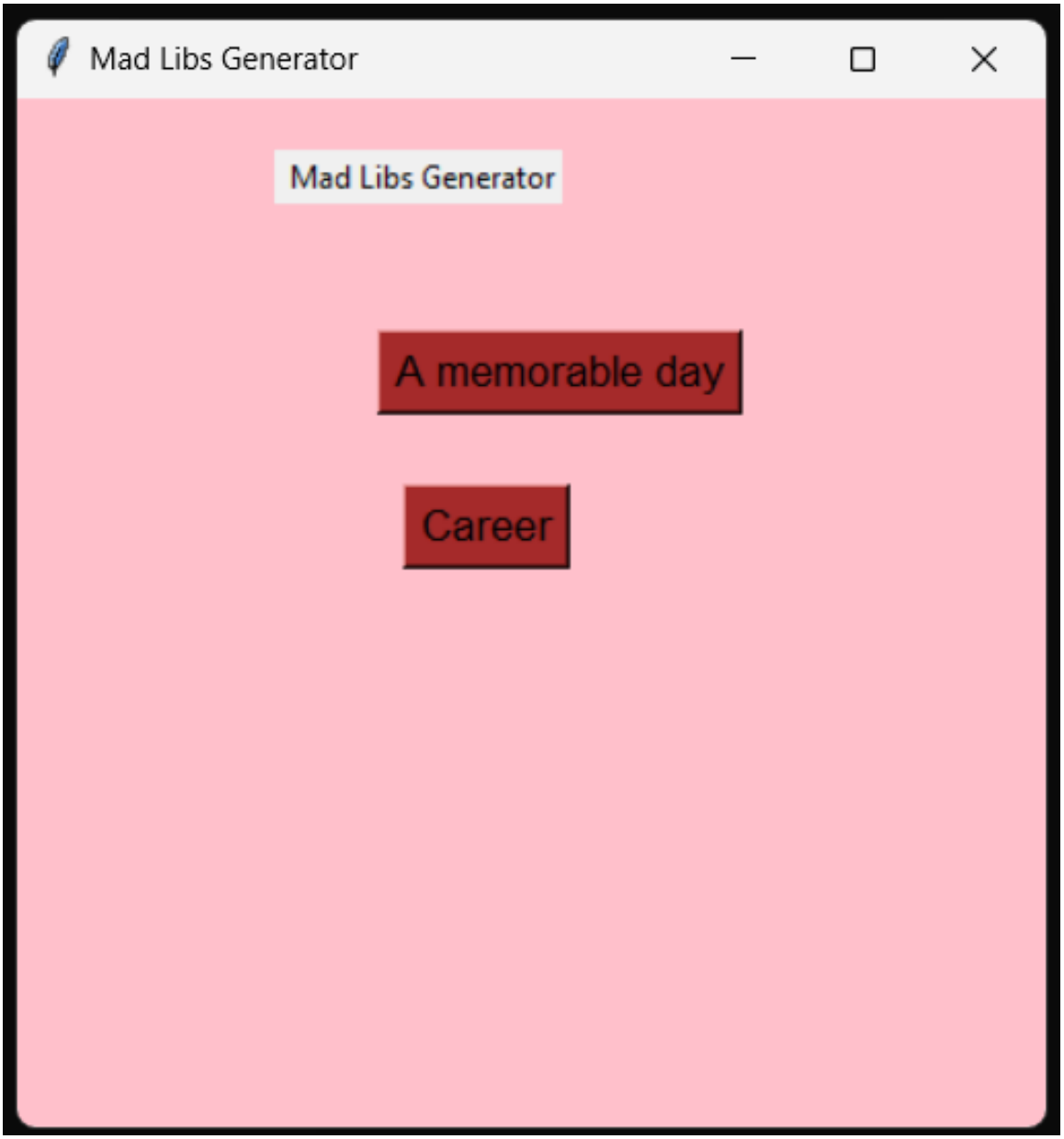
CONSOLIDATED CODE

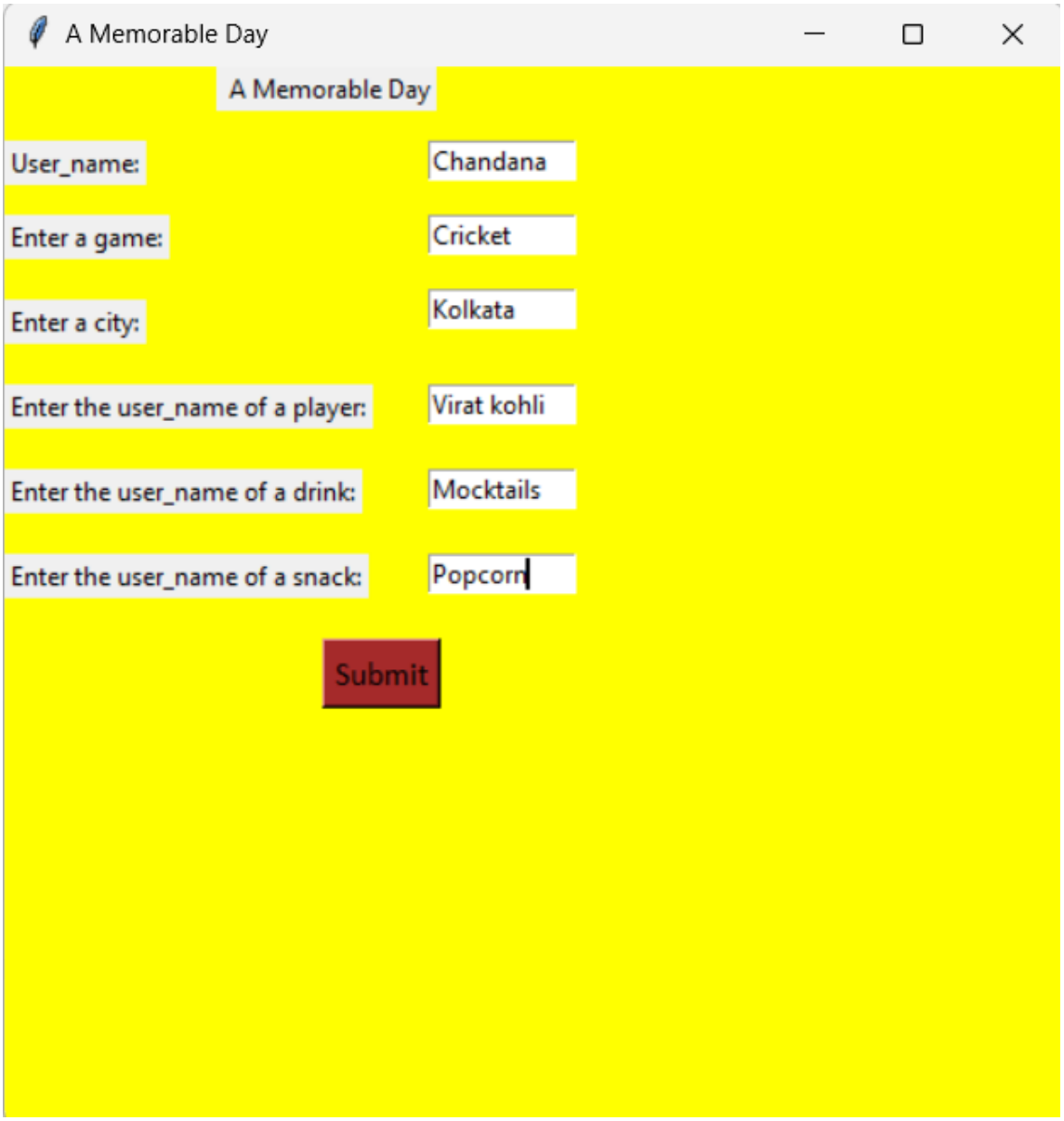
```
1. from tkinter import *
2. def Story11(win):
3.   def final(tl: Toplevel, user_name, sports, City, playeruser_name, drinkuser_name, snacks):

4.     text1 = f""" 
5.         We decided to play a "sports" game in "City" one day with our friend "user_name."
6. But we were unable to play. Consequently, we attended the game to watch our favourite pla
   yer, playeruser_name.
7. We consumed "drinkuser_name" and some "snacks" as well.
8. It was great fun for us! We are eager to visit once more and have fun.
9. """
10.    tl.geometry(newGeometry='500x550')
11.    Label(tl, text='Story:',  wraplength=tl.winfo_width()).place(x=160, y=310)
12.    Label(tl, text=text,wraplength=tl.winfo_width()).place(x=0, y=330)
13. NewScreen11 = Toplevel(win, bg='yellow')
14. NewScreen11.title("A Memorable Day")
15. NewScreen11.geometry('500x500')
16. Label(NewScreen11, text=' A Memorable Day').place(x=100, y=0)
17. Label(NewScreen11, text='User_name:').place(x=0, y=35)
18. Label(NewScreen11, text='Enter a game:').place(x=0, y=70)
19. Label(NewScreen11, text='Enter a city:').place(x=0, y=110)
20. Label(NewScreen11, text='Enter the user_name of a player:').place(x=0, y=150)
21. Label(NewScreen11, text='Enter the user_name of a drink:').place(x=0, y=110)
22. Label(NewScreen11, text='Enter the user_name of a snack:').place(x=0, y=230)
23. User_name = Entry(NewScreen11, width=11)
24. User_name.place(x=100, y=35)
25. game = Entry(NewScreen11, width=11)
26. game.place(x=100, y=70)
27. city = Entry(NewScreen11, width=11)
28. city.place(x=100, y=105)
29. player = Entry(NewScreen11, width=11)
30. player.place(x=100, y=150)
31. drink = Entry(NewScreen11, width=11)
```

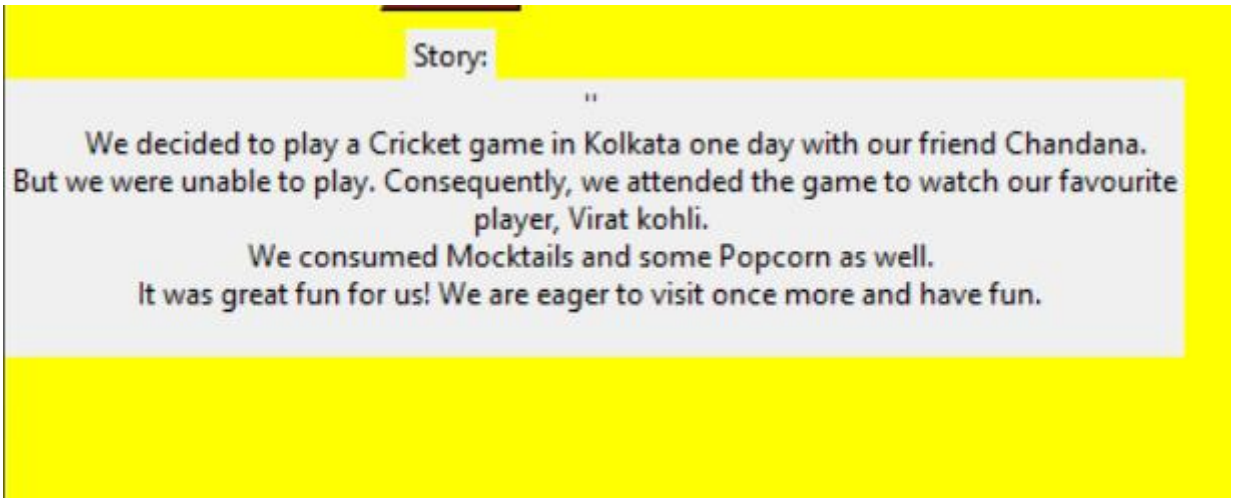
```
32.  drink.place(x=100, y=110)
33.  snack = Entry(NewScreen11, width=11)
34.  snack.place(x=100, y=120)
35.  SubmitButton1  = Button(NewScreen11, text="Submit", background="Brown", font=('Cal
     ibri', 12), command=lambda:final(NewScreen11, User_name.get(), game.get(), city.get(), p
     layer.get(), drink.get(), snack.get()))
36.  SubmitButton.place(x=150, y=270)
37.  NewScreen11.mainloop()
38. def Story22(win):
39.     def final(tl: Toplevel, path, pronoun, emotion, emotion,adverb):
40.         text1 = f"""
41.         When I was younger, I wanted to be a {path}, but as I got older, I became interested
     in the {pronoun} and decided to become an engineer. Then I started working at a job wher
     e I was not {emotional}.
42. I decided to pursue my passion after experiencing {emotion}
43. Despite receiving less {adverb} than I did in my prior employment.
44. I'm very sensitive. '"
45. #let us set the geometry of the project
46.         tl.geometry(newGeometry='500x550')
47.         Label(tl, text='Story:',  wraplength=tl.winfo_width()).place(x=160, y=310)
48.         Label(tl, text=text,wraplength=tl.winfo_width()).place(x=0, y=330)
49.     NewScreen11 = Toplevel(win, bg='red')
50.     NewScreen11.title("Career")
51.     NewScreen11.geometry('500x500')
52.     Label(NewScreen11, text='Career').place(x=150, y=0)
53.     Label(NewScreen11, text='Enter a path:').place(x=0, y=35)
54.     Label(NewScreen11, text='Enter a pronoun:').place(x=0, y=70)
55.     Label(NewScreen11, text='Enter a emotion:').place(x=0, y=110)
56.     Label(NewScreen11, text='Enter a emotion:').place(x=0, y=150)
57.     Label(NewScreen11, text='Enter a adverb:').place(x=0, y=110)
58.     Path = Entry(NewScreen11, width=11)
59.     Path.place(x=100, y=35)
60.     Pronoun = Entry(NewScreen11, width=11)
61.     Pronoun.place(x=100, y=70)
62.     Emotion = Entry(NewScreen11, width=11)
63.     Emotion.place(x=100, y=105)
```

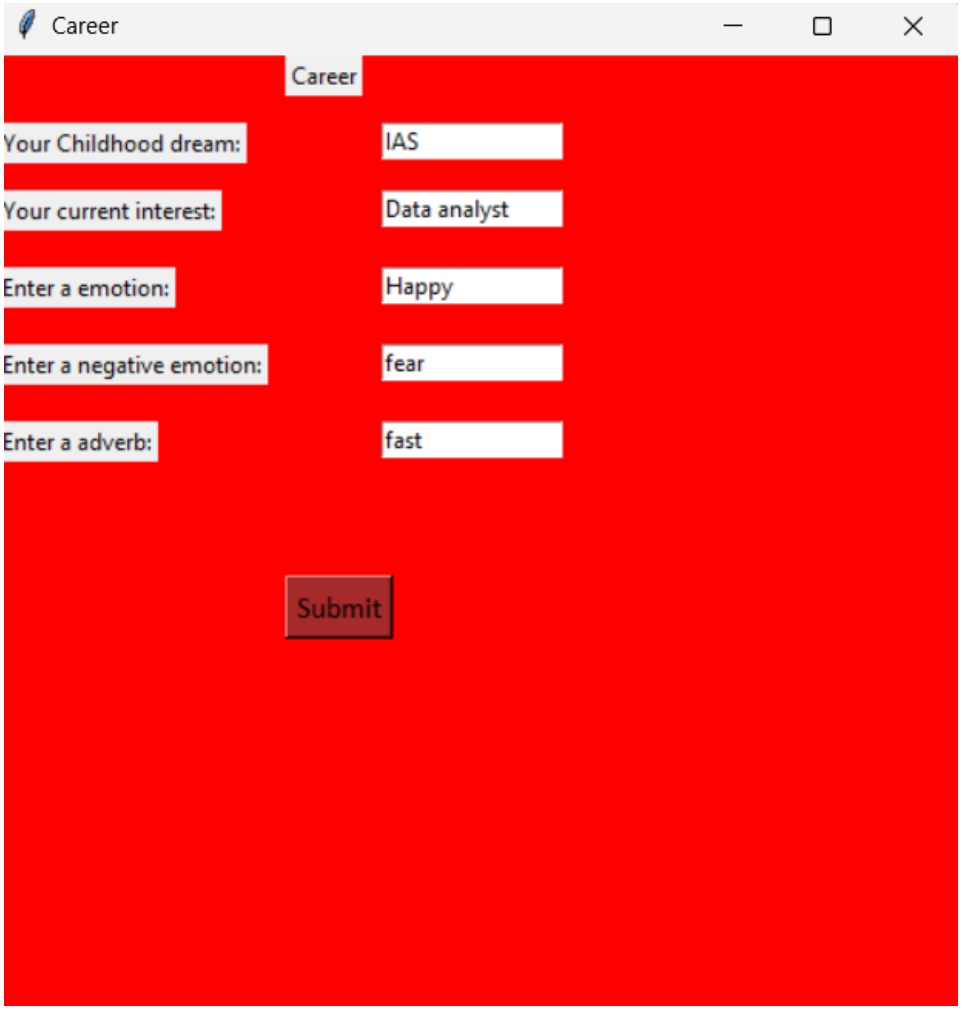
```
64.     Emotion= Entry(NewScreen11, width=11)
65.     Emotion.place(x=100, y=150)
66.     Adverb = Entry(NewScreen11, width=11)
67.     Adverb.place(x=100, y=110)
68.     SubmitButton1  = Button(NewScreen11, text="Submit", background="Brown", font=('C
        alibri', 12), command=lambda:final(NewScreen11, Path.get(), Pronoun.get(), Emotion.get(
        ), Emotion.get(), Adverb.get()))
69.     SubmitButton.place(x=150, y=270)
70. Screen11 = Tk()
71. Screen11.title(" Mad Libs Generator")
72. Screen11.geometry('400x400')
73. Screen11.config(bg="pink")
74. Label(Screen11, text=' Mad Libs Generator').place(x=100, y=20)
75. Story111Button = Button(Screen11, text='A memorable day', font=("Calibri New Roman",
        13),command=lambda: Story11(Screen11),bg='Brown')
76. Story111Button.place(x=140, y=90)
77. Story222Button = Button(Screen11, text='Career', font=("Calibri New Roman", 13),comma
        nd=lambda: Story22(Screen11), bg='Brown')
78. Story222Button.place(x=150, y=150)
79. Screen11.update()
80. Screen11.mainloop()
```

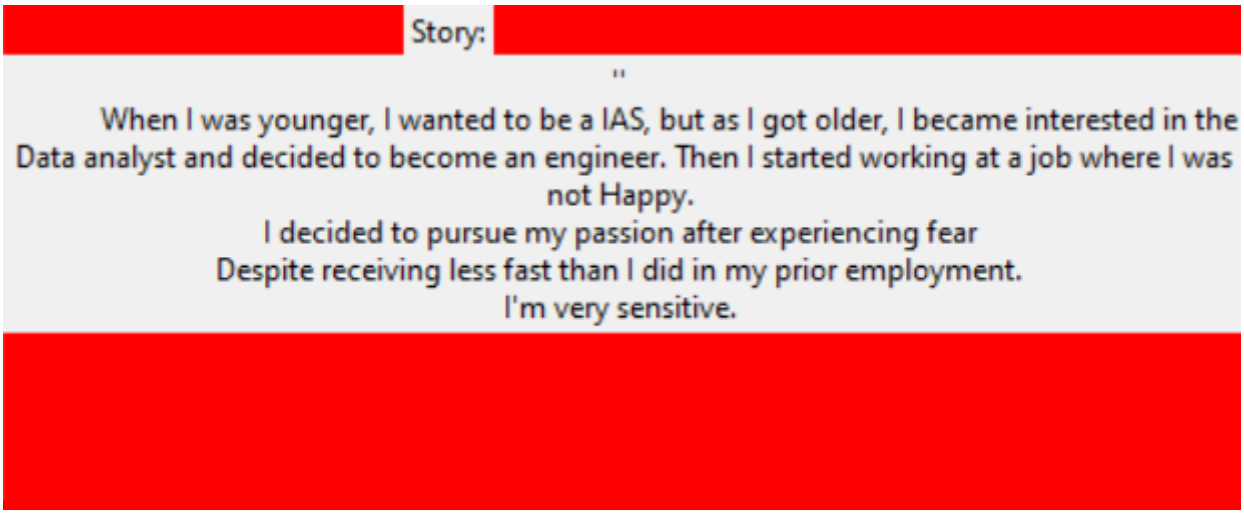
OUTPUT SCREENSHOTS




A Memorable Day
A Memorable Day
User_name:
Chandana
Enter a game:
Cricket
Enter a city:
Kolkata
Enter the user_name of a player:
Virat kohli
Enter the user_name of a drink:
Mocktails
Enter the user_name of a snack:
Popcorn
Submit


Story:
''
We decided to play a Cricket game in Kolkata one day with our friend Chandana.
But we were unable to play. Consequently, we attended the game to watch our favourite player, Virat kohli.
We consumed Mocktails and some Popcorn as well.
It was great fun for us! We are eager to visit once more and have fun.


Career
Career
Your Childhood dream:
IAS
Your current interest:
Data analyst
Enter a emotion:
Happy
Enter a negative emotion:
fear
Enter a adverb:
fast
Submit


Story:
''
When I was younger, I wanted to be a IAS, but as I got older, I became interested in the Data analyst and decided to become an engineer. Then I started working at a job where I was not Happy.
I decided to pursue my passion after experiencing fear
Despite receiving less fast than I did in my prior employment.
I'm very sensitive.