

|  |
| --- |
| **RAJALAKSHMI INSTITUTE OF TECHNOLOGY** |
| (An Autonomous Institution, Affiliated to Anna University, Chennai) |

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**ACADEMIC YEAR 2025 - 2026**

**SEMESTER III**

**ARTIFICIAL INTELLIGENCE LABORATORY**

**MINI PROJECT REPORT**

|  |  |
| --- | --- |
| **REGISTER NUMBER** | 2117240070025 |
| **NAME** | ARJUN V |
| **PROJECT TITLE** | TEXT BASED GAME USING TKINTER |
| **DATE OF SUBMISSION** | 29\10\2025 |
| **FACULTY IN-CHARGE** | **Mrs. M. Divya** |

**Signature of Faculty In-charge**

**INTRODUCTION**

A text-based game is an interactive story where the player makes choices to move through different story paths. Tkinter is Python’s built-in Graphical User Interface (GUI) toolkit used to develop desktop applications. In this project, Tkinter is used to create buttons and labels that allow users to interact with the game story. This project helps understand Python programming, GUI design, and basic game logic.

**PROBLEM STATEMENT**

To design and implement an interactive story-based adventure game using Python Tkinter where the player makes decisions using buttons and progresses through different story paths based on their choices.

**THEORETICAL BACKGROUND**

Tkinter is Python's standard GUI toolkit used to create graphical applications. In this game, the story progression is handled using a Finite State Machine (FSM) approach, where each story point is a state and user choices act as transitions to the next state. The story is stored using a nested dictionary structure in Python, where each state contains a text message and options for the next states.

**ALGORITHM EXPLANATION WITH EXAMPLE**

\* Start the game at the initial story state.

\* Display the story text and the available choices using buttons.

\* Wait for user input (button click).

\* Based on the input, transition to the next story state.

\* If a state has no options, display an ending message.

\* Provide a restart option to play again.

**IMPLEMENTATION AND CODE**

import tkinter as tk

# Define story states

story = {

"start": {

"text": "You are in a dark forest. Do you want to go left or right?",

"options": {"Left": "river", "Right": "mountain"}

},

"river": {

"text": "You reach a river. Do you want to swim or build a raft?",

"options": {"Swim": "shark", "Build Raft": "escape"}

},

"mountain": {

"text": "You climb a mountain. Do you want to camp or keep climbing?",

"options": {"Camp": "bear", "Keep Climbing": "escape"}

},

"shark": {

"text": "A shark attacks you! Game Over.",

"options": {}

},

"bear": {

"text": "A bear attacks you! Game Over.",

"options": {}

},

"escape": {

"text": "You escaped safely! You win!",

"options": {}

}

}

# Create main window

root = tk.Tk()

root.title("Adventure Game")

root.geometry("500x300")

root.config(bg="#222")

# Current state

current\_state = {"state": "start"}

# Story text label

story\_text = tk.Label(root, text="", wraplength=450, justify="center",

fg="white", bg="#222", font=("Arial", 14))

story\_text.pack(pady=40)

# Frame for buttons

button\_frame = tk.Frame(root, bg="#222")

button\_frame.pack()

# Function to display story

def display\_story(state\_key):

state = story[state\_key]

story\_text.config(text=state["text"])

# Clear previous buttons

for widget in button\_frame.winfo\_children():

widget.destroy()

# If end state (no options), show restart button

if not state["options"]:

end\_label = tk.Label(button\_frame, text="The End", fg="orange",

bg="#222", font=("Arial", 16, "bold"))

end\_label.pack(pady=10)

restart\_btn = tk.Button(button\_frame, text="Restart",

command=lambda: update\_story("start"),

bg="#444", fg="white", width=12, font=("Arial", 12))

restart\_btn.pack(pady=5)

return

# Create option buttons

for option\_text, next\_state in state["options"].items():

btn = tk.Button(button\_frame, text=option\_text, width=20, height=2,

bg="#5AC85A", fg="black", font=("Arial", 12, "bold"),

command=lambda ns=next\_state: update\_story(ns))

btn.pack(pady=5)

# Update story function

def update\_story(next\_state):

current\_state["state"] = next\_state

display\_story(next\_state)

# Start the game

display\_story("start")

root.mainloop()

**OUTPUT**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**RESULTS AND FUTURE ENHANCEMENT**

This project successfully demonstrates how to combine Python logic with GUI programming to build a simple game. It is easy to expand and improve. Future enhancements include:

* Adding background images and sounds.
* Adding more story paths and levels.
* Including a scoring system.
* Saving player progress.
* Adding animations and better UI design.

|  |  |
| --- | --- |
| **Git Hub Link of the project and report** | [**https://github.com/Arjun240025/AI\_MINI-PROJECT/blob/main/AI%20src%20code**](https://github.com/Arjun240025/AI_MINI-PROJECT/blob/main/AI%20src%20code) |

**REFERENCES**

* Python Tkinter Official Documentation – <https://docs.python.org>
* Automate the Boring Stuff with Python – Al Sweigart
* Real Python Tutorials [– https://realpython.com](–%20https:/realpython.com)
* GeeksforGeeks Python GUI [– https://geeksforgeeks.org](–%20https:/geeksforgeeks.org)
* W3Schools Python Tkinter – <https://w3schools.com>