NAME: ***SHIRISHA NAMBALA***

PROJECT TITLE: **ROCK PAPER SCISSOR**

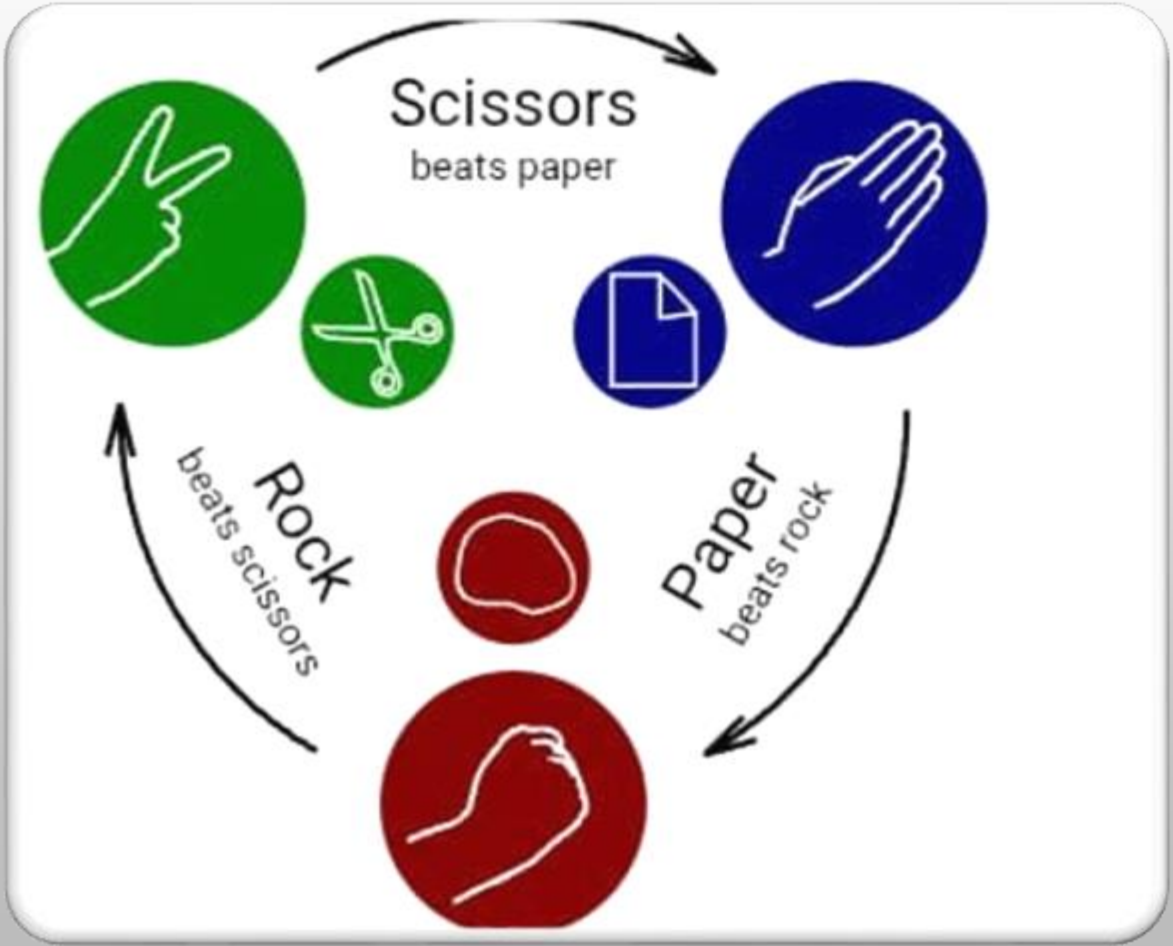
MICRO IT

INTRODUCTION:-

The rock–paper–scissors game is a simple python-based application that allows users to play the classic game against the computer. Using a graphical interface built with tkinter, players can choose rock, paper, or scissors, while the computer makes a random choice. The game then displays the result — win, lose, or draw — based on standard rules. This project is designed to demonstrate basic python programming concepts like GUI design, conditionals, and randomness.

OBJECTIVE:-

The objective of this project is to develop a user-friendly desktop application that simulates the classic rock–paper–scissors game. It aims to enhance user interaction through a graphical interface and reinforce fundamental programming concepts such as event handling, conditional logic, and random number generation. Additionally, the project serves as a practical exercise in GUI development using python's tkinter library, making it ideal for beginners to understand how logic and design come together in a simple game application.



FEATURES:-

- Interactive GUI to choose between rock, paper, and scissors
- Random choice generation for the computer
- Displays both player's and computer's choices
- Displays the result (win/lose/draw)
- Simple and user-friendly interface
- Restart/replay functionality (can be enhanced)

TOOLS AND TECHNOLOGIES:-

->TECHNOLOGIES USED:-

- Python 3
- Core programming language used for logic and game flow.
- Tkinter
- Built-in python GUI library for designing the user interface.
- Random module
- To generate the computer's move randomly (rock, paper, or scissors).

->TOOLS USED:-

- Visual studio code / pycharm / IDLE:

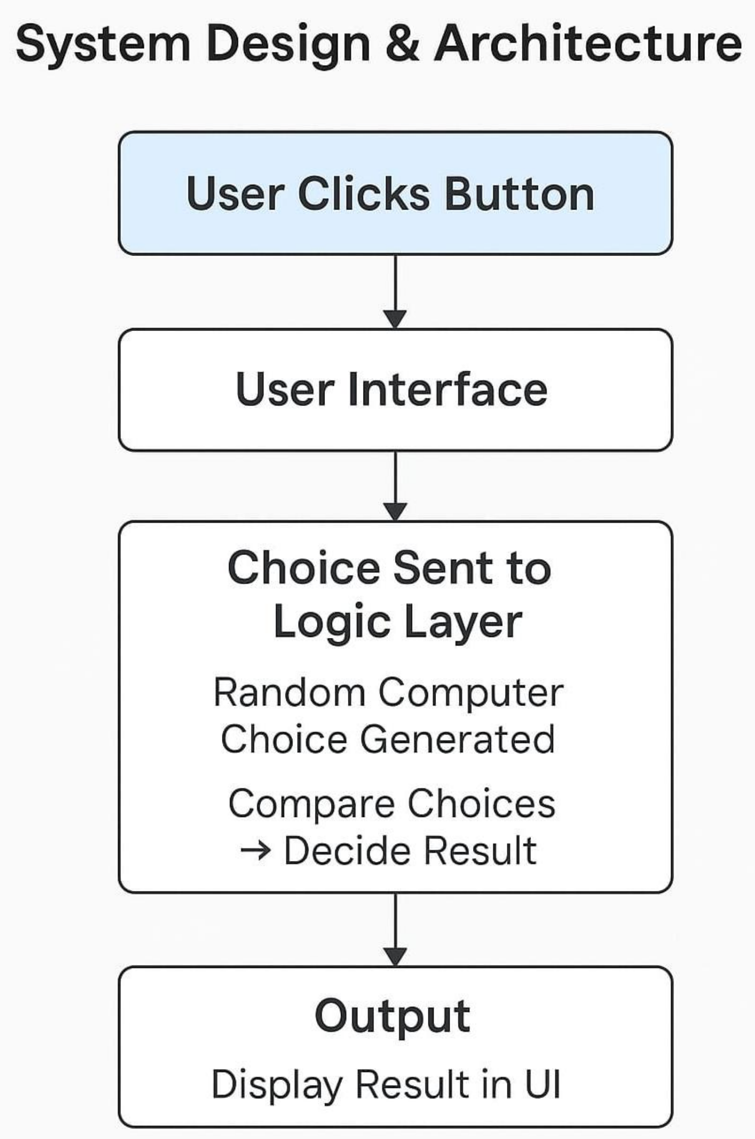
Code editors and ides used for development and debugging.

- Operating system:

Windows / macos / linux (any system with python installed)

- Command line / terminal:

To run and test the application during development.


System Design & Architecture
User Clicks Button
User Interface
Choice Sent to Logic Layer
Random Computer Choice Generated
Compare Choices → Decide Result
Output
Display Result in UI

SYSTEM DESIGN AND ARCHITECTURE:-

- 1. User interface layer (frontend)
- Built using tkinter
- Includes buttons for user input: rock, paper, scissors
- Displays: player's choice,computer's choice,game result (win/lose/draw)
- 2. Logic layer (backend)
- Written in python
- Handles: user input (button clicks),computer’s random choice using random.Choice(),game outcome logic using if-else conditions
- 3. No database / storage layer
- This version does not use file saving or persistent storage
- All actions are handled in-memory during runtime

IMPLEMENTATION DETAILS:-

LOGIC:

- Choices are handled using if-else logic to compare player vs. Computer. The result is updated dynamically on the screen.

KEY FUNCTIONS:

- 1. Play(player_choice) – handles game logic and updates labels.
- 2. Random.Choice() – picks the computer's move.
- 3. Label() – displays result, user and computer choices.

CODE STRUCTURE:

- Main file – contains GUI, logic, and gameplay using tkinter.
- Functions – event-handling for buttons and game outcome.
- Layout – designed using pack() for placing widgets neatly.

USER INTERFACE:-

↓

Title displayed at the top: "rock-paper-scissors game"

↓

Three buttons for user choices: rock, paper, scissors

↓

Labels to show: Player's selected choice, Computer's randomly chosen move, Game result (win, lose, or draw)

↓

Result text appears below buttons for instant feedback

↓

Color coding or font styling to highlight the outcome

↓

Simple and intuitive layout using tkinter's pack() method

CHALLENGES FACED:-

- Randomness handling:Ensuring the computer's choice is truly random for fair gameplay.
- UI responsiveness:Maintaining a smooth and responsive interface with instant feedback on button clicks.
- Result logic accuracy:Implementing accurate condition checks to correctly determine win, lose, or draw outcomes.
- Input validation:Preventing any errors due to unexpected user inputs or missing selections.
- Code optimization:Keeping the code clean, modular, and readable for easier debugging and future improvements.
- Aesthetic design:Balancing a simple yet engaging visual appearance with functional layout using tkinter.

FUTURE SCOPE AND SYSTEM REQUIREMENTS:-

->FUTURE SCOPE:-

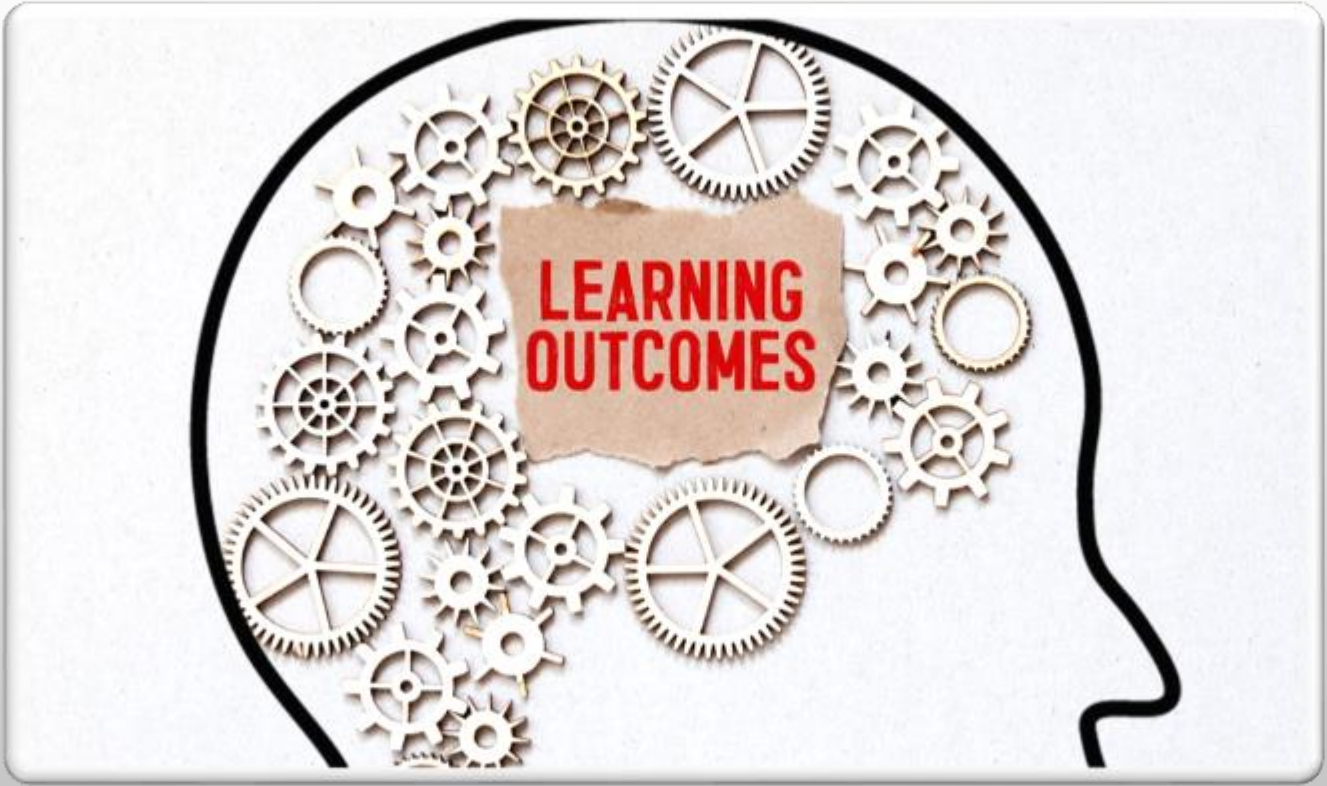
- Add multiplayer support for playing with friends
- Show scoreboard to keep track of wins and losses
- Add match history to review previous rounds
- Use AI to make the computer's moves smarter
- Improve graphics and animations for better user experience
- Add user profiles to save scores and settings
- Expand to mobile apps for android and ios

->SYSTEM REQUIREMENTS:-

- OS: windows, macos, or linux ,python 3.X installed
- Aleast 2 GB RAM ,basic processor (i3 or higher recommended)
- No external libraries needed (uses built-in modules like tkinter and random)
- Optional: code editor like VS code or IDLE for running the code

LEARNING OUTCOMES:-

- The internship project taught me:
- How to structure and manage a small to medium scale software project.
- The importance of clean code and modular programming
- Hands on experience in creating user friendly interface
- Basic of software testing and debugging
- Effective use of version control systems



CONCLUSION:-

The rock-paper-scissors game project enhanced my understanding of python basics, GUI design with tkinter, and logical decision-making through conditionals. It gave me hands-on experience in building an interactive app, managing user input, and creating a smooth user experience. Though simple, it strengthened my coding skills and prepared me for more advanced projects.