Project Report

on

Endless Runner Game
Linux Administration Lab
(25CAP-622)

Submitted by

Nitin Rajput

25MCC20018

Under the guidance of

Mr. Navdeep Singh Sodhi

in partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

CLOUD COMPTING & DEVOPS

Chandigarh University

November 2025

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+
Accredited University

Certificate

This is to certify that **Nitin Rajput**, a student of Master of Computer Applications (MCA) Cloud Computing & DevOps, has successfully completed the Minor Project titled **"Endless Runner Game"** under the esteemed guidance of **Mr. Navdeep Singh Sodhi**, Assistant Professor, University Institute of Computing (UIC), Chandigarh University.

This project was undertaken as part of the academic curriculum and is submitted in partial fulfillment of the requirements for the MCA program. The work presented is the result of independent research, diligent effort, and dedication, demonstrating the student's ability to apply theoretical knowledge to practical problem-solving..

The **Endless Runner Game** is a Python-based terminal game where the player must jump over obstacles to score points. The project incorporates Python programming, event-driven input handling, random obstacle generation, collision detection, game loop design, and scoring mechanics. The project demonstrates how Python, combined with Linux terminal manipulation, can be used to create interactive, real-time applications**.**

I hereby confirm that this project is an original work carried out by the student and has not been submitted elsewhere for the award of any other degree, diploma, or certification

Project Guide:
Mr. Navdeep
Singh Sodhi
Assistant Professor
University Institute of Computing
Chandigarh University

Acknowledgement

I sincerely express my gratitude to **Chandigarh University** and the **University Institute of Computing (UIC)** for providing me the opportunity to undertake this project, **"Endless Runner Game."** This project allowed me to apply my theoretical knowledge of Python programming into a practical, functional, and interactive application.

I extend my heartfelt appreciation to my mentor, **Mr. Navdeep Singh Sodhi**, for his constant guidance, insightful suggestions, and technical support throughout the project. His expertise in Python programming, terminal-based application development, and event-driven programming was instrumental in completing this project successfully..

I am also grateful to my friends and peers for their continuous encouragement, valuable feedback, and constructive discussions, which helped refine my approach and improve the game's features and performance. Their support has been invaluable throughout the development process.

This project has not only strengthened my coding skills but also enhanced my problem-solving ability, logical reasoning, and creativity. I hope that the **Endless Runner Game** serves as a solid foundation for my future projects in Python game development, software design, and interactive applications.

Nitin Rajput
MCA – Cloud computing & DevOps
Chandigarh University

CONTENTS

1. Introduction

In the era of digital computing, interactive applications and games have become a significant means of learning, entertainment, and skill development. Python, being a versatile and high-level programming language, offers an excellent platform for creating both GUI-based and terminal-based applications. Among various types of games, **Endless Runner Games** have gained popularity due to their simplicity, engaging gameplay, and requirement for reflex-based interactions. In an endless runner game, the player character continuously moves forward while the player must avoid obstacles by jumping or performing other actions. The simplicity of the game mechanics allows developers to focus on learning core programming concepts such as loops, event handling, collision detection, and scoring systems.

The **Endless Runner Game developed in Python is designed to run in a Linux terminal, making it lightweight, resource-efficient, and ideal for Linux-based systems like Ubuntu, Debian, and Mint. The game uses Python modules such as os for clearing the terminal screen, time for managing frame rate and speed, random for generating obstacles unpredictably, and system-level modules like termios, tty, and select to handle real-time key presses. These modules collectively provide an environment to simulate a game loop that updates player position, generates obstacles dynamically, checks for collisions, and increments score with each frame.**

The design of this project provides hands-on experience in **event-driven programming**, where user inputs (like pressing the spacebar to jump) directly affect the game state. Students learn how to implement responsive and interactive programs that react in real-time. The project also emphasizes **collision detection logic**, a critical component in gaming applications, which teaches how to track object positions and detect overlaps to determine game outcomes. This project also emphasizes **file handling and operating system integration**. Working on Linux helps students understand file management, directory navigation, and safe interaction with system resources. It equips students with practical skills in Python GUI programming, interactive application design, and user-centered software development.

The **Endless Runner Game, students gain practical knowledge of Python programming, Linux terminal operations, and interactive application design, which can serve as a foundation for advanced projects in game development or software engineering.**

2. Objective

1. Primary objective of the Endless Runner Game project is to develop a fully functional, interactive, and engaging terminal-based game using Python, with emphasis on real-time input handling, event-driven programming, and logical problem-solving. This project demonstrates the application of theoretical programming concepts to a practical and interactive scenario..

2. The project also emphasizes Linux terminal programming and environment management. By building the game for Ubuntu or other Linux systems, students learn terminal manipulation, clearing and updating screen content, and integrating system-level Python modules (os, termios, tty, select) for efficient input handling.

3. the project aims to enhance problem-solving, debugging, and optimization skills. Students must ensure smooth gameplay, minimal lag, responsive controls, and accurate collision detection. The project encourages modular programming, testing, and iterative development, fostering strong software engineering practices.

4. the objectives of the Endless Runner Game include building an interactive, responsive game, understanding event-driven programming, mastering terminal-based input/output operations, implementing dynamic game logic, tracking performance metrics, and applying Python programming knowledge to a practical, real-world application. This foundation equips students with skills applicable to advanced game development, software design, and interactive application programming.

3. Tools and Technologies Used

- The Endless Runner Game project is developed using a combination of programming languages, libraries, and system-level functionalities that together provide a robust and interactive terminal-based game
- **Python 3** is the primary language used for development. Python's simplicity, readability, and extensive standard library make it ideal for building interactive programs. Its support for procedural and object-oriented programming enables clean, modular code and easy maintenance. Python also provides system-level modules that allow real-time user input handling, terminal manipulation, and efficient computation, making it suitable for developing this game.
- **Linux/Ubuntu Operating System** is chosen as the platform for the game. Linux provides a stable and open-source environment where terminal applications can run efficiently. Developing on Linux familiarizes students with command-line interfaces, terminal-based interaction, and system-level programming, all of which are critical for professional software development in server-side applications or scripting tasks.

The project uses several Python modules to implement core functionalities:

- **os:** Enables clearing the terminal screen and managing system-level operations.
- **time:** Provides timing functions for controlling game speed and animation frame rate.
- **random:** Used to generate obstacles dynamically and unpredictably, keeping the gameplay challenging.
- **sys, termios, tty, select:** Handle real-time keyboard inputs and ensure non-blocking input detection, which is essential for responsive gameplay.

- The project emphasizes **text-based interface design**, demonstrating that even without graphical libraries like Pygame or Tkinter, interactive and visually intuitive games can be created using terminal characters and symbols. This approach reduces complexity while reinforcing core programming principles such as loops, conditional statements, and array/list manipulation.
- By using Python 3 in a Linux environment with these supporting modules, the **Endless Runner Game** becomes an excellent learning tool. It teaches students terminal-based game development, event-driven input handling, randomization algorithms, and score management.

4. Implementation Steps

- The Endless Runner Game was implemented using Python 3, leveraging system-level modules to create a terminal-based, interactive gaming experience. The project follows a structured development process to ensure clarity, responsiveness, and smooth gameplay.

- **Project Setup:**

  The development environment was set up on a Linux system (Ubuntu). Python 3 was installed along with the required standard modules (os, time, random, sys, termios, tty, select). A dedicated project directory was created to **organize** the source code, assets, and documentation, ensuring proper project management and version control.

- **Terminal Preparation:**

  Since the game runs entirely in the Linux terminal, system-specific configurations were applied. The terminal was set to accept non-blocking keyboard input using termios and tty modules, allowing real-time response to key presses without waiting for the Enter key. The select module was used to detect whether a key had been pressed, enabling responsive and continuous gameplay.

- **Game Loop Development:**

  The core of the game is a continuous event-driven game loop that handles input, updates the game state, manages obstacles, checks collisions, and renders frames in real time. Player movements (jumping) are tracked along with dynamic obstacle generation using randomized intervals. The loop also updates the score continuously, reflecting the player's progress.

- **Input Handling:**

  Keyboard input is central to gameplay. The spacebar triggers a jump when the player is on the ground, while pressing Q exits the game. Event-driven logic ensures that inputs are detected without interrupting the game loop, creating smooth and responsive player control.

- **Frame Rendering:**

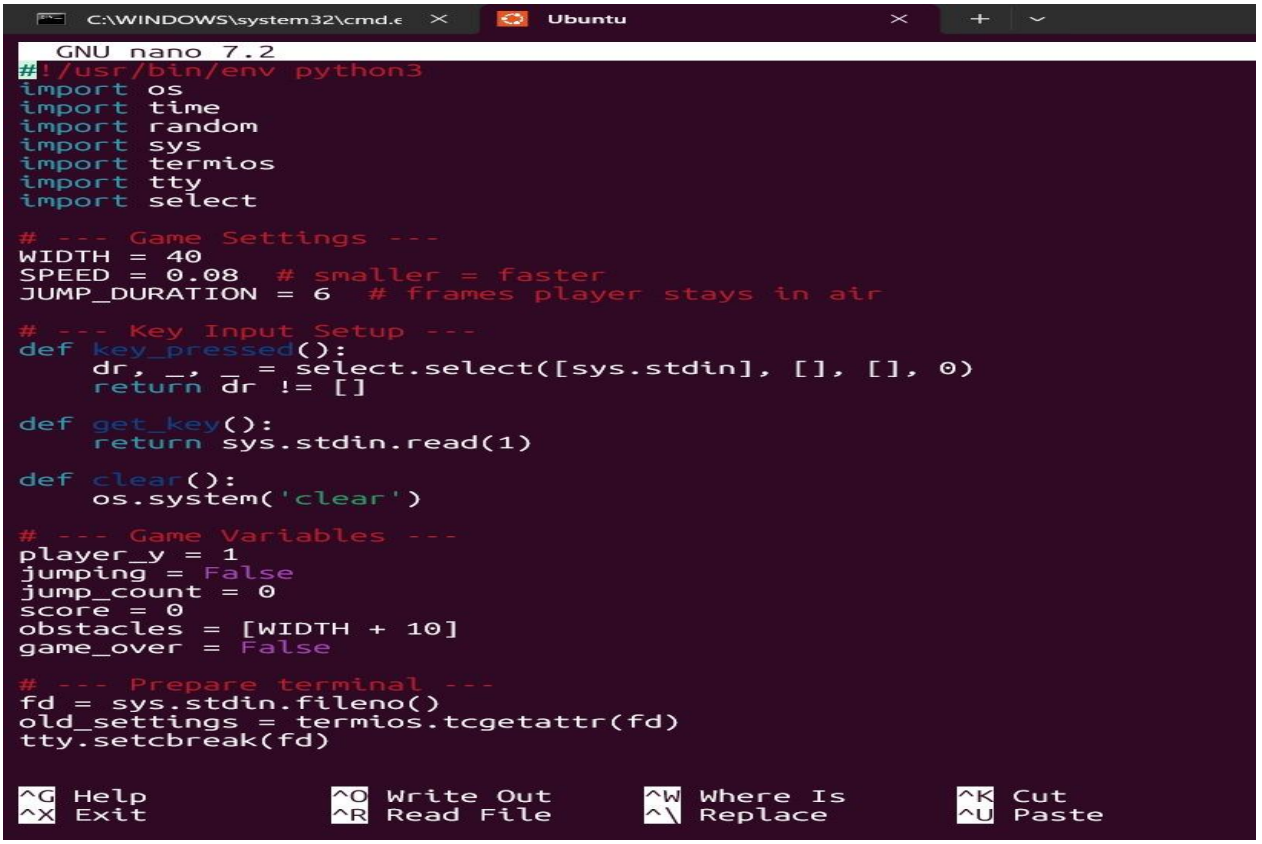
  Each frame is drawn in the terminal using ASCII and Unicode characters.
- **Testing and Optimization:**

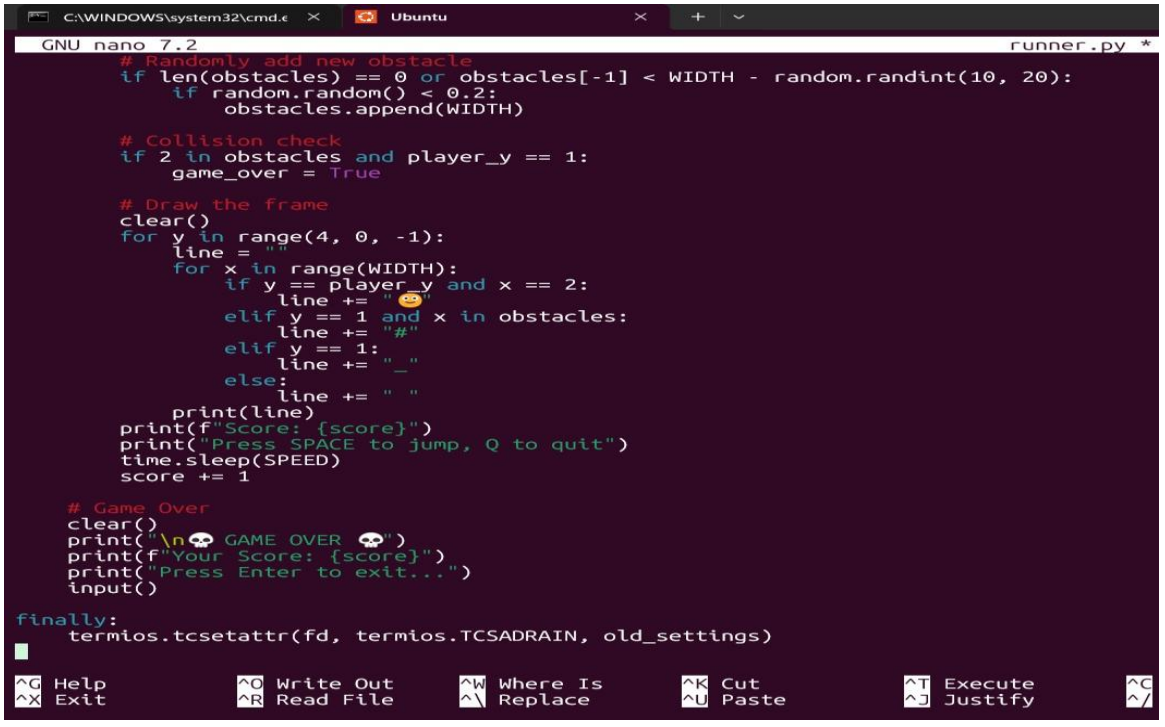
  The game was thoroughly tested on Linux terminals to ensure smooth performance, accurate collision detection, and correct scoring. Adjustments were made to obstacle generation frequency, jump duration, and game speed for optimal difficulty and player experience.

- **Execution:**

  After development, the game is executed from the terminal, providing an interactive, visually clear, and engaging gameplay experience, while demonstrating key programming concepts like loops, conditional logic, and event-driven control.

5. Screenshots/Result






C:\WINDOWS\system32\cmd.e
Ubuntu
GNU nano 7.2
runner.py *
try:
while not game_over:
# Input handling
if key_pressed():
key = get_key()
if key.lower() == 'q':
break
elif key == ' ' and not jumping and player_y == 1:
jumping = True
jump_count = 0
# Update jump
if jumping:
player_y = 3
jump_count += 1
if jump_count >= JUMP_DURATION:
jumping = False
else:
player_y = 1
# Move obstacles
new_obstacles = []
for o in obstacles:
o -= 1
if o >= 0:
new_obstacles.append(o)
obstacles = new_obstacles
# Randomly add new obstacle
if len(obstacles) == 0 or obstacles[-1] < WIDTH - random.randint(10, 20):
if random.random() < 0.2:
obstacles.append(WIDTH)
# Collision check
if 2 in obstacles and player_y == 1:
game_over = True
^G Help
^O Write Out
^W Where Is
^K Cut
^T Execute
^C
^X Exit
^R Read File
^\ Replace
^U Paste
^J Justify
^/

Figure 1: Main Interface of Project

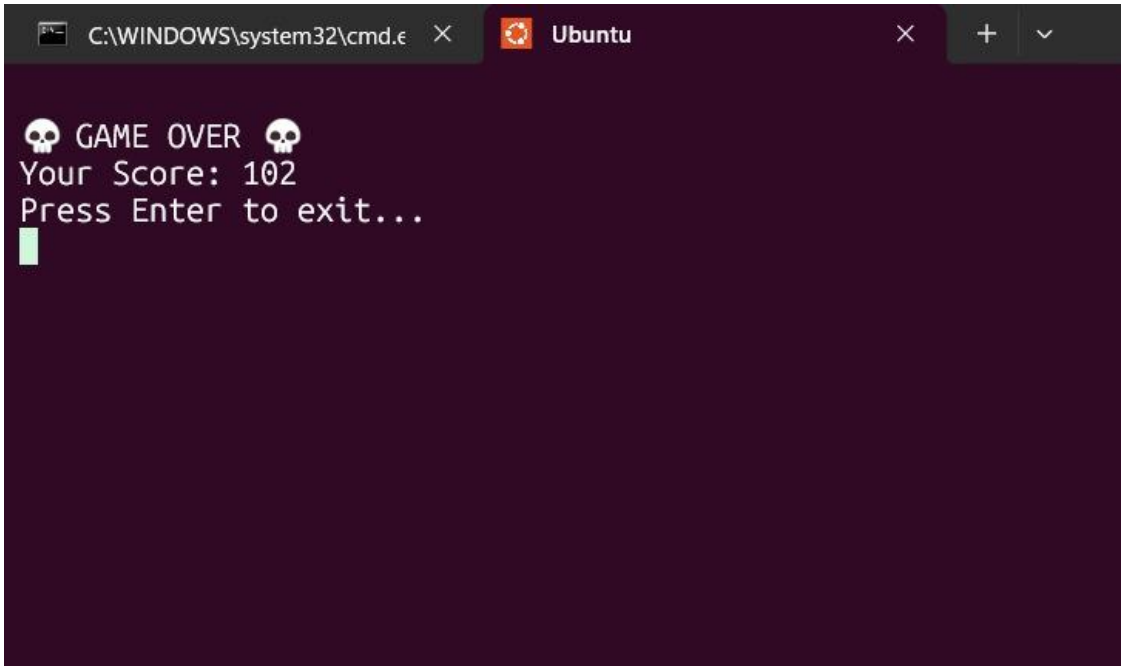


Figure : GameOver Interface of Project

6. Conclusion

The Endless Runner Game project is a terminal-based Python application that demonstrates how theoretical programming concepts can be transformed into a functional, interactive, and engaging software experience. Built entirely using Python, the game integrates core programming principles such as event-driven logic, control structures, loops, and real-time input handling, highlighting the capabilities of the language for building small-scale yet practical projects. By focusing on terminal graphics using ASCII and Unicode characters, the game illustrates how creativity can transform a simple text-based interface into an enjoyable interactive environment.

One of the primary accomplishments of this project is the implementation of a responsive gameplay system. The game's mechanics, including jumping, collision detection, and obstacle generation, are executed with precision, ensuring that player actions translate seamlessly into on-screen results. This responsiveness is essential in an endless runner game, where timing and reflexes play a crucial role in the overall user experience. The dynamic scoring system reinforces engagement, as players receive instant feedback on their performance and are encouraged to improve through repeated attempts.

The project also provides a comprehensive understanding of handling real-time input in a terminal environment using Python modules such as sys, termios, tty, and select. These modules facilitate smooth keyboard input capture and enable the game to process commands without interrupting gameplay. Additionally, the obstacle generation algorithm introduces randomness, making each session unique and challenging, which enhances replayability and adds depth to the user experience.

From an academic perspective, the Endless Runner Game reinforces essential software development skills, including problem-solving, logical thinking, debugging, and code optimization. Students learn how to structure a project, manage game states, and ensure consistent frame updates, all of which are transferable skills for larger programming projects. Furthermore, the project demonstrates how lightweight, text-based applications can still provide immersive and interactive experiences without relying on complex graphical libraries.

References

- **GitHub link – https://github.com/Nitinrajput07/Endless-Runner-Game**
- **Python Documentation – https://docs.python.org**
- **Python Terminal Input Handling – https://docs.python.org/3/library/tty.html**
- **Python select Module Documentation – https://docs.python.org/3/library/select.html**
- **Python termios Module Documentation – https://docs.python.org/3/library/termios.html**
- **Python Event-Driven Programming Concepts –https://realpython.com/python-gui-tkinter/**
- **Linux Terminal and Command-Line Guide – https://ubuntu.com/tutorials/command-line-for-beginners**