**THE GALAXY PROJECT**

### PROJECT REPORT

#### Submitted by

**Sanyam Gupta (2210993842)**

**Pratham Gupta (2210993827)**

Logo

Description automatically generated

**BE-CSE (Artificial Intelligence)**

***Guided by***

**Dr. Amanpreet Singh**

**CHITKARA UNIVERSITY INSITUTE OF ENGINEERING & TECHNOLOGY**

**CHITKARA UNIVERSITY, RAJPURA**

**February 2023**

**TABLE OF CONTENTS**

**TOPICS PAGE NO.**

Abstract……………………………………………. 3

1. Acknowledgement……………………………... 4
2. Chapters:

2.1 Introduction………………………………… 5

2.2 Problem Formulation………………………. 6

2.3 Methodology……………………………….. 7

2.4 Flowchart/ system flow diagram…………… 10

2.5 Software & Hardware requirement………… 13

2.6 Code………………………………………... 13

2.7 Results……………………………………… 17

2.8 Conclusion…………………………………. 19

1. Reference 19

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| S. NO. | LIST OF FIGURES | NAME | PAGE NO. |
| 1 | 1 | V1 Flowchart | 10 |
| 2 | 2 | V2 Flowchart | 11 |
| 3 | 3 | V3 Flowchart | 12 |
| 4 | 4.1 | V1 Output | 17 |
| 5 | 4.2 | V2 Output | 17 |
| 6 | 4.3 | V3 Output | 18 |
| 7 | 4.4 | V3 Output | 18 |
| 8 | 4.5 | V3 Output | 18 |

**ABSTRACT**

“THE GALAXY PROJECT” is a Python based game comprising of Kivy Library.

Kivy is an open-source, cross-platform software development kit for creating applications with user interfaces. It is written in Python and designed to be used for creating applications that can run on multiple platforms, including Windows, macOS, Linux, Android, and iOS. Kivy utilizes a novel approach to user interface design, allowing developers to write user interfaces in Python code and utilizing a custom language for defining the layout of the interface.

Kivy is designed to be easy to use and allows developers to create highly interactive, responsive, and aesthetically pleasing user interfaces. It offers a wide range of widgets, such as buttons, text inputs, sliders, and more, which can be combined and customized to fit the needs of any particular application. Kivy also provides support for touch input, making it suitable for use on mobile devices.

In addition to its user interface capabilities, Kivy also offers a number of features for game development, including support for hardware acceleration, multi-touch input, and gestures. It also includes a powerful event system, allowing developers to easily create interactions between different elements of the user interface.

Overall, Kivy is a powerful and versatile tool for software development that allows developers to create high-quality applications for multiple platforms with a minimum of effort.

Keywords: Python, Kivy, problem solving, multi – platform interface, problem formulation, multi – touch function.

1. ACKNOWLEDGEMENTS

With immense pleasure we, **Sanyam Gupta** and **Pratham Gupta** are  
presenting “THE GALAXY PROJECT” project report as part of  
the curriculum of ‘ BE-CSE (AI) ’.

I would like to express my sincere thanks to **Dr. Amanpreet Singh**, for his/her valuable guidance and support in completing my project.

I would also like to express my gratitude towards our dean **Dr. Sushil Narang** for giving me this great opportunity to do a project on “THE GALAXY PROJECT”. Without their support and suggestions, this project would not have been completed.

Signature: **……………...**

Name: **Sanyam Gupta** Name: **Pratham Gupta**

Roll No: **2210993842** Roll No: **2210993827**

**2.1 INTRODUCTION**

What’s Galaxy?..

* In this game, the player must navigate the spaceship to the left or right side to keep on the track, but if you get out of the track, your game is over.
* This project will include a grid formation with perspective and movement, random land generation algorithm for ship and to finalise the project, it’ll include a menu, scoreboard, custom background and custom background sounds.
* This project is sub-divided into 3 parts – V1, V2 and V3:
  + V1 - Grid, Perspective and Landscape Movements
  + V2 - Land Generation, Displaying the ship and Collisions
  + V3 - Finalising the game by adding Menu, Sounds, Background and Score

**PART WISE PROJECT DISTRIBUTION OF V1, V2, V3**

Figure 1

* 1. **Problem Formulation**
* The galaxy project is a partial 3-dimensional game where the player has to navigate a spaceship with cursor touch function in order to stay on the path slabs.
* If the player fails to follow up to the above criteria, then it’s game over.
* Input for cursor controls can be given using a mouse or a touch pad.
* As the player advances in the game, his scores will gradually increase with respect to the number of tiles travelled, the longer the player lasts in the game, the more score he gets, thereby making the Galaxy Game an obstacle avoiding game.
* Key Features of Kivy:

1. Cross-Platform Development: Kivy supports multiple platforms, including Android, iOS, Windows, and Linux. This allows developers to write code once and deploy it on multiple platforms.
2. User-Friendly Interface: Kivy provides a user-friendly interface that enables developers to create applications with a smooth and intuitive user experience.
3. Graphics Library: Kivy uses a powerful graphics library called OpenGL that allows developers to create visually appealing applications with a wide range of functionality.
4. Touch Input Support: Kivy provides support for touch input, making it an ideal choice for developing applications for mobile devices.
5. Widgets: Kivy provides a wide range of widgets that developers can use to create applications. These widgets include buttons, labels, text boxes, and more.
6. Modular Architecture: Kivy has a modular architecture that allows developers to easily add new functionality to their applications.
7. Good Documentation: Kivy has good documentation, making it easy for developers to learn how to use the framework and get started with developing applications.

* Applications of Kivy:

1. Mobile Applications: Kivy is widely used for developing mobile applications due to its support for multiple platforms and touch input.
2. Desktop Applications: Kivy can also be used for developing desktop applications, including games, utilities, and more.
3. Prototyping: Kivy is often used for prototyping due to its user-friendly interface and fast development time.
   1. **Methodology**
4. As talked earlier, this project comprises of 3 parts i.e., V1, V2 and V3.
5. For the project initialisation, **V1** is taken into account:

The initialisation of the project is with the base layout, for which we’ll use Grid Layout as it arranges everything in the form of matrix.

Now, in order to add the forward motion of the ship, we’ll have to change the position of the matrices of the grid layout such that grid lines of x and y axis begins to move backwards.

We have also added a perspective point from where each line generates. This perspective point also acts for the point from where path tiles generate.

Touch controls have also been added in this part of the project.

1. Now, we have taken **V2** into account for the following functions:

V2 adds the ship using canvas functions, along with the action to be taken after ship’s collision. As the grid moves backwards, the ship will eventually get out of the grid, in order to solve this, ship’s coordinates will update with respect to the coordinates of the transformed function of gird.

The path tiles for the ship should be generated in a random order for unexpected algorithm for tile formation, in order to do that, we’ll use random library.

At some point, if the ship gets out of the path tiles, a collision will occur, resulting in ending of the game. With every touch movement, collision coordinates for ship will also change. With each ending, the score board will also get reset.

1. For the rest, we have taken **V3** into account:

For finalising the game, elements like menu, sounds, background etc. are taken into consideration. For the menu screen, we have used relative layout.

As the program stars, the menu screen will get commenced out first with a start button and when the ship, collides, menu screen will be commenced again with a restart button.

In addition to the buttons, the screen will also contain score and title of the game. When the start button is pressed, a voice will declare the initialisation of the game.

When the player plays the game, a background music will also be played in a loop.

When the game ends, another voice will declare the ending of the game. This part of the code also contains an image that will be displayed as the background screen.

* **V\_NB\_LINES –** Contains the number of vertical lines**.**
* **V\_LINE\_SPACING –** Displays space between vertical lines.
* **H\_NB\_LINES –** Contains the number of horizontal lines.
* **H\_LINES\_SPACING –** Displays space between horizontal lines.
* **init\_vertical\_lines function –** Will append and display the vertical lines one after another in y axis.
* **init\_horizontal\_lines function –** Will append and display the horizontal lines one after another in y axis.
* **transform\_2d function –** Will change the perspective from 3d to 2d.
* **transform\_perspective function –** Will change the perspective from 2d to 3d with respect to a perspective point.
* **SPEED –** Defines a variable that will change the speed of y-axis.
* **SPEED\_X –** Defines a variable that will change the speed of x-axis.
* **update\_vertical\_lines and update\_horizontal\_lines functions –** Will update the vertical lines and horizontal lined by implementing a for loop.
* **NB\_TILES –** Will contain the number of tiles to be displayed.
* **init\_tiles function –** Will contain append function with loop for each tile.
* **pre\_fill\_tile\_coordinates –** Will fill and display the initial tiles straight.
* **generate\_tiles\_coordinates –** will contain the algorithm to display path tiles randomly in three directions, i.e., left, right and straight.
* **update function –** Will update tiles, lines and ship movements.
* **init\_ship functions –** Will display the ship.
* **update\_ship function –** Will update the ship coordinates with respect to change in y-axis.
* **check\_ship\_collision –** Will check if the ship collides with the non-path tiles or not.
* **check\_ship\_collision\_with\_tile –** Will change the collision hitbox for ship.
* **menu\_title –** Will update the text for menu screen.
* **menu\_button\_title –** Will update the text for menu screen button.
* When the ship collides **state\_game\_over** will be True and **menu\_title** will display **“**GAME OVER” and **menu\_button\_title** will display “Restart”.
* **init\_audio –** Will load audio files from the designated folder with volume.
* **on\_menu\_button\_pressed –** Contains “.play” argument which will play thedesignated sounds.
* **play\_game\_over\_voice\_sound –** Will play sound when the ship crashes.
* “.kv” file will contain canvas argument which will contain source for background image.
* **score\_txt –** Contains the updated score as the player advances in the game.
  1. **FLOWCHART**

Input the perspective for landscape with vertical and horizontal lines

Update Horizontal line function will update the number of lines in vertical with respect to the perspective point.

Update vertical line function will update the number of lines in vertical with respect to the perspective point.

Use Transform Perspective to change the perspective from 2-D to 3-D.

Use for loop in order to repeat the instructions repeatedly for landscape’s algorithm.

Use keyboard down function to specify control keys for the game; Use on desktop to check if game is running on desktop or not.

Use current offset x variable to update landscape for horizontal lines.

Use on touch down function to change the direction of landscape.

Use current offset Y variable to update landscape for vertical lines.

Use speed variable in order to update the speed for the vertical lines.

Update function will contain the instructions for updating layout for ship’s movement.

Figure 1

Use get\_tile\_coordinates function to get coordinates for the tiles.

Use get\_line\_x\_from\_index and get\_line\_y\_from\_index function to specify path tiles for ship.

Use init\_tiles function to display tiles and update\_tiles function to update the position of tiles.

Use generate\_tile\_coordinate (containing algorithm for generating tiles at random positions) to generate random pattern of path tiles.

Use init\_ship function to display the ship.

If False

If True

Use check\_ship\_collision\_with\_tile to check if the ship collides.

Use update\_ship function to update the position of the ship.

Figure 2

Use state\_game\_has\_started function to check if the game has started or not.

Use ship\_game\_over to check if the game has ended or not.

If false

If True

Use reset\_game to restart the game

We’ll make a menu.kv file to display the menu for galaxy game.

Figure 3

* Figure 1: Represents the flowchart of V1
* Figure 2: Represents the flowchart of V2
* Figure 3: Represents the flowchart of V3

**2.5 SOFTWARE,HARDWARE REQUIREMENTS**

**Software required:**

➤ Python Interpreter

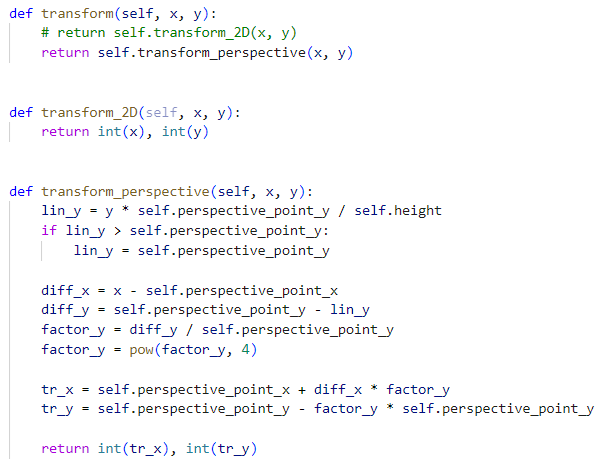
➤ Python Libraries and Packages

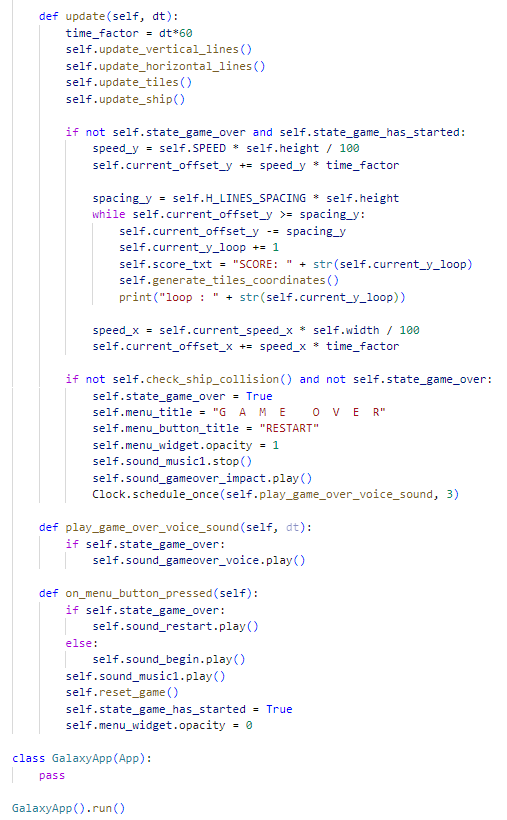
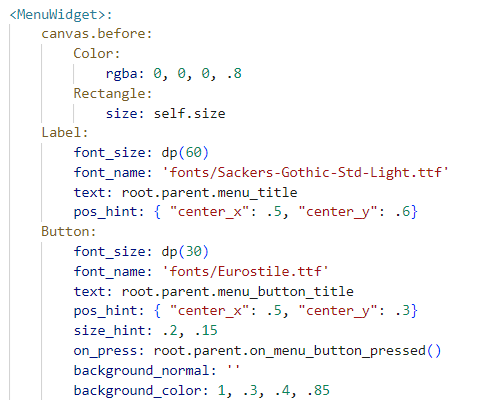
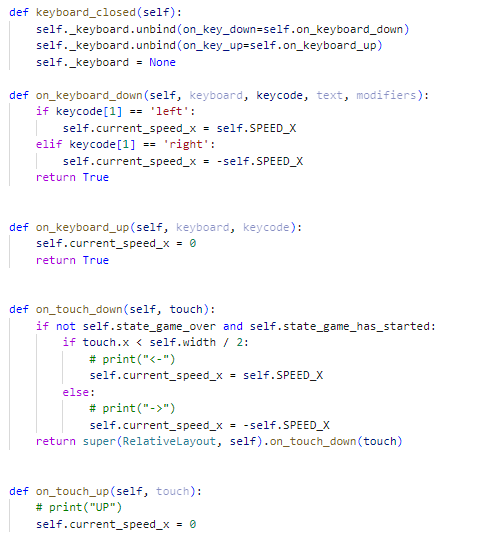
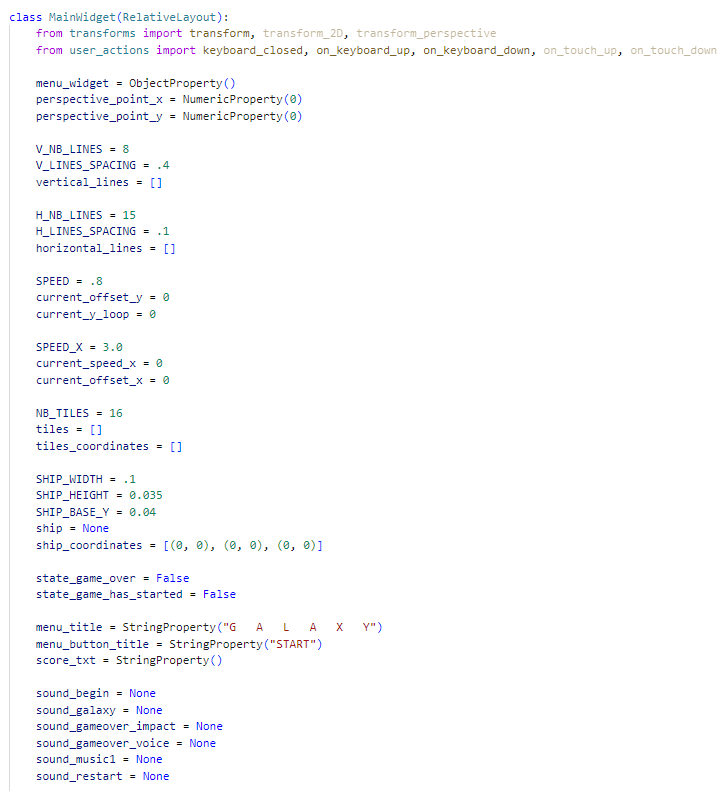
➤ Integrated Development Environment (IDE) like PyCharm, Visual Studio Code, Spyder etc..

**Hardware required:**

➤ Mouse/ Touch pad

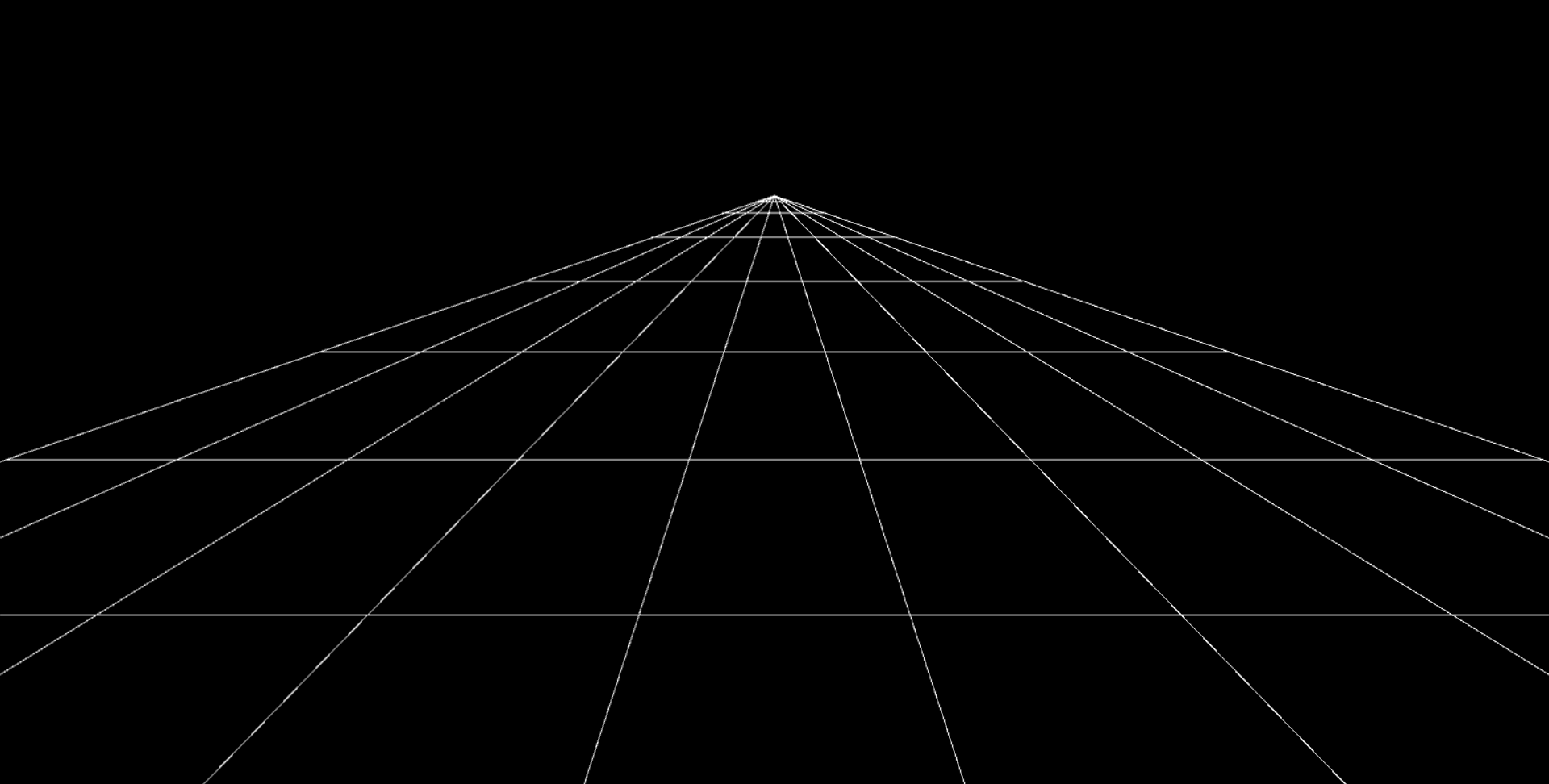
**2.6 CODE**

****

****

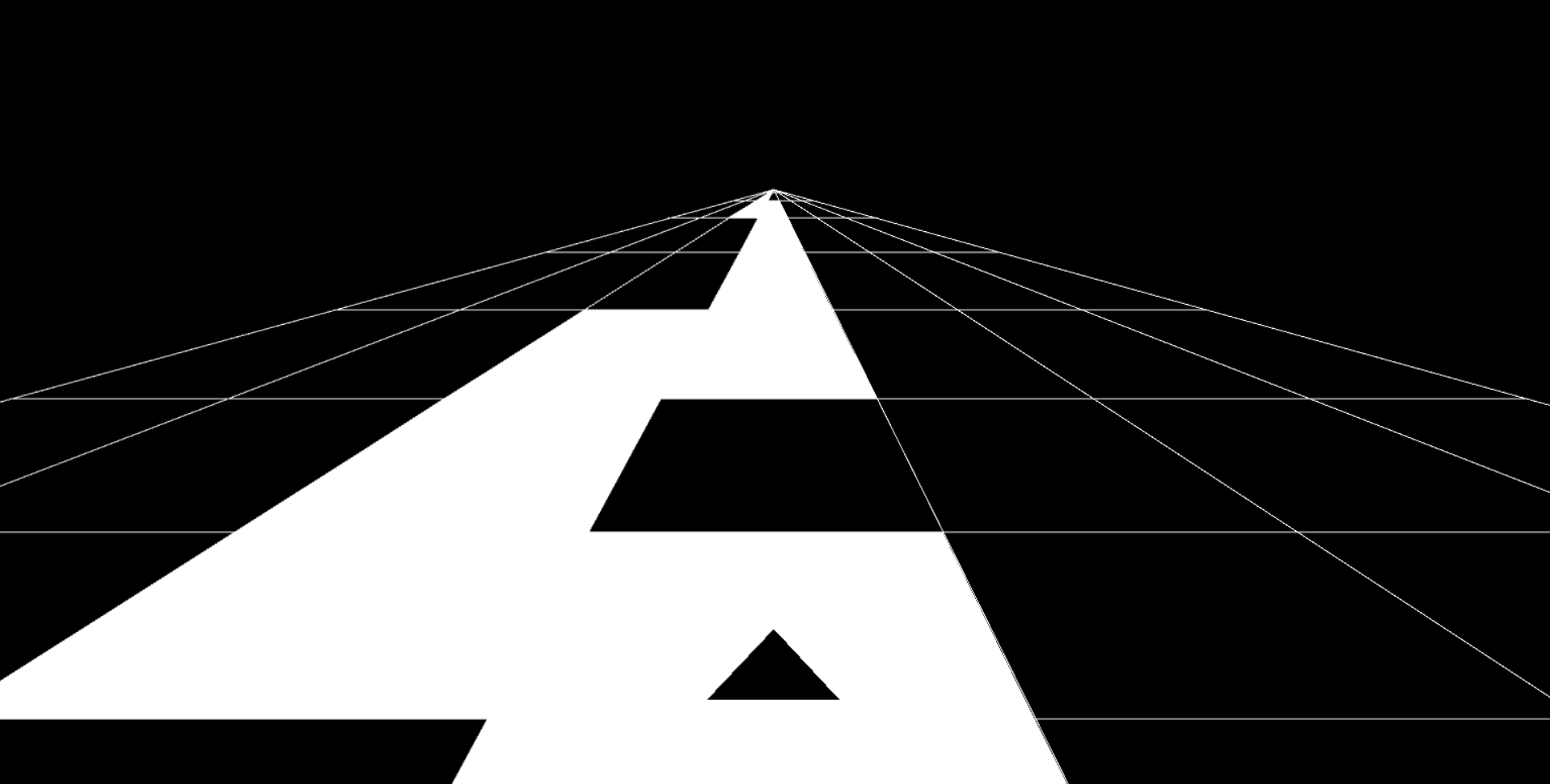
* 1. **RESULTS**

**V1 OUTPUT:**

****

**Figure 4.1**

**V2 OUTPUT:**

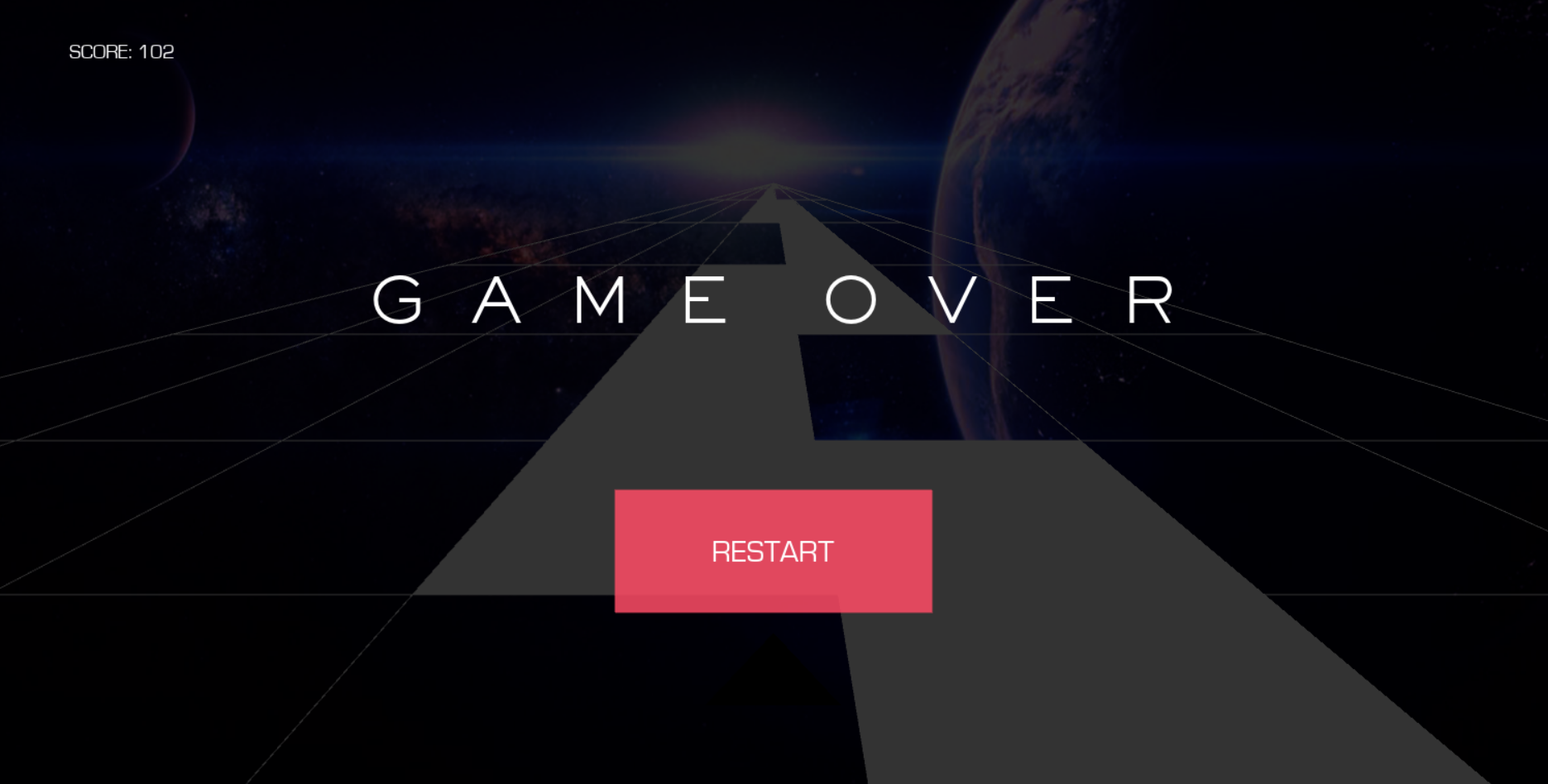
****

**Figure 4.2**

**V3 OUTPUT:**

****

**Figure 4.3**

****

**Figure 4.4**

****

**Figure 4.5**

* 1. **CONCLUSION**

In conclusion, Kivy is a powerful and flexible open-source Python framework for the rapid development of cross-platform applications. Its support for multiple platforms, user-friendly interface, and wide range of functionality make it an ideal choice for developing applications for both desktop and mobile platforms. Additionally, its good documentation and modular architecture make it easy for developers to get started with the framework and add new functionality to their applications.

**3. REFERENCES**

➤ [www.youtube.com](http://www.youtube.com)

➤ <https://www.geeksforgeeks.org>