**PYTHON PROJECT - CSM216**

**Section – K23CH**

**SPACE SHOOTER USING PYGAME**

**Submitted by:**

Akshit Salgundi

**Registration Number:**

12306490

**DATE: - 10/11/2024**

**In partial fulfilment for the requirements of the award of the**

**degree of**

**“B. Tech CSE Data Science and Machine Learning”** 

**“School of Computer Science and Engineering”**

**Lovely Professional University**

**Phagwara, Punjab**

**Table of Contents**

**1.Introduction………………………………………………………… 1**

**2.Objectives and Scope of the Project……………………. 2**

**3.Application Tools…………………………………………………. 4**

**4.Project Design…………………………………………………………. 5**

**5. Screenshots of the Execution……………………………… 6**

**6.FlowChart…………………………..……………………………… 10**

**1.Introduction:**

Combining technical skill with creativity, game creation has long been an intriguing branch of computer science. The goal of this project, Space Shooter Game Using Pygame, is to use the Python Pygame module to create an entertaining, interactive, and aesthetically pleasing 2D arcade game. The main objective is to improve programming abilities in game creation while offering users an engaging gaming experience.  
  
In the space shooter game, the player takes control of a spaceship in a dynamic space environment and must avoid obstacles and defend against waves of hostile ships. Fast-paced action is emphasized in the game, and in order to score highly, players must exhibit quick reactions and clever thinking. The game adds more difficult stages as players progress, increasing replay ability and player interest.

The project tackles the difficulty of creating a working game engine that incorporates essential elements like sound effects, character movement, collision detection, and scoring systems. To provide a user-friendly experience, graphical user interfaces (GUIs) must also be implemented. This project's instructional value is what makes it significant since it shows how to efficiently handle physics, game dynamics, and graphics rendering in Python.  
  
As a resource for prospective developers, our goal in finishing this project is to add to the expanding collection of beginner-friendly gaming projects. It also emphasizes how Python and Pygame may be used to create interactive applications outside of the realm of traditional software development.

1. **Objectives and Scope of the Project**

The Shooter Game in Space Using The goal of the Pygame project is to create an entertaining 2D arcade game. Designing and implementing a dynamic gaming environment where players may steer a starship, move through space, and engage waves of enemy ships while dodging obstacles is the main goal. This project improves the developer's technical abilities while offering an engaging and dynamic experience that demonstrates the actual application of game development ideas.

**Objectives:**

1.Game mechanics: Create responsive and fluid player controls for the spaceship's firing, mobility, and collision avoidance.  
2.Enemy AI: Construct increasingly difficult enemy ships with different characteristics and patterns of movement.  
3.Level Design: Use a system of level advancement to make the game harder as the player gets better.  
4.System of Scoring: Create a system of scoring to monitor player performance and promote high-scoring accomplishments.  
5.User Interface: To improve the game experience, create clear menus, health bars, and other visual cues.  
6.Sound and Visual Effects: To make the game more captivating and immersive, use sound effects and animations.

**Scope:**

This project's scope involves using the Pygame library to create a fully working game engine. Important game development ideas including event handling, sprite management, collision detection, and frame rate control will all be illustrated in this project. A well-designed user interface with options to start, pause, and display top scores will also be included in the game.  
  
This game is aimed mostly for beginners and intermediate players, but it may also be used as a springboard for future creators who want to experiment with more complex game creation ideas. The project's modular design ensures long-term survival and adaptability by enabling future improvements like power-ups, additional opponent kinds, and multiplayer capabilities.

**3.Application Tools:**

The foundation of this project is the Pygame library. A cross-platform collection of Python modules called Pygame was created especially for creating video games. It makes it easier to integrate essential game elements like sound integration, event management, and graphics rendering. The project uses Pygame's collision detection and sprite management features to produce a fluid gameplay experience.

The project utilizes PyCharm, a well-known integrated development environment (IDE) designed specifically for Python, to write and manage the code. PyCharm has several features that improve productivity and expedite the development process, such as intelligent code help, debugging tools, and an easy-to-use interface. It is especially well-suited for jobs involving game development because of its capacity to effectively handle big projects.

Any software project must have version control, and although Git might be used, it is not a key tool in this one. Git may be included into future improvements to better monitor changes and facilitate collaboration. All things considered, these technologies and tools work together to create a solid, intuitive, and visually appealing game, guaranteeing a seamless workflow and excellent results.

**4.Project Design**

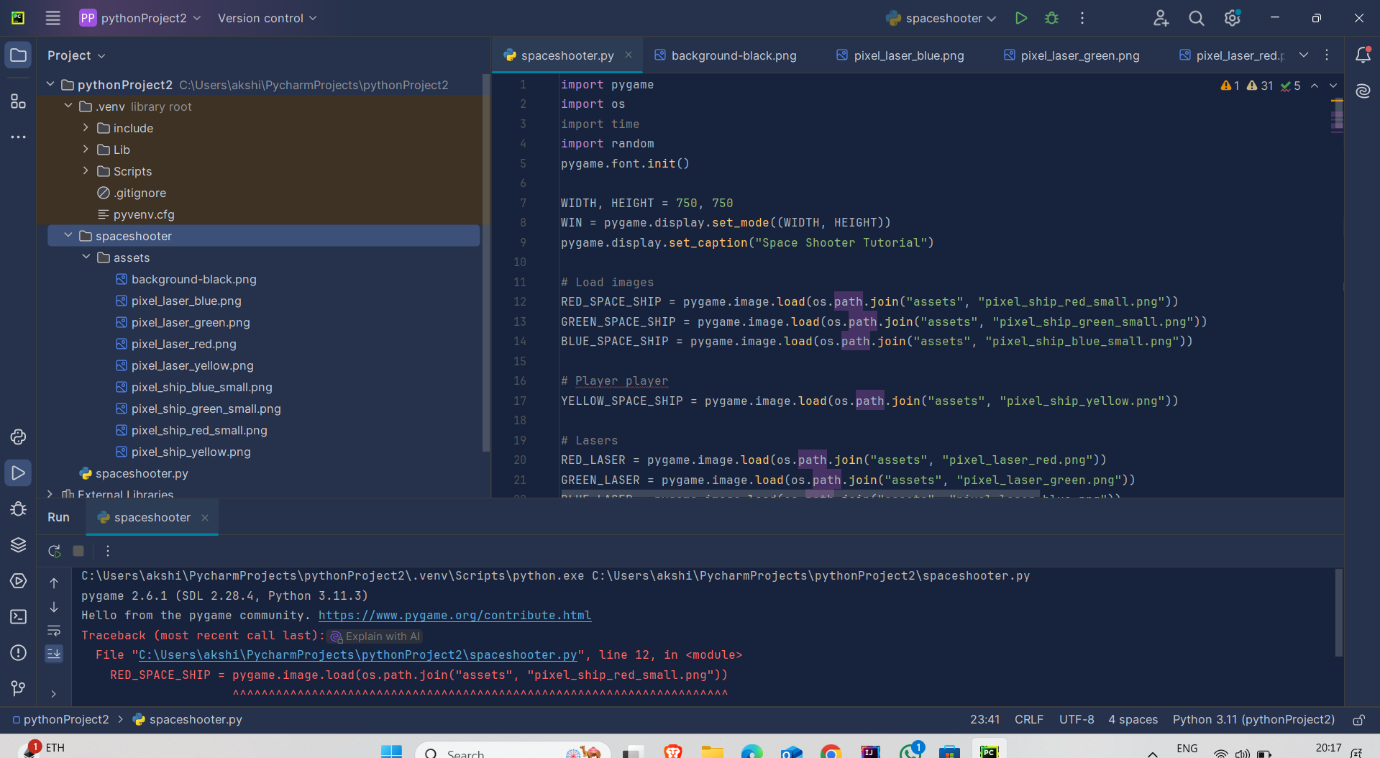
The Shooter Game in Space Making use of Pygame's architecture is modular, with several parts, methods, and classes cooperating to accomplish its goals. The three main classes that make up the structure are Laser, Ship, and their subclasses Player and Enemy. These classes together capture the core mechanics of the game.

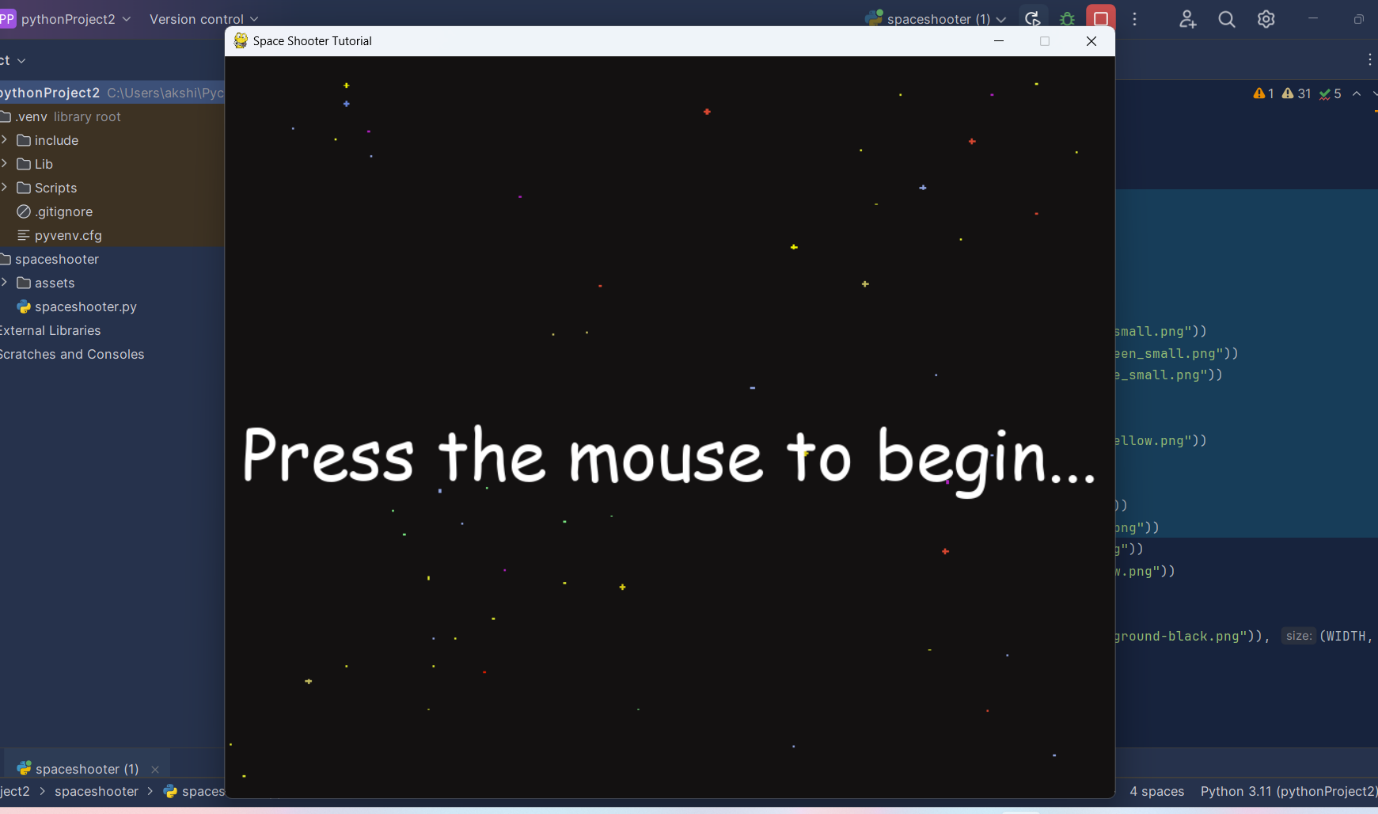
**Principal Elements:**  
**1.Lasser Class:** Projectiles launched by the player and opposing ships are represented by the Laser class. It includes functions for moving lasers (move), drawing them on the screen (draw), and determining if they are off-screen (off\_screen). To identify collisions with other objects, it also has a collision method.

**2.Ship Class:** Acting as a basic class for all ships, Ship specifies common characteristics like health, location (x, y), and a laser cooldown mechanism. It contains methods for shooting lasers (fire), moving lasers (move\_lasers), and drawing the ship and its lasers (draw). Additionally, it has useful functions for managing sprite dimensions, such as get\_width and get\_height.  
  
**3.Player Class:** This class modifies the player's ship and inherits from Ship. It modifies move\_lasers to handle collisions with adversaries in particular and adds ways for controlling the health meter (healthbar). This is where the player loads and assigns their ship sprite and laser.

**4.Enemy Class:** This class builds upon Ship and specifies special actions for adversary ships, such as shooting (shot) and moving (move). It distinguishes between different opponent classes and their corresponding sprites and lasers using a color-coded mapping scheme.

**Interaction**: After initializing the game loop, the main function creates an interactive environment in which player inputs are processed, adversaries are spawned, and collisions are managed. To guarantee fluid gaming, the redraw\_window method refreshes the screen. Every element works together to create a smooth gaming experience.







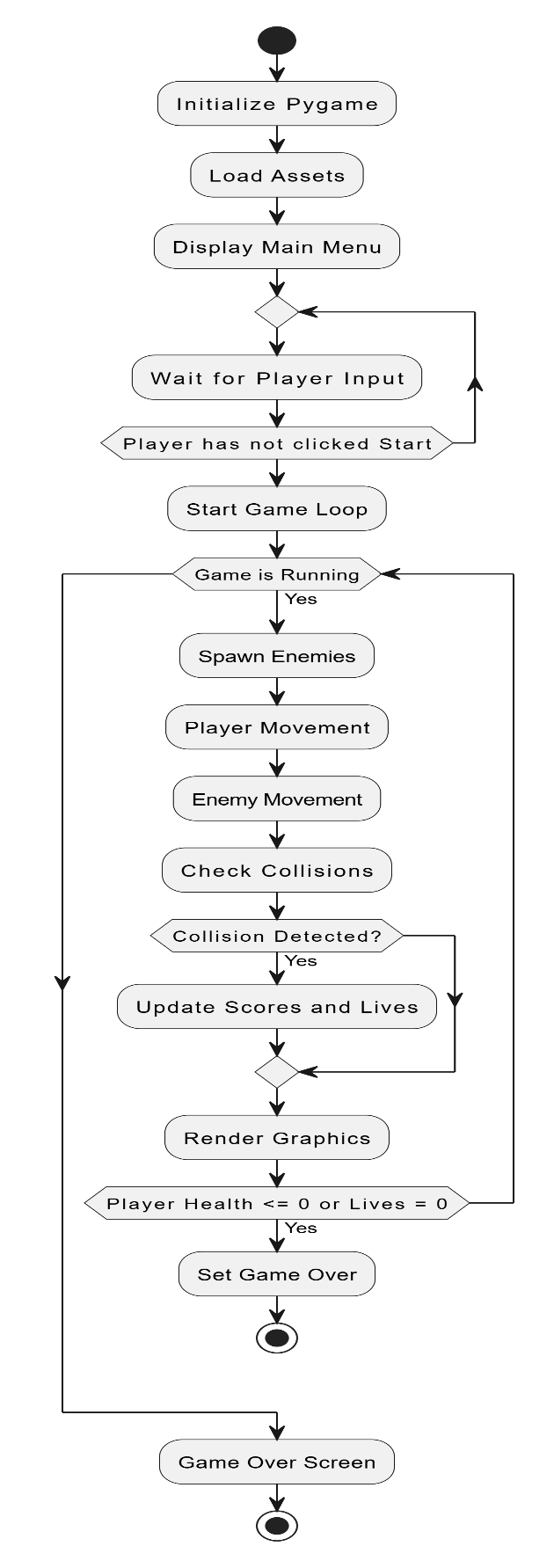


\





**5. Flowchart**

****