🧟‍♂️💥 ZombiChase: The Ultimate Survival Game! 💥🧟‍♀️

Welcome to ZombiChase, an exhilarating Rogue-like game developed using Python and Pygame. Prepare yourself for an adrenaline-pumping adventure where your main objective is to survive as long as possible against hordes of relentless zombies. Can you beat the odds and emerge as the ultimate survivor? 🏆

🎮 Gameplay 🎮

https://github.com/PaulZaman/ZombiChase/assets/64264952/f2d135a9-ff24-4e8b-a019-7c6518ba2a08

🌟 Game Features 🌟

🔫 Choose Your Weapon:

In ZombiChase, you have the freedom to select your preferred weapon to fend off the undead. Take your pick from three powerful firearms:

1️⃣ Rifle: Equip yourself with this long-range weapon to take down zombies from a distance. 2️⃣ Shotgun: If you prefer a more devastating impact at close range, the shotgun is your best bet. 3️⃣ Pistol: For those seeking a balance between range and firepower, the pistol offers reliable performance.

⚔️ Intense Gameplay:

Navigate through a dynamically generated, ever-changing world as you try to survive the relentless onslaught of zombies. The longer you survive, the higher your score climbs. Can you achieve the top spot on the leaderboards?

🎮 Easy Controls:

Move swiftly using either the arrow keys or the Z-Q-S-D keys on your keyboard. Take aim with your trusty mouse and unleash your weapon's firepower with a simple click or by pressing the spacebar.

Power-Ups:

🚀 Power-Ups: Prepare for the Ultimate Zombie-Slaying Arsenal! 🚀 As you progress through the game, you will encounter various power-ups that will give you the edge in surviving the zombie apocalypse. These power-ups include:

- 🩹 Health Packs: 💖 Restore your health and keep you in the fight for longer.
- 🎯 Precision Bullets: 🎯 Enhance your weapon's accuracy, ensuring your shots hit their mark with deadly precision.
- 💥 Bullet Damage: 💥 Amp up your weapon's bullet damage, turning the undead into mincemeat with each well-placed shot.
- 🔥 Fire Rate: 🔥 Increase your weapon's fire rate, unleashing a storm of bullets upon the zombie horde and leaving no chance for escape.

- 💥 Multi-Shot: 💥 Multiply your weapon's firepower by unleashing multiple bullets per shot, obliterating zombies in a single pull of the trigger.

🔮 Explore the game world, uncover these power-ups, and become an unstoppable force against the undead. May the power of these enhancements guide you to victory! 🔮

🗺️ Randomly Generated Maps:

Every playthrough of ZombiChase offers a unique and unpredictable experience. Brace yourself as the game generates completely random maps, ensuring no two games are ever the same.

🧟‍♀️🧟‍♂️💥 Endless Waves of Zombies:

The zombies in ZombiChase are relentless and will stop at nothing to devour you. As you progress through the game, the zombies will become faster and more aggressive. Can you survive the onslaught?

🏆 Leaderboards:

Compete against other players and see how your survival skills stack up against the rest of the world. Can you achieve the top spot on the leaderboards?

🔥 Installation Instructions 🔥

To embark on this epic survival adventure, follow these simple steps:

1️⃣ Set Up the Environnement:

To streamline the installation process, we have prepared all the dependencies in the requirements file. To install the dependencies, enter into your terminal at the main directory of the project

🚀 On macOS, Linux and Windows:

```
pip install -r requirements.txt
```

🏆 Database Setup:

To ensure the game's leaderboard functionality works as intended, you will need to connect to our database. To do so, you will need to rename the file "dbSetupEXAMPLE.py" to "dbSetup.py" and replace the placeholder cred_obj with the database credentials we have provided you with. Once you have done so, you will be able to connect to our database and view the leaderboard.

2️⃣ Launch the Game:

Once you have installed dependancies, run the game using the following command:

```
python3 main.py
```

🎉 Now it's time to immerse yourself in the heart-pounding action of ZombiChase and test your survival skills against waves of hungry zombies. May the odds be in your favor! 🎉

📝 Class Diagram and code organization 📝

In order to organize our code, we have seperated this whole programm into main classes:

- Menu
- Game
- Player
- Zombie
- Bullet
- PowerUp
- Map
- Room
- Tiles
- Window

We will describe each table, and how they interact with each other.

Table Description

Window

Attributes	Description
width (int)	the width of the window
height (int)	the height of the window
screen (pygame.Surface)	the game screen
w_tiles (float)	the number of tiles in the width of the window
h_tiles (float)	the number of tiles in the height of the window
clock (pygame.time.Clock)	the clock object for controlling the frame rate

Game

Attributes	Description
window (Window)	the game window object
screen (pygame.Surface)	the game screen
start_time_game (int)	the start time of the game
time_survived (int)	the time survived in milliseconds
difficulty (str)	the game difficulty level

Attributes	Description
back_to_menu (bool)	flag indicating if the game should return to the main menu
weapon_info (dict)	information about the player's weapon
sprites (pygame.sprite.Group)	group containing all game sprites
map (Map)	the game map object
player (Player)	the player object
zombies (pygame.sprite.Group)	group containing zombie sprites
shake_start_time (int)	the start time of screen shaking
shake_duration (int)	the duration of screen shaking in milliseconds
shake (int)	the magnitude of screen shaking

Menu

Attributes	Description
window	Represents the game window
screenIsOn	Represents the current screen being displayed in the game
fullscreen	Indicates whether the game is running in fullscreen mode or not
w_tiles	Represents the width of the game window in terms of tiles
h_tiles	Represents the height of the game window in terms of tiles
bg	Represents the background image of the game
game_difficulty	Represents the difficulty level of the game
game_weapon	Represents the selected weapon for the game
pistol_object	Represents the attributes of the pistol weapon
rifle_object	Represents the attributes of the rifle weapon
shotgun_object	Represents the attributes of the shotgun weapon
weapons	Represents a list of available weapons in the game
selected_weapon_index	Represents the index of the currently selected weapon
game	Represents the instance of the Game class that manages the game logic
run	Controls the main game loop

Map

Attributes	Description

Attributes	Description
game	Represents the instance of the Game class that manages the game logic
w	Represents the width of the map in terms of tiles
h	Represents the height of the map in terms of tiles
pos	Represents the current position of the map
powerups	Represents a group of powerup sprites in the map
n_powerups	Represents the number of powerups to be generated in the map
walls	Represents a group of wall tiles in the map
tiles	Represents a group of all tiles in the map
rooms	Represents a list of generated rooms in the map
grid	Represents a 2D grid that stores the state of tiles in the map
mini_map (commented out in the provided code)	Represents the mini-map surface for the map

Room

Attributes	Description
x	The x-coordinate of the room's top-left corner
y	The y-coordinate of the room's top-left corner
w	The width of the room
h	The height of the room
map	A reference to the map object the room belongs to
tilesize	The size of each tile in pixels
tiles	A pg.sprite.Group() object containing all the tiles in the room
walls	A pg.sprite.Group() object containing all the wall tiles in the room

Tile

Attributes	Description
type	Represents the type of the tile
pos	A pg.math.Vector2 object representing the position of the tile in (x, y) coordinates
map	Represents the map the tile belongs to

Attributes	Description
x	The x-coordinate of the tile divided by the tile size
y	The y-coordinate of the tile divided by the tile size
images_dir	A string representing the directory where the tile images are located
image	Represents the image of the tile
rect	A pg.Rect object representing the rectangular area occupied by the tile in the game

Player

Attributes	Description
pos	A vector representing the player's position
vel	A vector representing the player's velocity
acc	A vector representing the player's acceleration
speed	A float representing the player's movement speed
friction	A float representing the friction applied to the player's movement
weaponName	A string representing the name of the player's weapon
life	An integer representing the player's remaining life
walls	A sprite group representing the walls in the game
game	An instance of the game class
score	An integer representing the player's score
bullets_shot	An integer representing the number of bullets shot by the player
bullets_missed	An integer representing the number of bullets missed by the player
zombies_killed	An integer representing the number of zombies killed by the player
powerups_collected	An integer representing the number of power-ups collected by the player
idle_frame_index	An integer representing the current frame index for the idle animation
idle_animation_delay	An integer representing the delay between each frame in the idle animation
idle_last_update	An integer representing the time of the last update for the idle animation
image	The current image of the player
rect	The rectangle that defines the player's position and size
collision_rect	A rectangle used for collision detection
weapon	An instance of the Weapon class representing the player's weapon

Bullet

Attributes	Description
image	A Surface object representing the bullet's image
rect	A Rect object defining the position and size of the bullet
zombie_group	A sprite group containing the zombies in the game
walls	A sprite group containing the walls in the game
direction	A vector representing the direction of the bullet's movement
player	An instance of the Player class representing the player
speed	A float representing the speed at which the bullet moves

Weapon

Attributes	Description
player	An instance of the Player class representing the player
bullets	A sprite group containing the bullets fired by the weapon
bullet_image	A Surface object representing the image of the bullet
gun_offset	A vector representing the offset position of the gun relative to the player's position
weaponName	A string representing the name of the weapon
damage	An integer representing the damage inflicted by each bullet
fire_rate	An integer representing the time delay between consecutive shots in milliseconds
bullet_speed	A float representing the speed at which the bullets travel
n_bullets	An integer representing the number of bullets fired per shot
precision	An integer representing the precision or spread of the bullets
last_shot	An integer representing the time in milliseconds when the weapon was last fired

Powerup

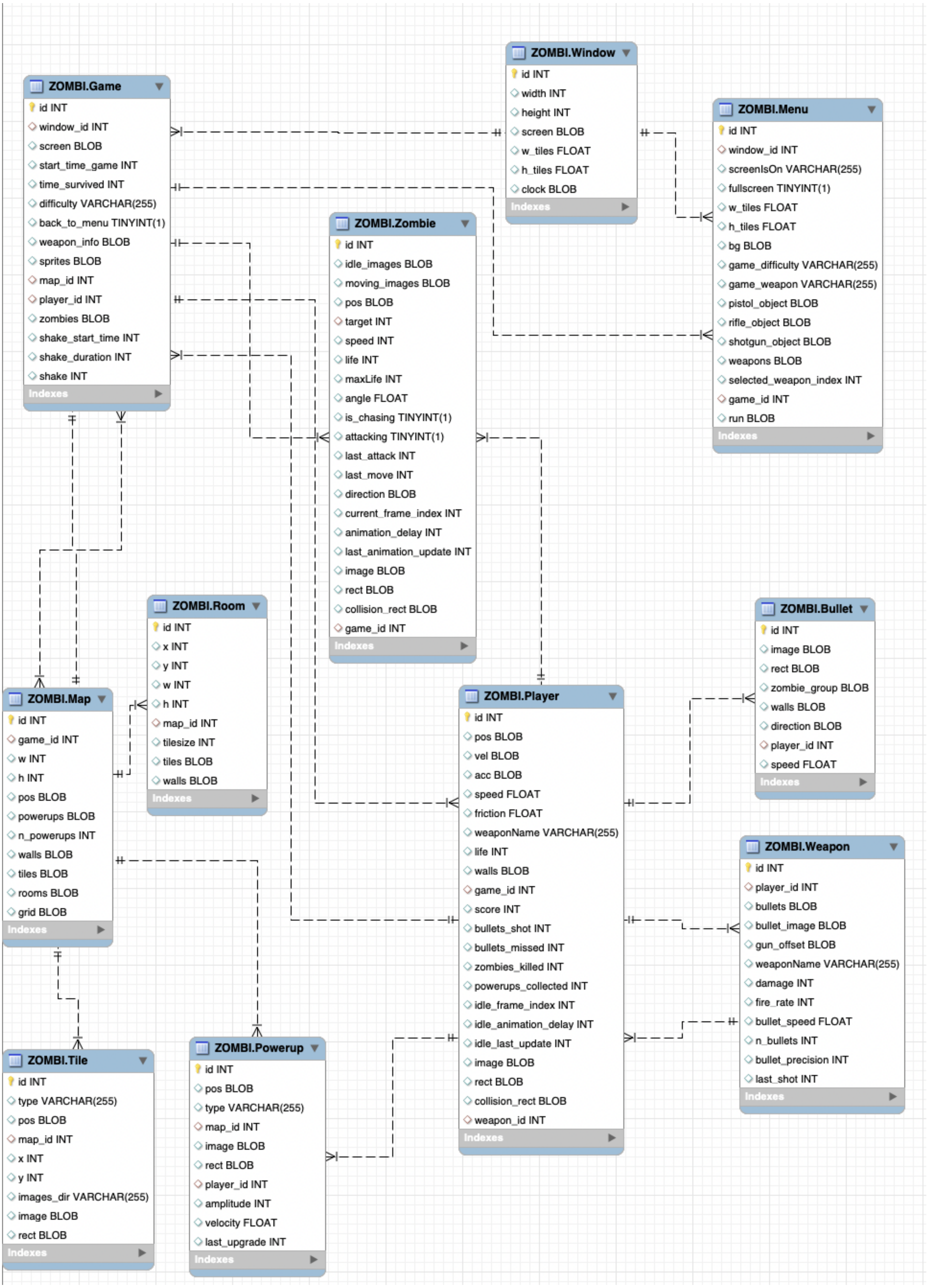
Attributes	Description
pos	A vector representing the position of the power-up
type	A string indicating the type of power-up
map	An instance of the Map class representing the game map
image	A Surface object representing the image of the power-up
rect	A rectangle representing the bounding box of the power-up image
player	An instance of the Player class representing the player

Attributes	Description
amplitude	An integer representing the height of the bounce animation
velocity	A float representing the speed of the bounce animation
last_upgrade	An integer representing the time in milliseconds when the power-up was last applied

Zombie

Attributes	Description
idle_images	A list of idle animation images for the zombie
moving_images	A list of moving animation images for the zombie
pos	A vector representing the position of the zombie
target	A reference to the player object that the zombie chases
speed	An integer representing the speed of the zombie
life	An integer representing the current life of the zombie
maxLife	An integer representing the maximum life of the zombie
angle	A float representing the angle at which the zombie is facing
is_chasing	A boolean indicating whether the zombie is currently chasing the player
attacking	A boolean indicating whether the zombie is currently attacking the player
last_attack	An integer representing the time in milliseconds when the zombie last attacked
last_move	An integer representing the time in milliseconds when the zombie last moved
direction	A vector representing the direction in which the zombie is moving
current_frame_index	An integer representing the current frame index for the animation
animation_delay	An integer representing the delay between each frame in the animation
last_animation_update	An integer representing the time of the last update for the animation
image	The current image of the zombie
rect	The rectangle that defines the zombie's position and size
collision_rect	A rectangle used for collision detection
game	An instance of the Game class

Class diagram



🚀 Main Execution 🚀

Let's go through the main execution of the program:

1️⃣ **The main.py file is executed, which will create an empty Window object. The file then creates a Menu object by mounting it on the Window.**

2️⃣ **The Menu object will then be executed, which navigates through the different menus and will create a Game object when the user clicks on the "Play" button.**

3️⃣ **The Game object will be initialized, creating the following objects: Player 🕹️ , Map 🗺️ , Zombie 🧟 , Bullet 🔫 , PowerUp ⚡ , Tiles 🧱 , and Rooms 🏰 . It will then start the game loop, updating and drawing the game on the screen.**

There are 3 main parts to our game loop:

✨ Events: This part handles all the events happening in the game, such as player movement, shooting, and power-up collection.

✨ Update: This part updates the game, including moving the zombies, bullets, etc.

✨ Draw: This part draws the game on the screen, displaying the player, zombies, bullets, etc.

4️⃣ **When the player dies, the Game object will be destroyed, sending the player back to the menu.**

🌟 Randomly Generated Maps 🌟

To generate a random map, when the Game object is created, we create a Map object. It generates a 2D array of Tiles objects. Each Tile object has a type, used to draw the map accordingly. Types can be "grass" 🌿 , "wall-br" 🧱 , "wall-bl" 🧱 , "wall-tr" 🧱 , "wall-tl" 🧱 , "wall-h" 🧱 , "wall-v" 🧱 , "inside" 🚪 , "grass-5" 🌿 .

The Map object also creates a random number of Rooms objects to generate the rooms. It then places the Player object in the top left of the map. Additionally, it randomly places a number of Zombie objects and PowerUp objects (within the rooms) on the map.

📝 Database 📝

To store player highscores, we've created a database using Firebase. The database is hosted online, allowing players to access it. The "scores" table in the database contains the following columns:

Name 👤 : Player's name Rank 🏆 : Player's rank Score 🎮 : Player's score Duration ⏳ : Game duration Zombies Killed 💀 : Number of zombies killed Difficulty 🌟 : Game difficulty Bullets Shot 🔫 : Number of bullets shot Weapon Info 🗡️ : Player's weapon information All this information is accessible on the highscores page in the menu.

📝 Dependencies and Technologies Used 📝

We used Python 🐍 to create this game, along with the pygame library 🎮 for game development. The pyrebase library 🔥 was utilized to connect to the Firebase database. We also made use of the random library 🎲 for generating random numbers and the math library ➕ 📐 for calculations.

Each element on the screen, such as tiles, players, zombies, and power-ups, is represented as a pygame.sprite. This allows easy movement of elements and collision detection between them.

Conclusion

ZombiChase is an exhilarating Rogue-like survival game developed using Python and Pygame. The objective of the game is to survive as long as possible against hordes of relentless zombies. The game offers a variety of exciting features, including the ability to choose from three powerful weapons (rifle, shotgun, pistol), intense gameplay with dynamically generated maps, easy controls, and a range of power-ups to enhance your zombie-slaying abilities. 💥 🧟 🔫

The game's randomly generated maps ensure a unique and unpredictable experience with each playthrough. As you progress, the zombies become faster and more aggressive, challenging your survival skills. Compete against other players worldwide and strive for the top spot on the leaderboards. 🗺️ 🌍 🏆

To play ZombiChase, simply follow the installation instructions provided in the project. Once you launch the game, immerse yourself in the heart-pounding action, aim for high scores, and test your survival skills against endless waves of hungry zombies. 🎮 🔥 🧟

The project's code is organized into several main classes, including Menu, Game, Player, Zombie, Bullet, PowerUp, Map, Room, Tiles, and Window. Each class represents a specific aspect of the game and interacts with others to create the overall gameplay experience. 👨‍💻 🧟 🧱

ZombiChase offers an immersive and challenging gaming experience for players who enjoy action-packed survival games. So, gear up, choose your weapon, and get ready to face the zombie apocalypse like never before! 🧟 ⚔️ 🔥

Enjoy playing the game! Have fun! 🎉 🕹️