🎮 *End of The World* – Game Documentation

Project Type: Intro To AI 12th Project – Game using Python & Pygame

Student Name: Zeyad Mohamed Ahmed Abdelwahab
University: Arab Academy for Science, Technology & Maritime Transport (AAST)

Faculty: Computer Science

Project Name: End of The World

⭕ Project Summary

The project is a top-down shooter game where you play as the **last human on Earth**, fighting off waves of enemies (zombies). The game was created using **Python and Pygame**, and the player uses the keyboard to move and the mouse to aim and shoot.

The game includes bullets, sounds, animated enemies, shooting mechanics, and real collision between bullets and enemies or the player and walls. A gun is attached to the player that rotates towards the mouse. When you press the mouse, a bullet shoots in the same direction and hits enemies.

Hierarchy:

EndOfTheWorld/

```
│
├── main.py              # The main file to run the game
├── player.py            # Contains the Player class and its movement
├── sprites.py           # Contains all types of objects (Gun, Bullet, Enemy, Collision)
├── settings.py          # General game settings (such as window size)
├── groups.py            # Sprite group management
├── data/
│   └── maps/world.tmx   # The game map designed using Tiled
├── images/
│   ├── enemies/         # Enemy images (animated)
│   └── gun/             # Images of guns and bullets
├── audio/               # Sound effects and music for the game
```

🧠 Core Classes & Functions

Game (in main.py)
Function: Manages the game, initialization, events, and updates.

- `__init__()` – Sets up the window, music, loads images, and initializes elements.
- `load_images()` – Loads images for bullets and enemies.
- `setup()` – Loads the map and places the player, enemies, and other elements.
- `run()` – Main game loop.
- `input()` – Captures input for shooting action.
- `gun_timer()` – Sets the wait time between shots.
- `bullet_collision()` – Handles collisions between bullets and enemies.
- `player_collision()` – Ends the game if the player collides with a forbidden object.

Player (in player.py)
Function: Represents the player and controls movement and direction.

- Receives direction from mouse movement (in `Gun.get_direction`).
- Always drawn at the center of the screen (Camera centers on the player).

Gun (in sprites.py)
Function: Rotates based on mouse direction and fires bullets.

- `get_direction()` – Calculates the direction between the player and the mouse.
- `rotate_gun()` – Rotates the gun image based on the direction.
- `update()` – Combines movement, rotation, and location.

Bullet (in sprites.py)
Function: Represents bullets fired from the gun.

- Moves in a specific direction at a defined speed.
- Deleted after a specified time or upon collision.

Enemy (in sprites.py)
Function: Represents enemies moving toward the player.

- `move()` – Moves toward the player and checks for collisions.
- `collision()` – Handles collision with walls and obstacles.
- `destroy()` – Starts the death effect and changes the enemy's appearance.
- `death_timer()` – Deletes the enemy after a set time following its death.

🧩 Additional Components

CollisionSprite
Represents objects that the player or enemies collide with (like walls or obstacles).

AllSprites (from groups.py)
A custom Sprite Group to draw everything relative to the player's center (camera effect).

📢 Audio

- `shoot.wav` – Sound of shooting.
- `impact.ogg` – Sound of bullet hitting an enemy.
- `music.wav` – Background music.

🗺️ Map (world.tmx)
The map is designed using Tiled and contains 4 layers:

- `Ground` – The background.
- `Objects` – Visible objects that are collidable.
- `Collisions` – Invisible elements that represent collision boundaries.
- `Entities` – Defines where the player and enemies are placed.

🛠️ Features

Feature	Description
Shooting towards the mouse	The gun rotates and follows the mouse cursor.
Smart collision	With enemies and walls.
Enemy wave system	Enemies spawn automatically at intervals.
Smooth movement with Vector2	Uses vectors for all movements.
Camera follows the player	Everything moves based on the player's position.
Sound effects	For every event (shooting, impact, music).