Bot Brawl 3D - Project Description

Project Overview: Bot Brawl 3D

Technologies Used

- Python
- PyOpenGL (OpenGL, GLUT, GLU)
- GLUT for rendering UI and capturing input

Core Components

1. Game Modes

- Menu: Startup screen with options.
- Single Player: Fight off waves of enemies.
- Multiplayer: Two players fight each other locally.
- Game Over: Displayed when all players die.

2. Player Mechanics

- Two players supported.
- Movement: WASD / Arrow keys
- Jumping: W / Arrow Up
- Weapon Attacks: SPACE or Ctrl or Mouse
- Weapon Types:
  - Sword (melee swing attack)
  - Gun (shoot bullets)
  - Grenade (area explosion)

3. Enemy AI

- Randomized movement with homing behavior.
- Basic platform collision and jump logic.
- Attacks player when close.
- Animated wings and blinking eyes.

4. Weapons & Projectiles

- Weapons randomly spawn on platforms.
- Each has a limited lifetime.
- Players can pick up only one weapon at a time.
- Projectiles:
  - Bullets (straight-line fast projectiles)
  - Grenades (physics-based lobbed projectiles that explode)

5. Explosions

- Caused by grenades.
- Damage nearby enemies and players.
- Expands visually with transparency for a fiery effect.

6. Platforms & Arena

- Fixed arena with predefined platforms.
- Ground and elevated stages for navigation.
- Collision detection for all entities.

7. Camera

- Dynamic third-person view centered on players.
- Smoothly follows average player position.

Visuals and UI

- 3D models built from primitives (cubes, spheres, quads).
- Background: Solid blue to simulate a sky.
- In-game HUD:
  - Health and weapon info.
  - Score and time survived.
  - Controls help text.
- Game Over Screen:
  - Displays winner or score.
  - Options to restart or quit.

Controls

Action	Player 1	Player 2 (Multiplayer)
Move Left/Right	A / D	Left / Right Arrows
Jump	W	Up Arrow
Attack (use weapon)	Space / Mouse	Down Arrow
Restart Game	R	R
Return to Menu	Esc	Esc

Game Logic Highlights

- All game logic is encapsulated within the Game class.
- Enemies and weapons spawn at timed intervals.
- Sword swings check for angle before registering hits.
- Grenade explosions calculate radial damage.
- Uses OpenGL blending for transparency effects (explosions, wings).

Notable Features

- Multiplayer support with two distinct player models/colors.
- Smooth camera tracking and physics-based movements.
- Weapon handling with time-based expiration.
- Full visual feedback for health (red flash) and explosions.