**CSCI 438 – ADVANCED GAME DEVELOPMENT**

**FINAL PROJECT**

**Matthew Cobbs & Ahmet Y. Cengiz**

**Game Overview:**

The game is inspired by the mobile arcade action game by Habby, “Archero”. Our hero faces different number of enemies every level, and he must destroy them to go to the next level and to finally face the boss enemy.

**Game Flow Summary:**

The game starts with a title screen. After the user progresses past this screen the first level loads to start the battle.

The user progresses through the levels by collecting the coin which appears in a random location after every enemy is defeated.

**Gameplay and Mechanics:**

The user controls their character’s movement with the arrow keys to dodge the bullets from its enemies. The character can only fire back when he is not on the move. Whenever he stops, the closest enemy is targeted and shooting starts. Thus, the user act strategically and choose wisely when to stop since this is the only way to change the course of the fight. The health bars on top of the hero and the enemies help to see their current HP(health point).

**Game Implementation:**

The game is implemented in Godot game engine with the help of Blender.

The hero is a kinematic body with manually applied gravity as a downwards force vector. The movement is based on the camera angle (*Figure 1*). The function “findClosestEnemy” (*Figure 2*) as anyone can imagine, locates the closest enemy. Then, the “aim” function *(Figure 3)* comes into play to point the hero to the closest enemy by using quaternions. When the aiming is done, “shoot” function (*Figure 4*) instances a bullet from a preloaded HeroBullet scene and sets its location to the gunContainer’s global location which is of type Position3D. This chain of events occurs at the signal “timer\_time\_out” which happens every one second when the character is not moving (*Figure 5*).

The enemy is also a kinematic body with manually applied gravity as a downwards force vector. They use the same aiming technique as the hero to shoot at the hero. However, its timer is triggered every two seconds instead of one.

Hero bullets and enemy bullets are both of type Area and they use signals to check if they hit something. Then they trigger the hit body to change its health and according to that their health bars.

Health bars are of type Sprite3D with a viewport containing a TextureProgress HealthBar2D. HealthBar2D contains the implementation of the health bar visuals (*Figure 6*). We activated the billboard mode to have the bar always angled towards the camera.

Levels resources are from the free Nature resource package available at <https://kenney.nl/assets/nature-kit>. Using those meshes I was able to construct a 3D grid map in Godot to help build levels. During the construction of the 3D grid map, all meshes that were imported were given static bodies and collision shapes.

When I was designing the levels, I broke the structure down into two layers. The first ‘base’ layer is a generic layer that contains an empty game area with no obstacles. All of the features of the game map that do not need to by dynamic are built in this layer. The second layer is built of numerous nodes that are all children of the base node. These children nodes are what the actual level map is composed of. Each individual node has the required obstacles for a given level and does not require any of it’s sibling nodes for a level to be playable.

Once the sub levels were constructed, I made them invisible and turned off collisions. I then implemented an algorithm in Godot that allowed the base node to randomly select one child node and make only that child visible, as well as restoring that child’s collision ability. This allowed the game levels to have a different sequence of levels each time that it is played.

**Figure 1**

A screenshot of a cell phone

Description automatically generated

**Figure 2**

A screenshot of a cell phone

Description automatically generated

**Figure 3**

A screenshot of a cell phone

Description automatically generated

**Figure 4**

A picture containing bird

Description automatically generated

**Figure 5**

A screenshot of a cell phone

Description automatically generated

**Figure 6**

A screenshot of a social media post

Description automatically generated