Requirements and Organization Document

AdventureFVTC

# Game Mechanics

The game is an adventure game. The player travels through the level collecting items. The player has a camera associated with it. Enemies attack and are attacked by the player. Enemies and Players are Units.

## Unit Mechanics

Units have health. Units have a type. Units can move. Units have animations. Units have a facing direction. When a unit has 0 health it dies. Units make sounds. Units collide with objects including other units.

## Player Mechanics

The player is a unit. The player should be able to pick up items. The player can interact with the environment. The player has an inventory. The player’s inventory signals if they have the unique items dropped in each level.

## Enemy Mechanics

An enemy is a unit. Enemies should attack the player when agitated. Enemies will use simple AI to track and attack the player. Enemies should move around even when not encountering the player. Enemies can drop items.

## Camera Mechanics

The camera should follow its player. The camera should be behind the player or moving along the terrain to give the player a sense of movement. The camera should always be looking towards the player. The camera should be slightly less than 45 degrees in relation to the player. The camera may focus on objects/places other than the player temporarily.

## Inventory Mechanics

Inventory is persistent through the levels and is used to check if the player can progress to the next level.

## Item Mechanics

An item stays in place, slightly above the ground and spins to let the player know they can pick it up. A player cannot collide with an item, it does not slow the player down when they walk into it. Once a player walk into an item they pick it up and it’s added to their inventory, and then the item is removed from the world.

## Audio Mechanics

Each level has its own sound track. These soundtracks can change based on the levels current cycle or how many times the player has seen this level before.

## Level Mechanics

A level has UI. A level plays music. A level has a night and day cycle. A level has a player, enemies, and items.

## Level Change Mechanics

The player starts in a room that has doors that allows a player to advance to the next level. When the player first enters the room they only have access to the first door. After the player completes the first level and returns to the door room, the second door will be unlocked and the player can enter it. As the player completes levels, new doors will unlock.

## Day and Night Mechanics

A level will have a duration for day and night. When the day cycle begins, music related to its day cycle will start to play. After the day duration has passed night will begin. Once night begins, aggressive enemies will spawn and music related to its night cycle will start to play. After a certain amount of time, the night cycle will end and its music will stop. Shortly before night ends, the spawned enemies will retreat back to their homes. After the night cycle ends the day cycle starts over again.

### AI Types

Any AI may use one of various types of AI. A defensive AI ignores the player and other enemies until it is agitated by being attacked. Once agitated, a defensive AI will attack and chase the player/enemy until the player/enemy gets a certain distance from the defensive AI and then the defensive AI will then return to its last location before it was attacked. An aggressive AI will seek out players or units of types other than itself to attack. Once an aggressive AI comes into range of a player or unit of another type it will attack that player/unit. Once the player/unit gets out of range of the aggressive AI, the aggressive AI will stop chasing and search for a new target. A boss AI will attack the player until the player or itself dies. The boss will have a few different attacks that it uses in a pattern to attack the player.

### Organization Layout

Script Types: Cameras, Items, Units, Players, Services

Material Types: Textures, Models, Physics, UI

Prefabs: camera, character, player