Software Design Description

Graze Save Survive

Version 0.9

2 February 2024

https://github.com/holiday-pettijohn/GrazeSaveSurvive (currently private)

Department of Math and Computer Science, Biola University

Overview. This document exists to demonstrate how the produced software will be structured in order to satisfy the requirements.

Target Audience. Development Team

Project Team Members. Holiday Pettijohn, Jacob Shirota, Caleb Zuniga, Nicklas Kuo

Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
0.1	Jacob Shirota	Initial documentation	01-30-2024
0.2	Holiday Pettijohn, Jacob Shirota, Nicklas Kuo	Description of Components	02-01-2024
0.3	Jacob Shirota	Application States, Entity Class Diagram	02-02-2024

Signatures of Approval

Version 1.0		
Name	Signature	Date
Holiday Pettijohn		
Jacob Shirota		
Caleb Zuniga		
Nicklas Kuo	Nicklas Kuo	02-02-2024

Table of Contents

1. Introduction

1.1. Design Overview

The software design is primarily split between User Interface, Application, and Database components, with a focus on Application components. Menus are expressed as a series of states which the software may be in. A class hierarchy for Entities, which encapsulates any moving object represented by a sprite during gameplay, is established.

1.2. Requirements Traceability Matrix

A separate Requirements Traceability Matrix has been designed and shared with all group members.

2. System Architectural Design

2.1. Chosen System Architecture

2.1.1. The program will have respective versions that can run on Windows and Linux

2.2. Discussion of Alternative Design

2.2.1. We discussed different game genre, mechanics, and UI options and found that the current design is a workable balance of feasible-to-implement goals. We considered some systems which would require more visual assets by making more sprites for items but decided on the current system due to time constraints.

2.3. System Interface Design

2.3.1. Window

2.3.1.1. The software will open in a 400px x 400px window.

2.3.2. Sprite Files

2.3.2.1. Sprite sheets will be stored in the install directory.

2.3.3. Audio Files

2.3.3.1. Audio files will be stored in the install directory.

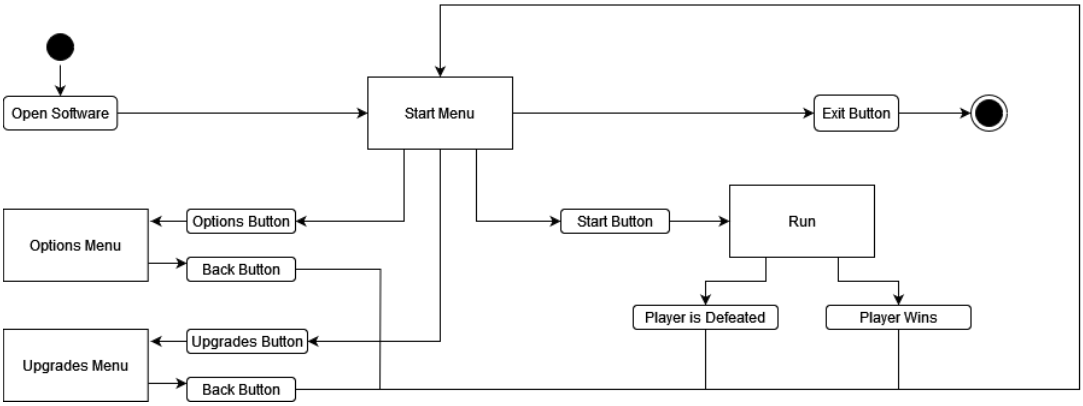
2.3.4. sqlite Database

2.3.4.1. The database which stores player progress will be stored in the install directory.

3. Detailed Description of Components

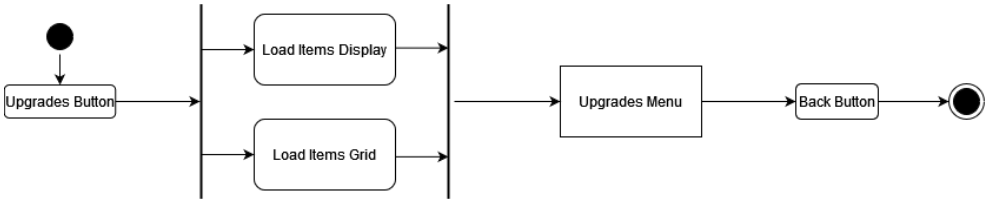
3.1. Application States

3.1.1. Start Menu



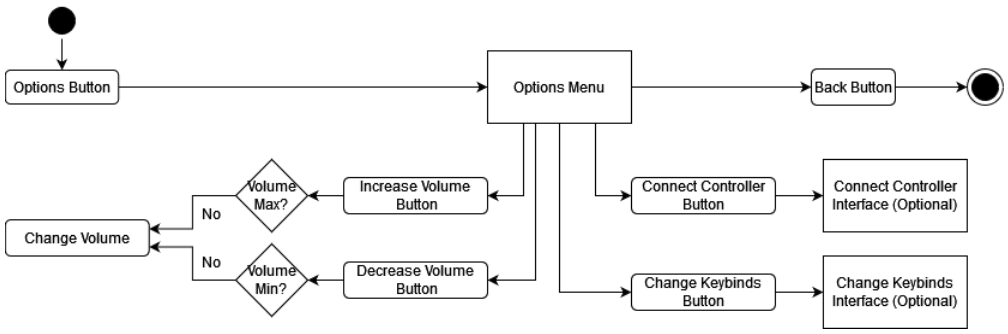
When the software is initially opened, the first state it enters is the Start Menu. This State can be exited depending on which button the User interacts with. This State can be returned to from the Options Menu, Upgrades Menu, and Run States.

3.1.2. Upgrades Menu



The Upgrades Menu State is entered when the User interacts with the Upgrades Button from the Start Menu. When this State is entered, the Application must load the Items Display and Items Grid (see §3.4). This State can be exited when the User interacts with the Back button.

3.1.3. Options Menu

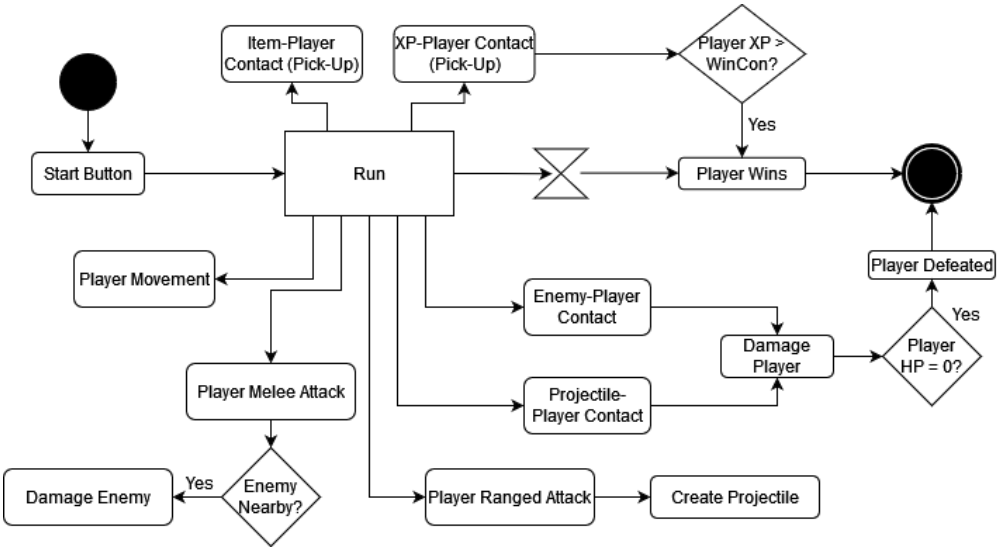


The Options Menu State is entered when the User interacts with the Options Button from the Start Menu. This State can be exited when the User interacts with the Back Button.

From this State, if the User interacts with the Increase Volume Button, if the volume is not at its maximum limit, the volume will be increased. If the User interacts with the Decrease Volume Button, if the volume is not at

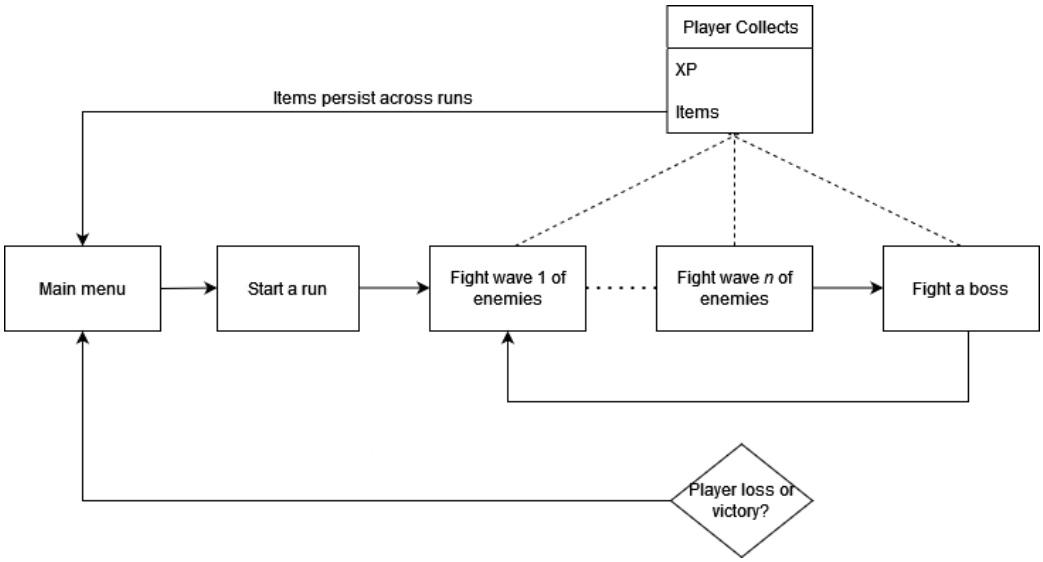
its minimum limit, the volume will be decreased. If the User interacts with the Change Keybinds button, an interface will be displayed from which they can change their keybinds. If the User interacts with the Connect Controller button, an interface will be displayed from which they can connect a controller.

3.1.4. Run

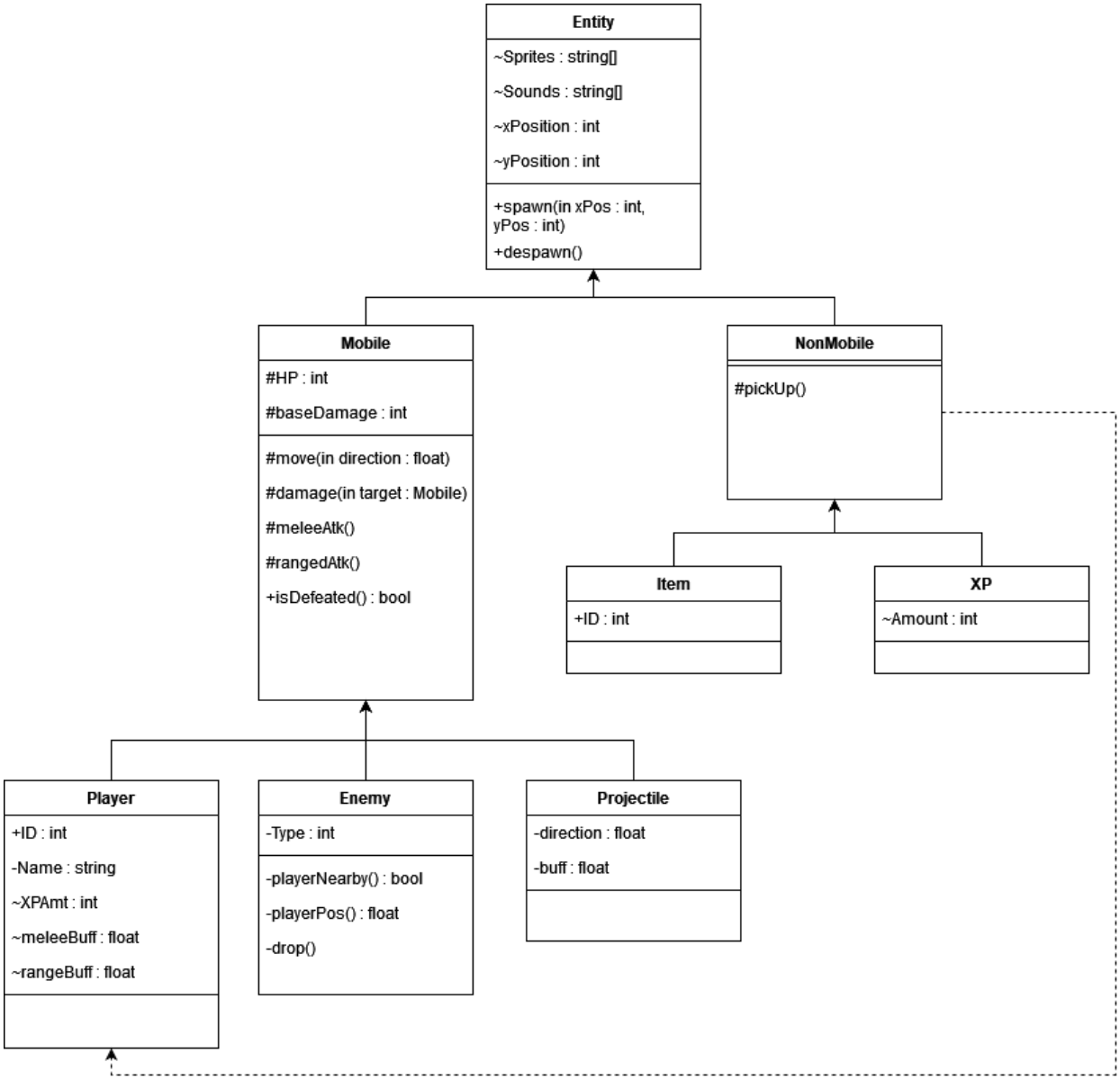


The Run State is entered when the User interacts with the Start button from the Start Menu. This State is exited when the Player “Wins” or is “Defeated”.
From this State, the User can Move, perform a Melee Attack, or perform a Ranged Attack using different keys.
If the Player comes in contact with an Enemy or a Projectile, they are dealt Damage. If, after being dealt Damage, the Player’s HP is 0, the Player is “Defeated”.
If the Player comes in contact with an Item or XP, the Player “picks up” that Entity. If, after picking up XP, the Player’s XP is above the win-condition threshold, the Player “Wins”.
After a fixed amount of time passes, if the Player is still in the Run state, the Player “Wins”.

Diagram of the general gameplay loop:


Player Collects
XP
Items
Items persist across runs
Main menu
Start a run
Fight wave 1 of enemies
Fight wave n of enemies
Fight a boss
Player loss or victory?

3.2. Entities



Entities are objects present during a Run which are represented by a visible sprite and are not part of the graphical background. They can be interacted with by other Entities.

3.2.1. Entities are created with an array of file locations for any Sprites and Sound files associated with them.

3.2.2. Entities are created with an initial x- and y-position where their sprite is located on the screen, relative to the center.

```
This function creates an Entity and places their Sprite at
a specified location on the screen.
Class Entity:
constructor spawn(xPos, yPos, sprite):
```

```
Set current_sprite to Sprites[default sprite]
Set xPosition to xPos
Set yPosition to yPos
```

3.2.3. *Mobile* Entities – Entities which are capable of moving around the screen and can cause Damage.

3.2.3.1. Mobile Entities are spawned with a set amount of HP and a base Damage value.

3.2.3.2. Mobile Entities can move in a specified direction.

3.2.3.3. Mobile Entities can deal Damage to another Mobile Entity when the hitbox of one of their attacks overlaps with the hurtbox of the recipient.

```
This is a class for attacks attached to hitboxes,
which will be passed to the receiving entity. They
contain the properties of the attack should it land.

Class Attack:

Int BaseDamage
Array StatusEffects

This function reduces an inputted Mobile's HP.

Class MobileEntity:

function damage(attack)
self.hp = self.hp - (BaseDamage-self.defense)
```

3.2.3.4. Mobile Entities can perform a melee Attack.

```
This function allows a Mobile to perform a melee
Attack. This shows the appropriate Sprites and deals
Damage to any other Entities within a fixed radius.

function meleeAtk():
FOR EACH attack sprite in Sprites do
  Set current sprite to attack sprite
ENDFOR
Set current sprite to default sprite
FOR EACH enemy do
  IF playerNearby do
    damage(enemy)
  ENDIF
ENDFOR
```

3.2.3.5. Mobile Entities can perform a ranged Attack.

```
This function allows a Mobile to perform a ranged
Attack. This shows the appropriate Sprites and spawns
```

```
a Projectile.

function rangedAtk():
FOR EACH attack sprite in Sprites do
  Set current sprite to attack sprite
ENDFOR
Set current sprite to default sprite
Spawn(0, 0) projectile
Set projectile direction to mouse position
Move(direction) projectile
```

3.2.3.6. Mobile Entities can be ‘Defeated’ when their HP is reduced to 0, at which point they despawn.

```
This function checks if a Mobile is Defeated.

function isDefeated():
IF HP is less than or equal to 0 do
  return True
ELSE
  return False
ENDIF
```

3.2.3.7. *Player* – The Entity controlled by the User.

3.2.3.7.1. The Player spawns into a Run when the Run begins. It is always visible during a Run.

3.2.3.7.2. The Player is spawned in with a set HP value based on the Upgrades Menu.

3.2.3.7.3. The Player accumulates XP. After accumulating certain thresholds, the Player’s meleeBuff and/or rangeBuff will be increased. This is shown in the table below.

XPAmt	meleeBuff or rangeBuff	Increase
100	meleeBuff	*1.2
200	rangeBuff	*1.2
300	meleeBuff	*1.5
400	rangeBuff	*1.5
500	meleeBuff and rangeBuff	*2.0
Every 250 after	meleeBuff and rangeBuff	*2.0

3.2.3.7.4. When a Player is Defeated, the Run ends (see §3.1.4). This is checked every time the Player receives Damage from another Mobile Entity.

3.2.3.8. *Enemies* – Entities controlled by the computer which oppose the player.

3.2.3.8.1. Enemies are spawned in with a Type, which determines their HP, base Damage, amount of XP dropped, and chance to drop an Item. This is shown in the table below.

Type	HP	baseDamage	Base XP Amount	XP Range (+/-)	Item Chance
1	5	2	1	0	0
2	10	5	3	1	0.05
3	25	15	5	2	0.05
4 (Boss)	300	80	75	0	0.25
5	30	25	10	3	0.1
6	40	35	15	3	0.1
7	50	45	20	4	0.2
8 (Boss)	500	150	250	0	1.00

3.2.3.8.2. Enemies are spawned in waves offscreen at the left and right of the player. The Type and amount spawned at once is based on the amount of time a Player has been in a Run and the number of remaining Enemies. A sample implementation of this (subject to change after testing) is shown in the table below.

Minimum Time	Max Enemies Remaining	Type	Amount Spawned
0:10	0	1	4
0:30	4	1	6
0:50	4	2	4
1:05	6	2	4
1:30	2	2	8

2:00	8	1	6
2:10	4	3	4
2:40	4	2	6
3:30	6	3	4
5:00	0	4	1
10:00	Any	Exit Run via Player Wins	

3.2.3.8.3. When an Enemy is Defeated, they may drop an Item at the location they were Defeated at.

```
This function spawns NonMobile Entities after
an Enemy is Defeated.

function drop():
Spawn(xPosition, yPosition) XP
Set XP Amount to
  base XP Amount [Type] +
  ( random(-1,1) *
    random(0, XP Range) )
IF random(0,1) is less than or equal to
itemChance do
  Spawn (xPosition, yPosition) Item
```

3.2.3.9. *Projectiles* – Entities spawned by a ranged Attack performed either by a Player or an Enemy.

3.2.3.9.1. Projectiles move in a single fixed direction, which is determined when they spawn.

3.2.4. *NonMobile* Entities – Entities which the Player can interact with and do not cause Damage.

3.2.4.1. NonMobile Entities spawn in the location where an Enemy was Defeated.

3.2.4.2. The rate at which Defeated Enemies drop certain NonMobile Entities is governed by a combination of chance and the type of Enemy which died (See §3.2.1.8.1).

3.2.4.3. NonMobile Entities can be ‘picked up’ when the Player comes within a certain distance of them, at which point they despawn.

3.2.4.4. *Items* – NonMobile Entities which persist outside of Runs and can be interacted with in the Upgrades Menu.

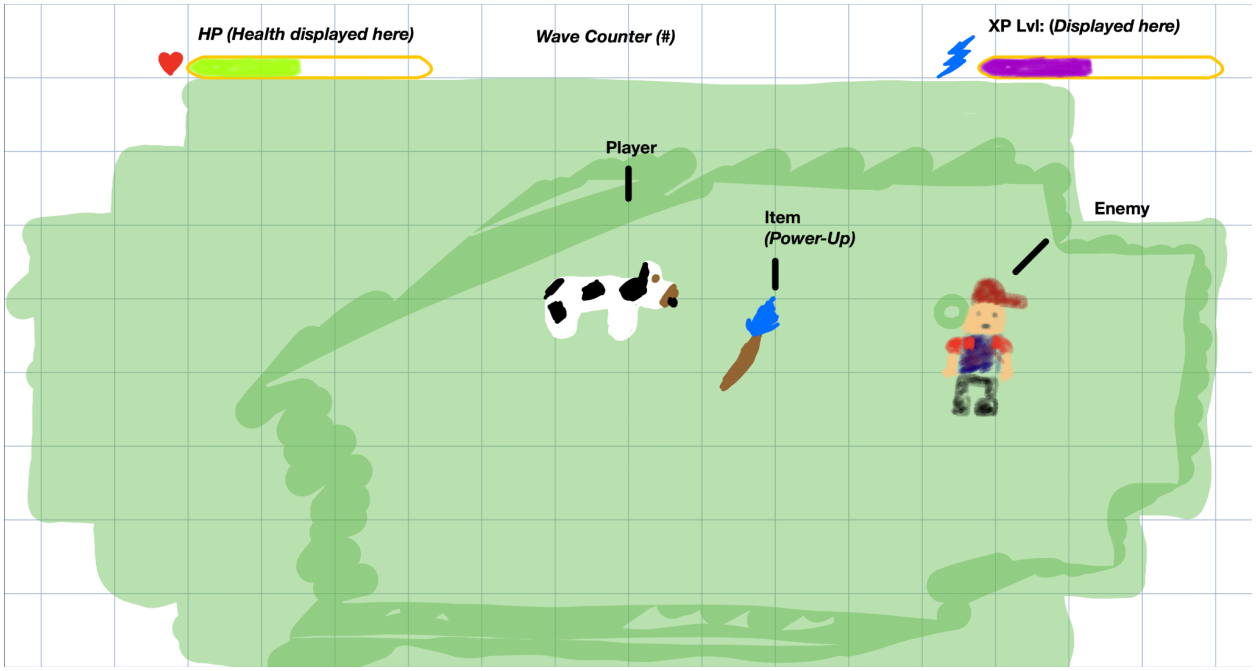
3.2.4.4.1. Picked up Items are stored in the database.

3.2.4.5. *XP* – NonMobile Entities which increase the Player’s XPAmt and do not persist after a Run.

3.2.4.5.1. XP are spawned in with a set Amount by which the Player's XPAmt is increased when they are picked up.

3.2.4.5.2. When the Player gains a sufficient amount of XP, they receive certain 'buffs' (see §3.3.2).

3.2.5. Entity Object Diagram:



3.3. In-Run Mechanics

3.3.1. Player Input

3.3.1.1. Movement

3.3.1.1.1. The player is controlled using the movement keys on the keyboard (default WASD), or the controller's joystick. For the keyboard, pressing multiple keys at once (e.g. left and up) should move the player in diagonal directions.

3.3.1.1.2. The player has a set movement speed which can be modified by the Tiles system

3.3.1.2. Attacks

3.3.1.2.1. Attacks are visible effects during gameplay which, when they interact with an entity with HP (a player or enemy), reduce that entity's HP (this is called 'dealing damage') and/or apply additional effects to that entity. Both the Player and enemies are able to perform the attacks listed below. The player can control their attacks manually with the respective input, but enemies attack based on their programmed behavior.

3.3.1.2.2. The player's attacks have a cooldown which they will have to wait for before they can attack again. This can be reduced by the Cooldown Reduction % statistic from the Tiles system

3.3.1.2.3. Melee

3.3.1.2.3.1. Has a cooldown period where the attack can't be used. A static area at the player's position appears and deals damage to enemies in close proximity.

3.3.1.2.4. Ranged

3.3.1.2.4.1. Has a cooldown period where the attack can't be used. A moving projectile is created at the player's position and moves in the direction the player is facing. Can support modifications in trajectory depending on upgrades.

3.3.1.3. Win/Loss Conditions (HP)

3.3.1.3.1. Win: The player wins by decreasing the HP of the final boss to zero (or by timing out the clock?)

3.3.1.3.2. Loss: When the player's HP decreases to zero, the run is over. A "Game Over" splash screen will appear before the run summary screen pops up.

3.3.1.3.3. Run Summary Screen: Regardless of whether a player wins or loses a run, a final screen will appear with a summary of what the player achieved in the run. This summary screen will display information about item drops and the number/types of enemies defeated. They are taken back to the main menu.

3.3.2. XP Leveling

3.3.2.1. XP leveling occurs when the amount of XP picked up by the player passes a certain numerical threshold, then the level increments, and the collected XP amount begins at zero again. The player begins at level 0 with no XP. The threshold to gain a level increases with the level. Levels increase player's base stats, and item/equipment stats, such that the player's overall strength scales as the run progresses.

3.3.2.2. For every level, the player gains at base:

3.3.2.2.1. +1 HP

3.3.2.2.2. +0.25 Defense

3.3.2.2.3. +1 Attack Power

3.3.2.3. These bonuses may be augmented by buffs from the Tiling System

3.3.3. Enemy Behavior

3.3.3.1. Enemy Types

3.3.3.1.1. Enemy behavior depends on the type of enemy in question. Enemies spawned from standard waves which are not categorized as bosses have two general behavior types: Melee and Ranged. Regardless of the distance from the player, enemies will behave the same aside from only using melee attacks while in a certain range of the player.

3.3.3.1.1.1. Melee Enemy: Moves directly toward the player and attempts to use a melee attack to hit them. Melee attacks vary in hitboxes, but the size and shape of each hitbox is shown with clear in-game indicators.

3.3.3.1.1.2. Ranged Enemy: Fires projectile(s) of various types in an attempt to hit the player. Where and how these projectiles are aimed varies from enemy to enemy

3.3.3.2. Enemy Statistics

3.3.3.2.1. Enemies have two main statistics: HP and Base Damage

3.3.3.2.2. HP denotes the amount of damage they need to take from player attacks before they are defeated and disappear

3.3.3.2.3. Base Damage denotes the amount of damage enemies deal to the player when their attacks connect, whether melee or ranged

3.3.3.2.4. Enemy Speed: The speed at which enemies move

3.3.4. Gear

3.3.4.1. Weapons

3.3.4.1.1. Weapons change the type of attacks the player is able to perform. Weapons are separated into two different categories: Melee and Ranged. Melee weapons create hitboxes around the player while ranged weapons create projectiles originating from the player. The player may have one ranged and one melee weapon equipped at a time.

3.3.4.1.2. The hitboxes of melee weapons and the sizes of projectiles from ranged weapons may be enhanced by the Area %, Ranged Area %, and Melee Area % from Tiles

3.3.4.1.3. Melee Weapons: (These are ideas, subject to change)

3.3.4.1.3.1. Horns (Default)

3.3.4.1.3.1.1. Creates a triangular hitbox pointing out from the player's head. This hitbox can automatically adjust to aim at enemies within a specified angle of the direction the player is facing.

3.3.4.1.3.2. Sword

3.3.4.1.3.2.1. Creates a moderately wide hitbox in front of the player when the player attacks. This hitbox can automatically adjust to aim at enemies within a specified angle of the direction the player is facing. Deals more damage than the club but does no knockback.

3.3.4.1.3.3. Flail

3.3.4.1.3.3.1. Creates a series of curved hitboxes around the player which rotate in a circle. These do not automatically adjust to hit enemies but fire continuously in an orbital pattern. Cooldown reduction makes the hitboxes spin faster.

3.3.4.1.3.4. Spear

3.3.4.1.3.4.1. Creates a thin and long hitbox in front of the player when the player attacks. This hitbox can automatically adjust to aim at enemies within a specified angle of the player's facing direction. Deals similar damage to the sword but deals no knockback

3.3.4.1.4. Ranged Weapons:

3.3.4.1.4.1. Enchanted Horns (Default)

3.3.4.1.4.1.1. Appearance: Cow horns with glowing energy surrounding them.

3.3.4.1.4.1.2. Shoots a simple projectile in the direction the player is facing. The projectile originates from between the player's horns.

3.3.4.1.4.2. Homing Hooves

3.3.4.1.4.2.1. Cow hooves with glowing energy surrounding them

3.3.4.1.4.2.2. Shoots four homing projectiles at enemies. Each projectile can lock on to an individual enemy.

3.3.4.1.4.3. Behemoth Staff

3.3.4.1.4.3.1. Shoots a fireball a set distance from the player which turns into a persistent projectile that damages enemies. The

projectile is represented by the ground burning in the damaging area.

3.3.4.1.4.4. Grapes of Wrath

3.3.4.1.4.4.1. Shoot multiple grapes projectiles at an angle which spread out as they move farther away from the player

3.3.5. Tiling System

3.3.5.1. Tiles

3.3.5.1.1. Tiles are a type of gear which may be activated by the player in the Tiles Menu by placing them on a grid.

3.3.5.1.2. Tiles are shaped as continuous bitmaps which are no larger than 5x5 in any direction

3.3.5.1.3. The player may only utilize tiles which fit on the grid. Tiles successfully placed on the grid without overlapping are considered ‘equipped’ and will apply their respective bonus to the player when they enter their next run.

3.3.5.1.4. Tile Generation

3.3.5.1.4.1. When an enemy drops a tile, the tile constructor takes the enemy type as an argument.

3.3.5.1.4.2. The constructor chooses the tile size, S. The more powerful the enemy type defeated (8 > 4 > all others in numeric order), the higher probability the tile will have a higher size.

3.3.5.1.4.3. The constructor then chooses how many traits the tile will have, T, from 1-3. This converts to the scaling coefficient, C, as described below:

3.3.5.1.4.3.1. T = 1 => C = 1

3.3.5.1.4.3.2. T = 2 => C = 1.5

3.3.5.1.4.3.3. T = 3 => C = 2

3.3.5.1.4.4. The constructor then chooses randomly from the below traits, where N = S/C:

3.3.5.1.4.4.1. +#N HP gained on level

3.3.5.1.4.4.2. +#N attack power gained on level

3.3.5.1.4.4.3. +#N defense gained on level

3.3.5.1.4.4.4. %#N Increased damage

3.3.5.1.4.4.5. %#N Increased ranged damage

3.3.5.1.4.4.6. %#N Increased melee damage

3.3.5.1.4.4.7. %#N Increased area

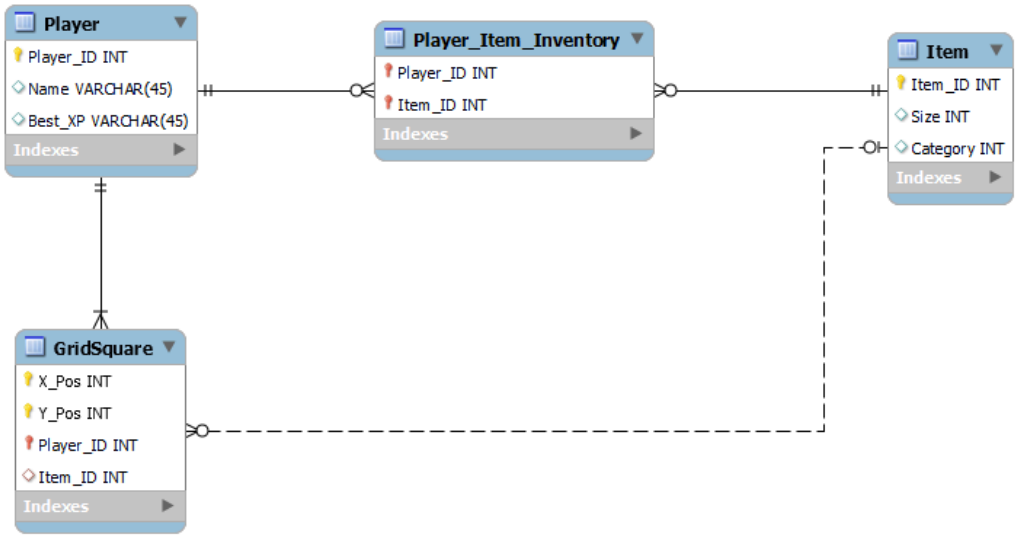
3.3.5.1.4.4.8. %#N Increased projectile area

3.3.5.1.4.4.9. %#N Increased melee area

3.3.5.1.4.4.10. %#N Decreased attack cooldown
3.3.5.1.4.4.11. +N movement speed
3.3.5.1.4.4.12. +N increased projectile count (can only drop from the boss)

3.3.5.2. Grid
3.3.5.2.1. The grid on which tiles may be placed. It is 5x5.

3.4. Database and Persistence



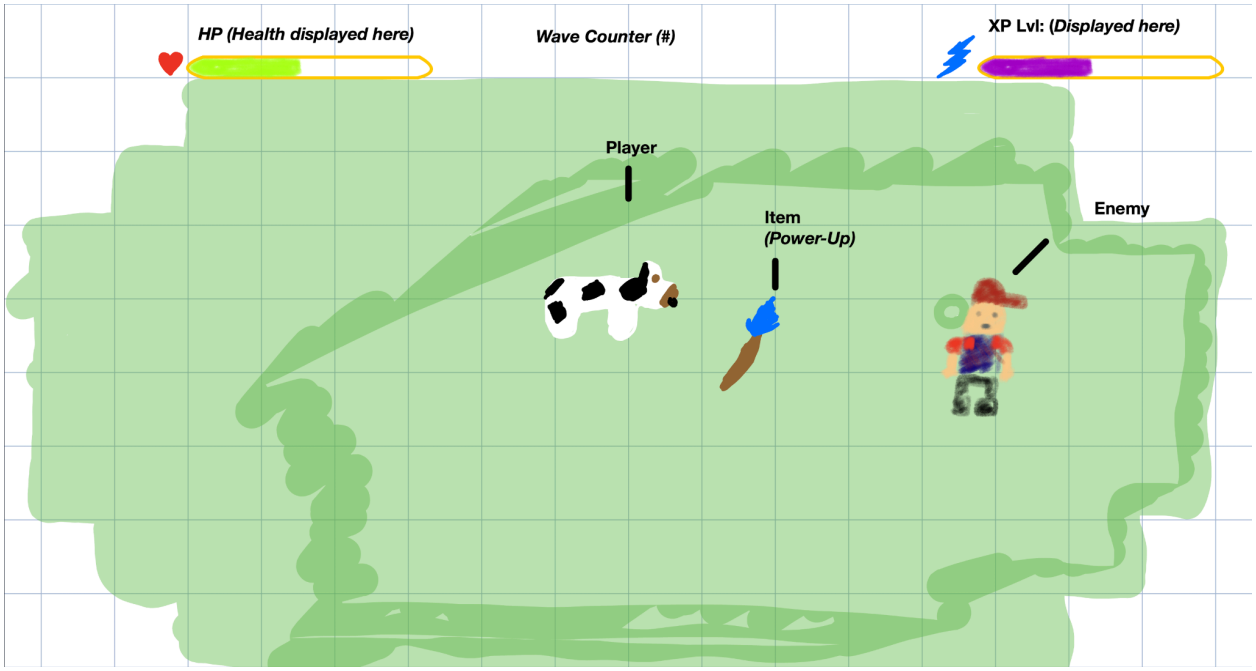
3.4.1. Relations
3.4.1.1. Player - Stores Unique ID, Name, Best Run XP
3.4.1.2. Item - Stores Unique ID, Size, Category (eg. Increased Damage)
3.4.1.3. Player_Item_Inventory (associative) - Stores Unique Player ID and Item ID for each Item picked up by Player
3.4.1.4. GridSquare - Stores which Item is in a given Square (X_Pos, Y_Pos) for each Player

3.4.2. Business Rules
3.4.2.1. A Player may have zero or many Items.
3.4.2.2. A Player may only have up to one of each unique Item.
3.4.2.3. Each Item can be owned by zero or many Players.
3.4.2.4. A Player must have 25 GridSquares.
3.4.2.5. Each GridSquare must be associated with one Player.
3.4.2.6. A GridSquare may have zero or one Items.
3.4.2.7. An Item can be in zero or many GridSquares.
3.4.2.8. An Item cannot be in more GridSquares associated with one Player than the Item's Size.
3.4.2.9. The X_Pos and Y_Pos of a GridSquare may only have values in [0, 4].

4. User Interface Design

4.1. Description of the User Interface

4.1.1. Screen Images



4.1.2. Objects and Actions