**Siege**

**Technical Design Document**

Alpha 2

**Version History**

|  |  |  |
| --- | --- | --- |
| **Version Number** | **Edited By** | **Date** |
| Alpha 1 | Nathan Sherwood | Hantao Zhu | 2021-03-11 |
| Alpha 2 | Sahil Shaikh | 2021-04-17 |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

Game Overview ............................................................................................................................. 3

Game Summary ................................................................................................................................ 3

Platform ........................................................................................................................................... 3

Development Overview ................................................................................................................. 4

Development Team .......................................................................................................................... 4

Development Environment .............................................................................................................. 4

Development Hardware.................................................................................................................. 4

Development Software.................................................................................................................... 5

External Code................................................................................................................................. 5

Game Mechanics ............................................................................................................................ 5

Main Technical Requirements .......................................................................................................... 5

Architecture .....................................................................................................................................6

Game Flow ....................................................................................................................................... 7

Graphics ........................................................................................................................................... 7

Audio ............................................................................................................................................... 7

Artificial Intelligence ........................................................................................................................ 7

Networking (If Applicable) ............................................................................................................... 7

Physics (If Applicable) ..................................................................................................................... 8

Game Objects and Logic ................................................................................................................... 8

Data Management and Flow ............................................................................................................. 8

User Interface ................................................................................................................................ 9

Game Shell .......................................................................... Play Screen ...................................................................................................................................... 9

Technical Risk ................................................................................................................................. 10

**Game Overview**

Siege is a 2 player competitive tower defense game where the players take turns

attacking and defending a static objective.

## **Game Summary**

Are you saddened by the loss of all the Flash tower defense games? Do you wish you could play tower defense with your friends? Siege aims to fix both these issues. It is a medieval themed tower defense game where the players take turns placing towers and sending attacking units. It is designed to be minimally intensive for modern hardware while still having visually interesting units and towers. The attacking player gains victories by successfully destroying the defending player’s base within the allotted time. While the defending player wins by not losing their base.

## **Platform**

The game is designed to be played on PC using keyboard and mouse using modern hardware as of 2015. An internet connection is highly recommended for play but is no technically required as a simple bot is to be implemented as an attacking player.

# **Development Overview**

This section discusses key aspects of the development of the game, as opposed to the game itself.

## **Development Team**

Sahil Shaikh – Team Lead, Programmer

Seth Grinstead – Programmer

Dante DeSousa– Artist

Hantao Zhu – Artist

Nathan Sherwood – Artist

## **Development Environment**

Thanks to the wonders of developing through a global pandemic, we are all working from home and cooperating through various digital means. We use Discord for all our communications vocal and textual as well as any quick, small file transfers. We use GitHub for all our large permanent file transfers and storage. Our organization is done via Trello.com and is divided into 2 week sprints. For software we use Unreal Engine 4.25.4 as our engine and map creation tool, 3ds Max is the program we use for 3d modelling

### **Development Hardware**

Since we are working from home we each have a wide variety of systems with a wide range of performance characteristics. Including both laptops and desktops. However, we all are operating on windows 10 so we can at least be sure software will run on all our machines.

### 

### **Development Software**

We use Discord for all our communications vocal and textual as well as any quick, small file transfers. We use GitHub for all our large permanent file transfers and storage. Our organization is done via Trello.com and is divided into 2 week sprints. For software we use Unreal Engine 4.25.4 as our engine and map creation tool, 3ds Max is the program we use for 3d modelling.

### **External Code**

Other than the various programs and software we’d mentioned before, (particularly the various utilities of Unreal Engine) we have not used any external code.

# **Game Mechanics**

After every round players get to buy troop or towers, which they can strategically spawn and place during the match to defeat the other player. Towers once placed can not be moved while a round is in progress but can be destroyed and replaced with another tower of choice. Troops are individually deployed,and can’t be moved or guided once spawned.Player are suppose to pre-plan their next moves.

## **Main Technical Requirements**

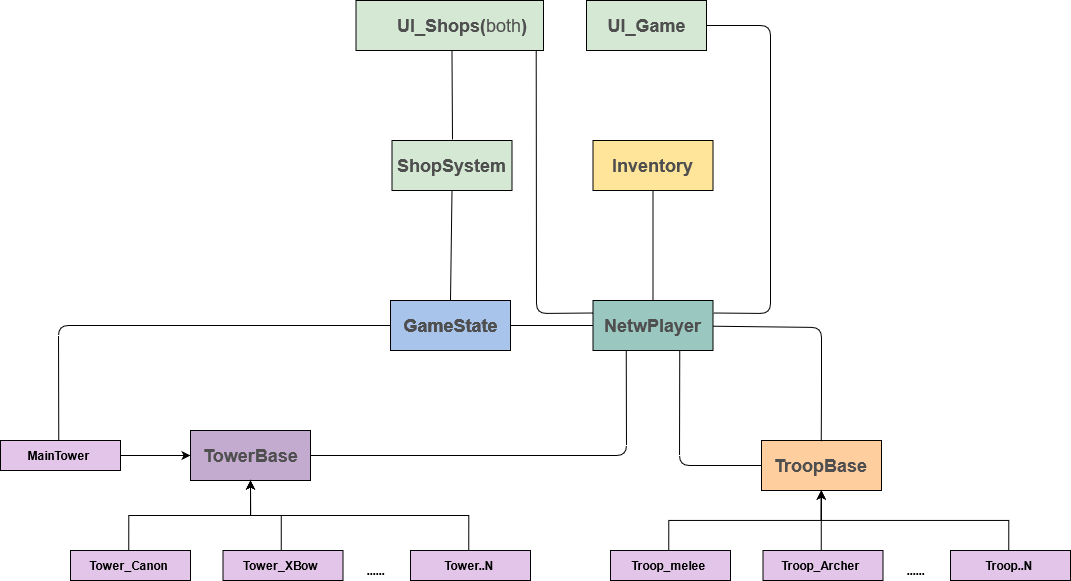
The game will require large maps with several entities, each with its own AI logic. These entities will have to roughly follow a predetermined path as well as break from the path to attack nearby towers if applicable. Most of the units will have ranged attacks which will quickly multiply the number of entities on the screen. As such, the pawns and their projectile’s logic will need to be simple to not cause undue stress on target hardware. Their visual footprint as well must therefore be minimized while still being distinct to the player.

## 

## 

## 

## **Architecture**



**NetwPlayer:** This handles events happening during a match, such as spawning troops or towers on spawn points on user input. It stores player data like

Inventory,roundsWon.It also has functions for UI to interface to inventory.

**Inventory**: It keeps tracks of player’s bought items and player’s money. It also provides functions to other classes such as shopsystem, player to add or remove a troop pr tower from inventory.

**ShopSystem**: This handles the shop interface for players. It has blueprint functions that are triggered on the user interacting with the shop UI to purchase items.

**GameState:** GameState is responsible to switch between UI widget based on current state of the game. It tells ShopSystem to display which widget based on the player's role(attacking/defending).It transitions between states like play, intermission(buying period),GameOver.It performs UI actions and clear units from world when round ended.

**TowerBase:** This is responsible for detecting troops in range and calling attack() on the child class. All Tower are similar when it comes to troop detection so that code is defined in the base class while each different tower has a different type of attack, hence that code is present in the child class called by the TowerBase class on detecting troops in near range.

**MainTower:**It is the main tower in the map that when destroyed notifies GameState and declare victory of the player attacking.

**TroopBase:** This class detects towers in its path and notify the child class to perform attack or any other response. Similar to TowerBase, TroopBase does the tower detection while the actual attack code which is unique to the troop is in the derived troop class.

## 

## **Game Flow**

The final game flow is expected to be such,player click on start a match ,the game will look for another player in the que and match them against each other.Randomly one player will start as attacker and another defender.Players are given some gold to start with and purchase there units/tower.For every unit/tower kill players get rewarded ,along with reward for winning ,finish a round.after 2 rounds both will switch roles and will still carry the same gold they had.Whoever wins the majority of the rounds will be the winner.This game flow will be working once multiplayer has been im-plemented

## **Graphics**

In order to keep performance at a maximum, the units visual design will be low poly count 3d model with simple textures. These assets are created to depict a generally accurate but cartoonish rendition of a fantasy medieval setting.

We are using assets we have created ourselves as much as possible so as to keep a consistent visual theme. Advanced lighting will be used sparingly to keep performance high.

## **Audio**

Audio will be limited to mostly Foley sounds and music that has been sourced from non commercially licensed files available online. They will comprise of mostly organic sounds in keeping with the medieval setting.

## **Artificial Intelligence**

Each unit entity (tower or mobile unit ex: a soldier) will possess its own AI to direct it. Tower’s will be very simple, consisting only of functionality to turn to and fire at hostile units. The unit’s on the other hand will direct it down a preset path until it detects a tower within its range. At which point it will run off the path to engage the tower with whatever weapon it has. The computer opponent will simply be a preset list of units to spawn.

## 

## **Networking (If Applicable)**

Siege is 1V1 online multiplayer game ,it uses Unreal networking protocols.The client computer only can input actions but the actual code of objects are performed on Server.

In our game ,both NetwPlayer class keeps track of player’s data that needs to be synced, GameState on server uses that data to change state,

identify winners

## **Physics (If Applicable)**

Both troops and towers contain 2 collider.1 for physical collision and 2nd for detecting one another . When a tower detects a troop it calls the attack function in the child class to shoot projectile.Similarly troop detects tower in there path move towards it and start there attack, and once the tower has been destroyed it returns to its path and continues moving towards the base.

## **Game Objects and Logic**

**Tower\_canon:** Objects of this class are derived from TowerBase class,which provide troop detection and other sets basic variables and functions.Canon shoots a parabolic projectile and does splash damage.It has low fire rate and does heavy damage to troops.

**Troop\_melee:** Objects of this class are derived from TroopBase,which contains sets of common variable and functions needed by troops.This troop detects tower in its path,sets it as target and moves towards it,once close enough ,it start damaging the tower at a slow rate and does moderate damage.

**Player:** This handles events happening during a match, such as spawning troops or towers on spawn points on user input.

**Inventory**: It keeps tracks of player’s bought items and player’s money. It also provides functions to other classes such as shopsystem, player to add or remove a troop pr tower from inventory.

## **Data Management and Flow**

The game doesn’t require any saving and loading functionality; Things like currency, troops and towers purchased are only meant to be use for that particular match and have no other use.

# **User Interface**

## **Game Shell**

The game will be presented in the Eastern European style of the 15th century. The main color is wood brown.

The game comes with sound and music, which will cover the sound effects of soldiers‘ fighting, moving, cannons’ bombing, steering, etc. The background concert uses a tighter sound.

The game interface uses visual effects that look like kraft paper and font uses Script.The combination of the two will be a retro art style.in addition, decorating with Sealing wax, Ink stains and Dip pen can enhance the classic effect.

## **Play Screen**

Basically, the attacking side player and the defending side player have the same interface base, just a little bit different.

First on the top left it shows the current round od the game, and in the top middle is the round wins of both player

In the botten, one big box is here. in the big box, the type of troop/tower show here

After round ends, UI Text will be shown saying “Won” or ”Loose”.

# **Technical Risk**

Unfamiliarity with development tools being used is a big problem. We started using Unreal Engine 4 a half year ago, but we are not very proficient in using this tool, and sometimes we don’t even know which file type to import even, for example the player mesh needs to use. fpx but we try to use .3dx.

It’s like trying to perform things beyond our capabilities or trying to do many things with limited engines/hardware. Just like the number of units we plan to spawn may be too much for our personal CPU and GPU.

When we program the game, a huge number of bugs come up. Want to limit the purchase, the number of troops is limited, this problem. Even if we mitigate its risks. Loss of scope is always a risk.

In this game we hope to use our own model. For this goal, we have to learn how to build a player model (We did not learn how to build a human body in 3ds max yet). It has been a big challenge for us. In addition, we spend many different tools to build for example maya, ZBrush, 3dx max etc.. This caused incompatibility between softwares.