Game design report

Game name:

Long Liu 20104729

Software system practice Year 4

Date:

Supervisor:Mairead Meagher

**1. game overview**

**1.1 game concept**

This game is Horizon version adventure game. Players can move, left and right. Players have a base to upgrade abilities. The enemy will appear at a fixed range and attack the player. As time goes on, the player can collect materials to make tools and update modes. By these objects, payers can increase the maximum life value. When the game enters some specific days, some enemies will actively attack the player's base. Player need to guard base to avoid failure of the game. Player can gain materials by killing enemies and collecting. Materials can be used to upgrade base and player. On the base, player can build defense buildings.

**1.2 genre**

The type of the game will be a horizontal version adventure game. Players will watch the game horizontally from top to bottom and scroll horizontally.

**1.3 Target Audience**

My target audience will be people who have interests in adventure. It mainly refers to players who can accept pixel style, combat with enemies and discover new things. Targeting the market for high school students and college students..

**1.4 Game Flow Summary**

The game will load the main menu, where players can select new games, options, or archives. In the game itself, player needs to upgrade base to the highest level, and hold the base during an attack triggered by the upgrade.

**1.5 Look and Feel**

I want the game to look and feel like a pixel adventure game. Similar to “Kingdom: New Lands”or”Until We Die”. Here are some relevant games pictures:



Kingdom: New Lands

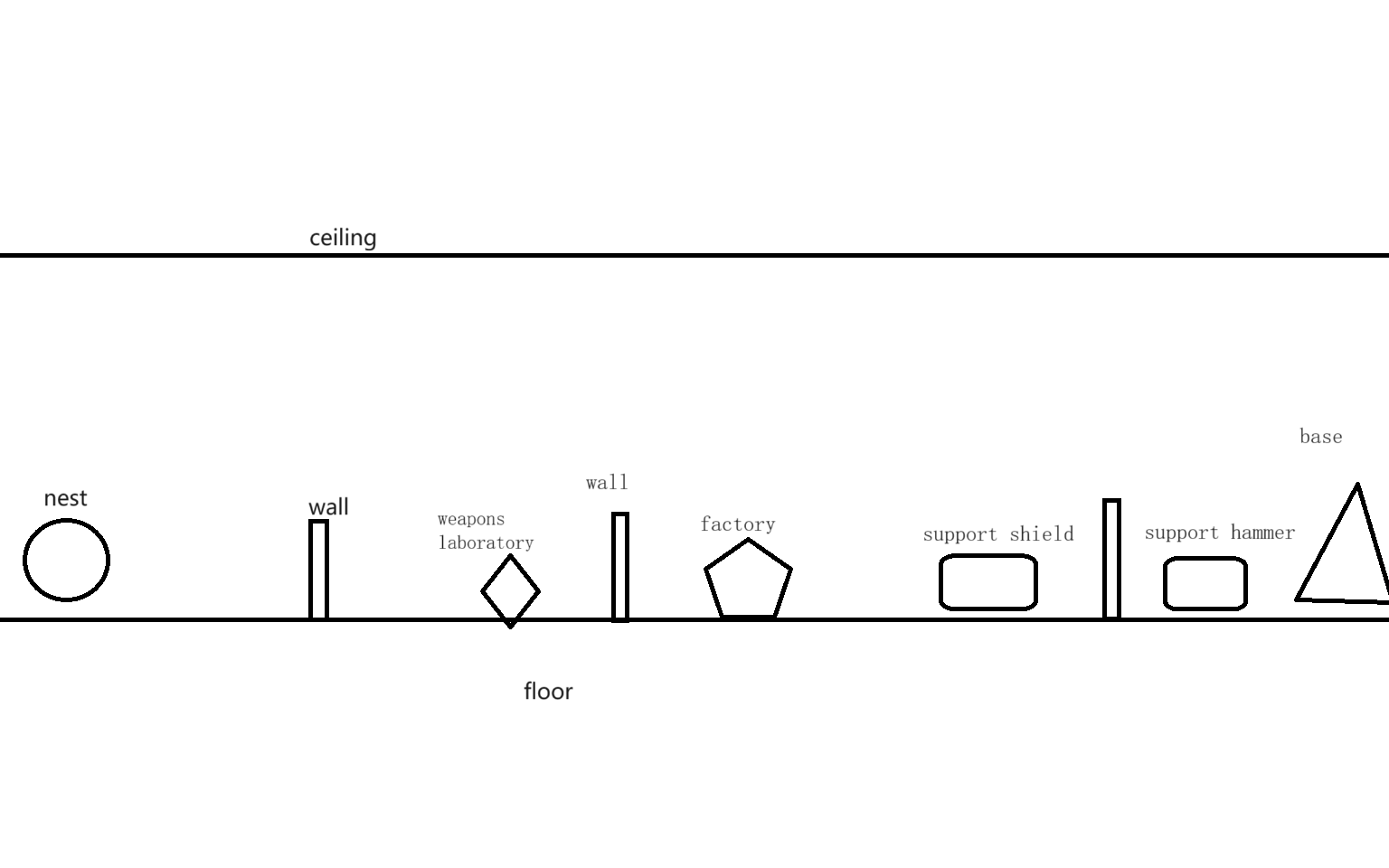


Until We Die

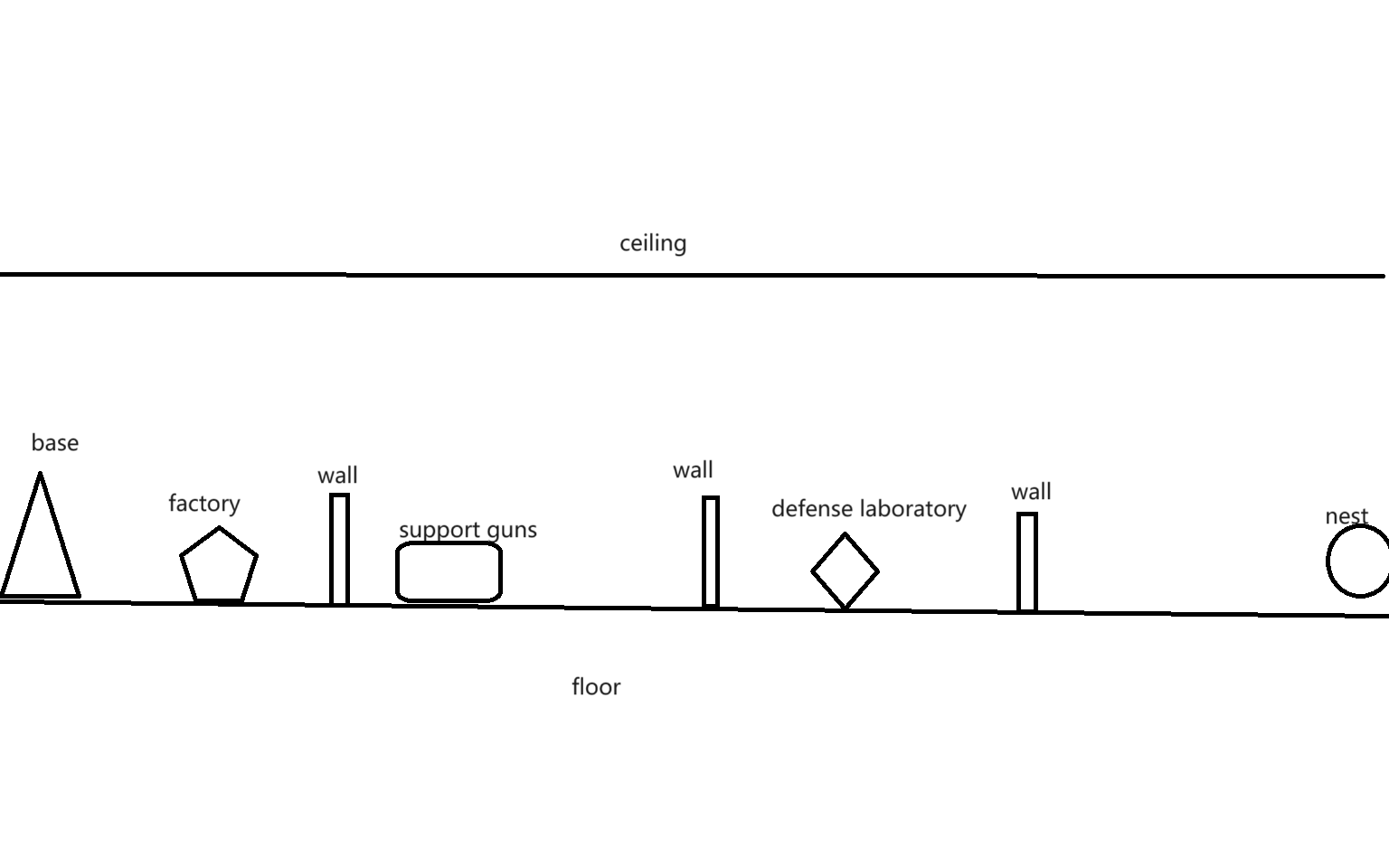
**1.6 Project Scope**

**1.6.1. Number of Location**

Player and enemies will born to an only one map. Player and base will appear on the centre of the map, and enemy nests will appear on the two ends of the map. Some broken buildings or hidden buildings will appear on the space between base with nests. Player can repair these broken buildings to get some annexe. These annexe can support some extra functions.



Left half map



Right half map

**1.6.2. Number of Enemies**

There will be three kinds of enemies.

Crawler: attack from above, low health, fast speed , medium attack power

Attacker: attack from frontal, medium heath, medium speed ,medium power

Thrower: range attack, low heath medium speed, medium attack power

Every enemies come from nest.

**1.6.3. Number of Buildings**

Base location: 1 base, 2 nests

Annexe location:

6 wall building points

3 kinds of buildings for support tools

1 weapons laboratory

1 defense laboratory

2 factories

Base: improve other buildings’ max level, attract new refugees interval time, have 3 levels,

If base is destroyed, the game will over.

Wall: defense attack ,have 3 levels

Weapons laboratory: improve attack power, have 2 levels, need workers to finish update projects

Defense laboratory: improve wall health, have 2 levels, need workers to finish update projects

Factory: produce materials, have 2 levels, need workers to work

buildings for support tools: support hammers, guns, shield, need workers to work

Nest: enemies will appear there, have 3 levels.

**1.6.3. Number of NPC**

There are 5 kinds of NPC.

Civilian: refugee will be civilian after they get one piece of material.

Worker: build buildings ,repair buildings, need hammer

Soldier: range attack, need guns

Defender: have shield for blocking attack. Defender will use lance to attack, when they behind wall.

Refugee: the primary level people,need material

There are some neutral organisms.

Huge-mouse: It will provide some materials after it dies.

**1.6.3. Number of Player**

Range attack, have a bag, improve the bag volume rely on the level of base, medium attack power, high health, medium speed, use power increase speed in a short time,

If player died, the game will end.

**4. Gameplay and Mechanics**

**4.1. Gameplay**

**4.1.1. Game Progression**

Starting out, the player will appear at the base with some initial materials. These materials can be used to hire civilians. Subsequently, the player can create tools like guns or hammers by constructing corresponding buildings. This will change the jobs of the civilians: workers will operate the factory to gather more materials, while soldiers will hunt neutral organisms for additional resources. Meanwhile, nests will periodically produce enemies that attack the base. As the game progresses, the player will construct more buildings, and the nests will generate increasingly more enemies. The player's objective is to destroy the two nests before being overwhelmed by the enemies.

**4.1.2. Mission/challenge Structure**

Every night, the nests will generate a few enemies to attack the base. Every five days, they will produce a large number of enemies. The player needs to defend the base and eliminate all enemies; otherwise, the game will end. On the following night, when no enemies are generated, the player can take advantage of this time to explore the map.

**4.1.4. Objectives – What are the objectives of the game?**

The player should protect themselves and the base, and aim to destroy the nests.

**4.1.5. Play Flow – How does the game flow for the game player**

The player can use materials to make refugees into civilians. They will leverage these materials to produce more tools, transforming civilians into workers and soldiers. Workers can build more structures after the materials for these buildings are provided. The player can construct additional walls and annexes. Once the base has enough buildings, the player can produce shields to turn civilians into defenders.

As time progresses, enemies will attack the base, and soldiers will defend against these attacks using the walls, ultimately eliminating the enemies. After the battle ends, workers can repair damaged buildings with materials. Through the cycle of development, defense, and repair, the player will accumulate enough soldiers and defenders to attack the nests. The player can summon combatants to launch an attack. The game will end in failure if the player's health points reach zero or if the base is completely destroyed. Victory is achieved when the two nests are destroyed.

**4.2. Mechanics – What are the rules to the game, both implicit and explicit? This is the model of the universe that the game works under. Think of it as a simulation of a world, how do all the pieces interact? This actually can be a very large section.**

**4.2.1. Physics – How does the physical universe work?**

Physics in games monitor collision detection to keep track of game objects in contact with each other. Since this is a 2D game, the character can only move left and right. It does not account for height differences, so bullets will not fall. However, the bullet will still disappear after reaching its range. The game map consists only of x and y planes, so all physical calculations will mostly be two-dimensional.

**4.2.2. Movement**

**4.2.2.1. General Movement**

Players use the 'A' and 'D' keys on the keyboard to move and the left mouse button to control attacks. The perspective is always centered around the character controlled by the player. Players move at a fixed speed. Pressing 'Shift' allows players to expend energy for quick movements for a short period. After running out of stamina, the player will stop their fast movement and move at a slower speed than usual. Stamina will slowly recover. If the player stop moving, your stamina will recover more quickly.

**4.2.2.2. Other Movement**

Soldier/Defender: They will stand after the wall. If the player pay some materials, they will move with the player.

Worker: They will move to any places that need them.

Civilian: They try to move the places that exist tools.

Refugee: They randomly move in a fixed area.

Enemies: They attack any buildings.

**4.2.3. Objects**

**4.2.3.1. Picking Up Objects**

**4.2.3.2. Moving Objects**

**4.2.4. Actions**

**4.2.4.1. Switches and Buttons**

**4.2.4.2. Picking Up, Carrying and Dropping**

**4.2.4.3. Talking**

**4.2.4.4. Reading**

**4.2.5. Combat – If there is combat or even conflict, how is this specifically modelled?**

**4.2.6. Economy – What is the economy of the game? How does it work?**

**4.3. Screen Flow**

**4.3.1. Screen Flow Chart – A graphical description of how each screen is related to every other**

**4.3.2. Screen Descriptions – What is the purpose of each screen?**

**4.3.2.1. Main Menu Screen**

**4.3.2.2. Options Screen**

**4.3.2.3. Etc.**

**4.4. Game Options – What are the options and how do they affect game play and mechanics?**

**4.5. Replaying and Saving**

**9. Technical**

**9.1. Target Hardware**

MINIMUM:

|  |  |
| --- | --- |
| OS: | Windows 7 |
| Processor: | Intel 4th Gen Dual Core 2.0Ghz |
| Memory: | 2 GB RAM |
| Graphics: | Nvida GTX Series 8 |
| Storage: | 4 GB available space |
| Additional Notes: | The game can likely run on lower rated hardware, but I can't guarantee the performance or provide support. |

**9.2. Development hardware and software**

**9.2.1 Hardware**

|  |  |
| --- | --- |
| OS: | Windows 11(x64) |
| Processor: | 13th Gen Intel(R) Core(TM) i9-13980HX 2.20 GHz |
| Memory: | 32.0 GB |
| Graphics: | Nvida RTX 4090 laptop |
| Storage: | 1 TB available space |

**9.2.2 Software**

Godot v4.12 is the latest official version when I started developing the prototype.

Git,Microsoft Visual Studio,etc.

**9.3. Game Engine**

**Godot** is a cross-platform, free and open-source game engine released under the permissive MIT license. It was initially developed by Argentine software developers Juan Linietsky and Ariel Manzur for several companies in Latin America prior to its public release in 2014. The development environment runs on many platforms, and can export to several more. It is designed to create both 2D and 3D games targeting PC, mobile, and web platforms and can also be used to develop non-game software, including editors.

Godot allows video game developers to create 3D and 2D games using multiple programming languages, such as C#,C++,GDscript.

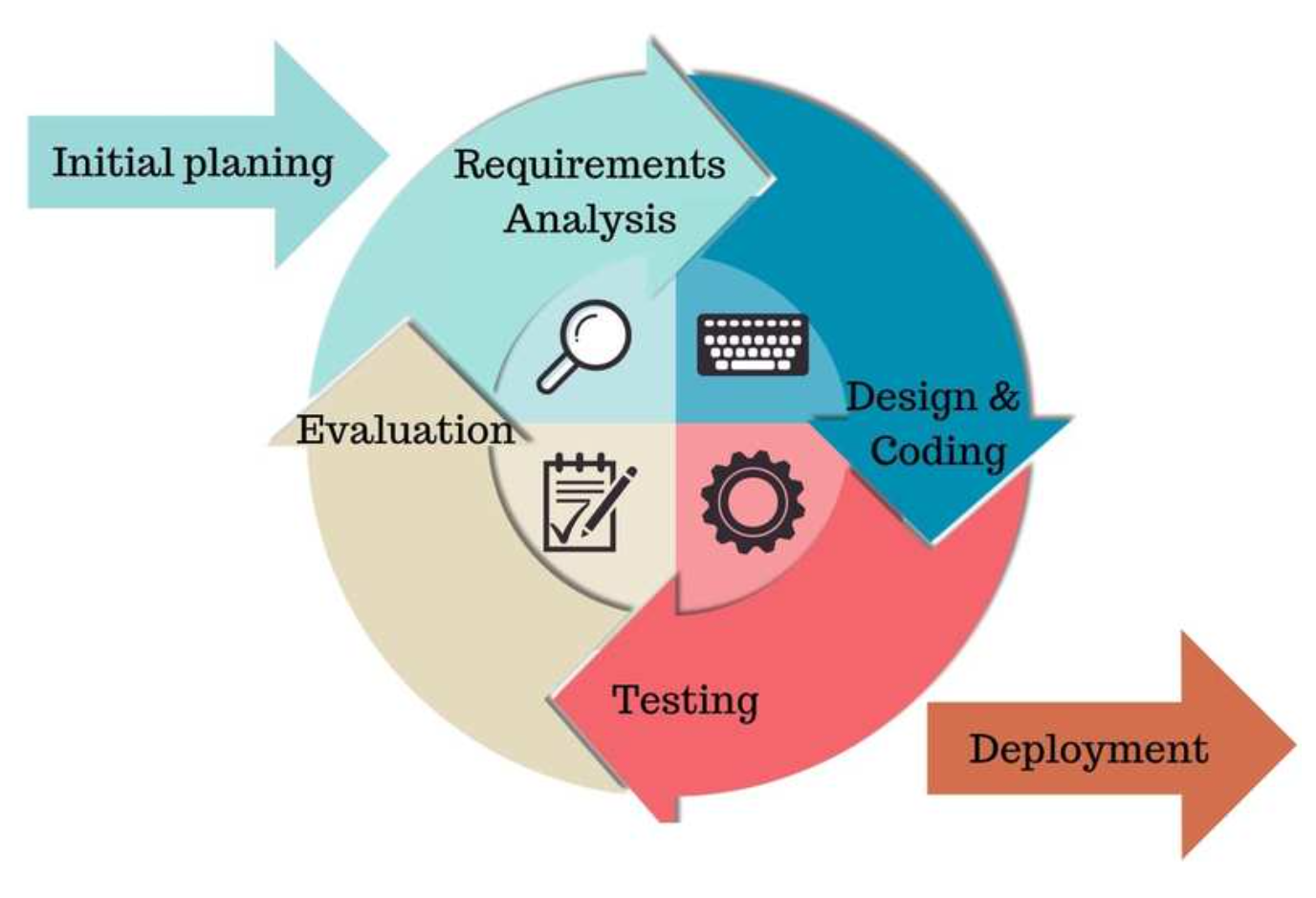
**9.5. Scripting Language**

The scripting language that will be used for this project is C#. Because it is one of the main languages of choice for scripting in the Godot game engine.

**11. Project Management**

**11.1. Project Methodology**

In this game development project, I chose to adopt an agile development method. Agile methodologies are known for their high adaptability to rapid change and uncertainty and are well suited to meet the ever-changing needs and challenges of game development. In contrast to the waterfall model, agile development divides the development process into shorter iterative increments. At the end of each iteration, a final usable product is produced. If requirements change, the method can be developed in the next iteration cycle.



**11.2. Detailed Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sprint(2 weeks) | Goals | Tasks |
| 1 | Prototype development and basic framework | Establish the basic framework and prototype of the game. | Develop the most basic game mechanics and a simple user interface. |
| 2 | Core Game Gameplay Development | To achieve the core gameplay of the game | Encode the main game features, such as motion control and basic interaction. |
| 3 | Map Design and Development | Design and implement preliminary game map. | Create map of the game and integrate them into the game |
| 4 | Advanced features and interactivity | Add advanced gaming features and interactivity. | Implement special effects, complex game logic, etc. |
| 5 | Testing and Optimization | To test the game and resolve any issues discovered | Conduct system testing, optimize performance, and improve user interface. |
| 6 | Preparing for Beta Release | Prepare for the beta version of the game. | Make final adjustments and optimizations |

**11.3. Version control**

Github is a hosting platform for open source and proprietary software projects. As an open source repository and version control system, Github has more than 9 million developer users. As more and more applications move to the cloud, Github has become the preferred method for managing software development and discovering existing code. So I'll use Github to keep pushing code.

However, once the project exceeds 100MB, Github will refuse to upload it. So I only uploaded the part that designed the key code, not the whole project.

**11.4. Risk Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| Risk Description | Risk Probability | Risk Impact | Risk Mitigation Strategy |
| Underestimating engineering difficulty | High | High |  |
| Contracting illness | Low | Low | Avoiding Sickness |
| Inability to meet deadlines | Med | Med |  |
| Lose of Report/Documentation | Low | High | Documentation stored on Baidu Yun Drive with regular Copies stored on several devices locally. |
| Lose of Project/Workspace | Med | High | Project stored on Baidu Yun Drive with regular Copies stored on several devices locally.kept on local devices. |

**11.5. Test Plan**

|  |  |
| --- | --- |
|  | Test Plan |
| When? | At the end of every sprint. |
| Who? | Members of available game groups |
| Where? | Send testers a test version of the game via the game forum. The lack of connection between developers and testers facilitates critical analysis. |
| What? | Specific problems found during development. |
| How? | The developer will not be present and will provide playtesters with the game executable and a short questionnaire. Outside of the questionnaire, they can also make any complaints or suggestions they may have. |