Technical Architecture

Evaldas Cesna X00112310

1 Use Cases

1.1 Title (goal) : Log In

1.2 Primary Actor : User

1.3 Scope : Login system

1.4 Level : User Goal

1.5 (Story) : The User logs in and is authenticated to use the system.

2.1 Title (goal) : Play Game

2.2 Primary Actor : User

2.3 Scope : Game System

2.4 Level : User Goal

2.5 (Story) : The User clicks "Play" on the GUI which initiates the single player game.

3.1 Title (goal) : Join Game

3.2 Primary Actor : User

3.3 Scope : Multiplayer Network System

3.4 Level : User Goal

3.5 (Story) : The User joins a multiplayer game chosen from the GUI.

4.1 Title (goal) : Create Game

4.2 Primary Actor : User

4.3 Scope : Multiplayer Network System

4.4 Level : User Goal

4.5 (Story) : The User creates a multiplayer game from the GUI for others to join.

5.1 Title (goal) : Movement

5.2 Primary Actor : User

5.3 Scope : Game System

5.4 Level : User Goal

5.5 (Story) : The User moves his character after pressing the movement buttons.

6.1 Title (goal) : Interact with Inventory

6.2 Primary Actor : User

6.3 Scope : Game GUI System

6.4 Level : User Goal

6.5 (Story) : The User equips items and manages his inventory, after clicking the inventory button.

2 Technical Architecture

2.1 Software Components

Unity Engine

Unity uses a component based software architecture, which means that the whole game is structured from different components you make. One component can give behaviours to another component which is more efficient for game development as it does not need to use inheritance or polymorphism to accomplish simple tasks. Unity is also able to make the same code usable for different operating systems which means you can make a game and have it work on PC, Android etc.

Android

Android OS is a stack of software components divided into five sections and four layers which are Linux Kernel, Libraries, Android Runtime, Application Framework and Applications.The game will run on the Applications layer and use some of the Application Framework for resource management.

SQLite

SQLite compiles SQL text into bytecode and runs the bytecode using a virtual machine.

2.2 Platform Libraries

Unity High Level API

Used to add multiplayer capabilities to a unity game. Its built on lower level transport real-time communication later and handles common tasks that are required for multiplayer games.

C#

C# will be used on the Unity engine which is compatible with Android OS.

SQL

SQLite queries will be written in SQL to register the user to the database and authenticate the user when he logs in.

2.3 Distribution and Deployment

1. The multiplayer server will be hosted by Unity Multiplayer which uses UDP based protocol, multi channel design that avoids head-of-line blocking issues, supports a variety of Quality of Service per channel and has a flexible network topology that supports peer-to-peer or client-server architecture.

2. With some simple obfuscation of code it can stop some cheaters from editing their game, also the login system will authenticate the user that is playing the game and in the event of a person cheating in game like manipulating items or health the account can be banned.

2.4 Risks

The risks are figuring out what can be stored in the SQL database other than login data as the server itself could handle it better by itself and making the SQL database pointless.

Another risk is not having enough time to figure out the unity engine and how to develop a multiplayer game at the same time is a big issue as it is traditionally not a one person job.

3 Prototype

3.1 Prototype Deliverable for week 8
The "Login", "Play" and "Movement" use cases are deliverable for week 8 and to test them I will run the application on my computer and my Android phone.

3.2 Prototype Deliverable for week 11
The "Join", "Host" and "Inventory" use cases are deliverable for week 11 and to test them I will run the Unity Multiplayer server and the application on my computer and my Android phone.