Faculty of Technology, Design and Environment

Oxford Brookes University

School of Engineering Computing and Mathematics

BSc (Single Honours) Degree Project

Programme Name:…… BSc Computing Project……..,……

Module No. ……COMP6013: ……..…...

Surname: …….St Johnston…….……

First Name: ……Sebastian……..

Project Title: ……. Lemonade Stand | Multiplayer, Economy-based Android Game Concept…

Student No ……..19129576…….……

Supervisor ……...Tjeerd Olde Scheper……..

2™ Supervisor ……

(if applicable)

Date submitted……March 12th of 2022……..……..

A report submitted as part of the requirements for the degree of BSc (Hons) in Computer Science

At

Oxford Brookes University

Student Conduct Regulations:

Please ensure you are familiar with the regulations in relation to Academic Integrity. The University takes this issue very seriously and students have been expelled or had their degrees withheld for cheating in assessment. It is important that students having difficulties with their work should seek help from their tutors rather than be tempted to use unfair means to gain marks. Students should not risk losing their degree and undermining all the work they have done towards it. You are expected to have familiarised yourself with these regulations.

https://www.brookes.ac.uk/regulations/current/appeals-complaints-and-conduct/c1-1/

Guidance on the correct use of references can be found on www.brookes.ac.uk/services/library, and also in a handout in the Library.

The full regulations may be accessed on-line at https://www.brookes.ac.uk/students/sirt/student-conduct/

If you do not understand what any of these terms mean, you should ask your Project Supervisor to clarify them for you.

I declare that I have read and understood Regulations C1.1.4 of the Regulations governing Academic Misconduct, and that the work I submit is fully in accordance with them.

Signature …Sebastian St Johnston…… Date …10/12/21……..

REGULATIONS GOVERNING THE DEPOSIT AND USE OF OXFORD BROOKES UNIVERSITY MODULAR PROGRAMME PROJECTS AND DISSERTATIONS

Copies of projects/dissertations, submitted in fulfilment of Modular Programme requirements and achieving marks of 60% or above, shall normally be kept by the Library.

| agree that this dissertation may be available for reading and photocopying in accordance with the Regulations governing use of the Library.

Signature ……Sebastian St Johnston…….. Date ……17/02/22……..

Table of Contents

COMP6013: BSc Computing Project: Lemonade Stand | Multiplayer, Economy-based Android Game Concept

Chapter 1: Introduction

1.1 Background Information

In the era of technology creation and innovation, projects, problems, and research ideas are being promoted to advance improving quality of life or inquire in compelling solutions. The final year in Computer Science, COMP6013 – Computing Project, has assigned a task to investigate, and document the process involving a study of a solution to any practical problem. With the intention of the solution is represented to be a culmination of knowledge gained through the course. In general, applications, programs, systems, or solutions are engineered or developed as a product of these problems. The timeline and the methodologies of the process must be achieved using high quality standards and documented as if it could be reimagined.

Understanding how to solve this problem calls for a tactical method to encompass the problem with a highly populated demographic feature by developing and designing for a product that everyone has. Nowadays, everyone in this century will have a smartphone in their pockets. This highly promotes creating and innovating mobile applications as everyone will be able to use the product and access it anywhere.

Next, an issue that is concerned in this project and is discussed immediately, money. The world runs on money, and everyone needs money. To expand on this, money, economics and how to generate revenue are concepts everyone should understand naturally, and they are applicable real-world skills. The way of understanding and teaching business and economics is tough especially when the studies and education is based through learning theories and applying case studies. In contrast, using practical real-world applications is an effective way to convey how assets and capital are affected.

The project that will be designed and developed for this solution will be a personalised, enhanced version of a classic videogame called ‘Lemonade Stand’ (Wikipedia, 2021). The game is defined as a Business Simulation game and involves a player purchasing lemonade stock: sugar, lemons, water; then moves into a selling state where the player sells lemonade using their stock and recipe. Depending on the state of the day such as the weather or temperature, sales and demand of lemonade will fluctuate, and the player will have varying profits or losses per day.

The first commercial ‘Lemonade Stand’ game was developed in 1973 by Bob Jamison of the Minnesota Educational Computing Consortium which was developed for the Apple I and ported to the Apple II. In general, the objective of the game is to generate profits and upgrade the virtual assets within the game. For instance, a player may be saving up to buy higher quality lemons or a nicer lemonade stall. This project’s vision of ‘Lemonade Stand’ aims to promote capitalistic objectives such as making as much money as possible or creating the best type of lemonade that will generate the most revenue. Furthermore, the game will be multiplayer based enabling competition between players and allowing them to trade.

Introducing the concept of trading opens the idea of a virtual economy. Where players exchange virtual goods in a virtual environment. Buying an economy-based game will require an understand of economics and how it can be applied to this application. The main four aspects of economics involve: scarcity, supply & demand, cost, benefits & incentives. In the multiplayer game ‘Lemonade Stand’, creating a scarcity of the main commodities will allow the virtual economy to exist as it creates a flow of supply and demand for the available commodity stock. The game will be designed for players to create demand for other players commodities and items. Some players may have a surplus of goods that will create a market economy amongst themselves. To build on this, a feature of the game could include different types of lemons, create competing commodities with different demands. For example, a special ‘Alien lemon’ will introduce ‘Alien customers’ that will pay in an ‘Alien currency’. Furthermore, to get a unique lemon will require a system of lemon breeding to introduce the lemons

into the economy, where ‘lemon a’ + ‘lemon b’ may generate an ‘Alien lemon’ or any other unique lemon. Another method of bringing unique lemons into the economy is through a basis of chance by any means.

Finally, this project hopes to involve several interesting topics as it entertains the idea of a generic mobile game with a unique multiplayer economy feature with further, yet discussed, concurrent technologies.

1.2. Aim

The aim in this project is to improve the issues of understanding money and the economy by building an accessible application, as mentioned. The project will involve teaching and applying business and economic theory throughout the game. Targeting a large demographic with time to learn and provide them with the ability to play with a virtual economy. As a result, the idea requires to develop a mobile application game that involves a multiplayer implementation so that the players can create the virtual economy based on lemons and lemonade; By using an application framework for building a server that will host the networking requirements with a database.

1.3. Measurable Objectives

The ‘Measurable Objectives’ table showcases the measurable objectives found in the project. Each object is numbered, given a summary and when it is due. This will provide an overview of how the project is to be completed and the timeline of the task completion due dates.

Measure Objectives		
Objective	**Task and Deliverable(s)**	**Goal & Due date(s)**
Obj. 1.1	Complete a full background review and analysis of existing/similar products.	End of Semester 1, Week 4.
Obj. 1.2	Review ethics and legal aspects of developing this project.	
Obj. 1.3	Design a ‘Project Proposal’ form of intentions.	
Obj. 2.1	Immediate development of the product.	Start of Semester 1, Week 5.
Obj. 2.2	Establish UML planning, and artefact creation.	
Obj. 2.3	Review and analyse formal due dates to complete the project.	Start of Semester 1, Week 6.
Obj. 3.1	Develop a minimum viable product application.	End of Semester 1, Week 11.
Obj. 3.2	Design a ‘Progress Report’ for intentions and reference.	End of Semester 1, Week 12.
Obj. 3.3	Develop servers and a database for the application.	Start of Semester 2, Week 2.
Obj. 3.4	Design and develop an algorithm for the virtual economy.	
Obj. 3.5	Virtual economy algorithm implementation in application and database servers.	End of Semester 2, Week 3.
Obj. 4.1	Begin writing Dissertation.	
Obj. 4.2	Iterate features, innovate on the product, and submit a beta.	Start of Semester 2, Week 4.
Obj. 5.1	Final test and evaluation.	
Obj. 6.1	Release, plans for release and maintenance.	End of Semester 2, Week 7.
Obj. 7.1	Final Report & Present the work.	End of Semester 2 Week 8.

1.4 Product Overview

The product overview of the project has been performed using Nesta (Nesta, 2021). Nesta is an innovation foundation charity that develops techniques and exercises that efficiently and rapidly identify plans for the project and tries to stretch it into a different identity. Nesta uses four key approaches before developing and designing a project: evidence planning, fast idea generation, problem definition and understanding the project's theory of change.

Nesta Evidence Planning:
Identifies what the aim of the project is and defines why it has value.

Key Focus of the 'Lemonade Stand' Project:

- Profit? Generally, the method is obvious once the idea has manifested.
- Develop a working and reliable application of a 'Lemonade Stand' game.
- A working and reliable virtual economy found within the game.

Enhance: What does 'Lemonade Stand' bring new value to?	**Replace:** What makes 'Lemonade Stand' less desirable?	**Re-use:** What does 'Lemonade Stand' build upon?	**Limit:** What could be the negative effect when pushed to extremes?
It provides people with a way to play with a virtual economy to be able to gain a sense of accomplishment and learning.	When a player can pay real money to enhance their gameplay or there are incentives to do so.	Other games with virtual economies; Cryptocurrency, virtual coin-breeding; Mobile application development.	Legal and ethical issues when real money is involved.

Nesta Fast Idea Generator:
A viable method of developing new ideas by thinking differently and opening new perspectives. Some of these methods and ideas will not be used but are effective in providing a better grasp on what this project could be.

Inversion: Turn common practice upside down.	Product placement found within a game is highly beneficial if it is done well and not seen as 'selling out'.

Extension: Extend the offer.	Players cannot directly purchase lemonade stock but trade a bond for in-game currency – that will be used for purchases.
Differentiation: Segment the offer.	The game implements bonds into the game economy.
Addition: Add a new element.	The game will be open to allowing 'real-world trading' or a method to trade the game items for monetary value.
Subtraction: Take something away.	Players cannot trade between themselves and therefore will have no virtual economy.
Translation: Translate a practice associated with another field.	When creating a virtual economy to use techniques to define game rules that will keep the economy running translates to Politics and Economics policies.
Grafting: Graft on an element of practice from another field.	Players can define the policies that will run the game. Therefore, the player will have an influence in the development and design by requesting or proposing features.
Exaggeration: Push something to its most extreme expression.	Player-run policies and economy (Player-driven development and gameplay). Players must pay real money for lemons or in game currency.

Problem Definition: Defines the main problems by exploring underlying factors, which allows a greater overview understanding of the product.	
What is the key issue you are trying to address and why is it important?	This solution is aimed towards solving problems of lack of entertainment and having a potential ability to create a virtual economy. Teaching economics and running a business to understand assets and money. Applies to all real-world applications and is the basis of how our life works.
Who is it a problem for?	This solution to the problem is aimed towards people who want to learn more about economics in a practical way and people who want to play a virtual economy-based game. Therefore, it is important to design for both parties so that the target audience can invest their time into this game.
What social/cultural factors shape this problem?	Ethics of creating a virtual-based economy and the money that is involved with it. Game addiction and mobile-game addiction. Furthermore, not properly teaching economics in a practical fashion or having many virtual economies.

What evidence do you have that this is worth the investment?	Mobile app development is saturated but calls for niche apps. A large population for the target audience means that the investment will turn out to be worth it and potentially profitable.
Can you think of this problem in a different way? And can you reframe it?	Creating a project that involves multiple features of computer science to show evidence of developing a performing product. Open-source code that involves the process and discussion of building a mobile multiplayer economy game.

Theory of Change (Key assumptions): Involves the future and defining the goals of the project and how to achieve them.	
What is the problem you are trying to solve?	Teaching economics and business creating a virtual economy business simulation mobile application game.
Who is your key audience?	People with phones and have time to invest in a virtual economy business simulator game.
What is your entry point to reaching your audience?	In this case, having an android mobile phone with internet. The time to play the game and the willingness to have a basic understanding of the game and economics.
What is the measurable effect of your work?	If the 'Lemonade Stand' product produced meets the objectives and performs properly.
What are the wider benefits of your work?	Players may be able to earn virtual wealth in a game.
Measurable effect?	Performance in each feature implementation.
Wider benefits?	Working and running game economy.
What is the long-term change you see as your goal? (Stakeholders)	Further implementation of features for profits and maintenance. Release and port to PC, iOS. Developers and investors will create this goal into a reality. Create a brand and develop more games. Players gets to have more content for their game.

1.4.1. Project Scope

The product overview of the application will involve a mobile-based, multiplayer economy, game based on 'Lemonade Stand'. The game will feature a virtual economy developed by an algorithm that will establish a supply and demand for the different items and commodities found in the game. The application system will involve a database implementation to be able to have a personal account system and multiplayer gameplay and trading. The game will be built in Android Studio (Google, 2021), using Java. Figure 1 showcases the concept art designed for the lemonade stand game and displays the essential states found within the game.

1.4.2. Target Audience

The target audience is aimed towards people who have time to invest into a simple game with very minimal input and time cost. These people enjoy long term 'grinding' (a term to describe long-term, repetitive gains) to earn benefits as they play the game and build their wealth. Furthermore, any person that wants to play a mobile application game and learn about virtual economies.

Figure 1: Concept Art for Lemonade Stand

Chapter 2: Background Review

This chapter will aim to review a formal background description of the project and explain its relevant topics. It will compare other existing approaches to the project such as similar game genres, virtual market system implementations and methods to the core game elements. Furthermore, the Summary of Literature section will perform a literature review of all the relevant topics and how it will be applied to this development of the project.

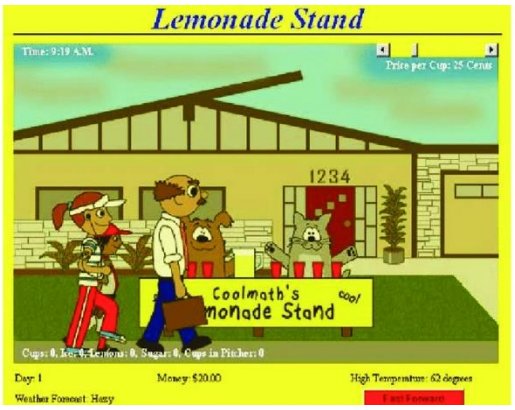
2.1. Summary of Existing Approaches

When creating a project, it is important to explore inspirations and state of the art solutions that already exist. The research documented will be essential and effective in helping to design and developing the final product. Being able to obtain the research would be achieved through a way of systematic searching. Systematic searching, defined as organising and performing the search process in a structured and pre-planned way. Comparing and critically analysing literature, software and papers that correspond to the main concepts in this project. Therefore, the search protocol must be defined precisely. Identifying a 'Table of Features' allows a clearer understanding of what is involved with the project.

Table of Features:	
1.	**Core 'Lemonade Stand' game systems**
1.a.	Recipe system – to change the recipe(s).
1.b.	Pricing system – to change the pricing.
1.c	Prep state – To start the game.
1.d	Sell state – To sell lemonade.
2.	**Player-account system**
2.a.	Register account system – to create an account.
2.b.	Secure encrypted account system – to secure player account and assets.
2.c.	Login account system – to allow it to work reliably and with a database.
3.	**Multiplayer game system**
3.a.	Choose rent place system – to choose a multiplayer rent space.
3.b.	See high scores system – to see all the players' high scores.
3.c.	Multiplayer sell state – to sell lemonade in the multiplayer state
4.	**Player-item commodity system**
5.	**Item-inventory management system**
5.a.	Item manipulation system – to breed different lemons and manage them.
6.	**Store-item stock system**
6.a.	Buy system – to buy different things from the shop.
6.b.	Item stock system – to have a stock timer system.
6.c.	Microtransactions system – to profit the game with real money.
7.	**Player to player market system**
7.a.	Buy offer system – to purchase items from other players on the market.
7.b.	Sell offer system to sell items to other players on the market.
7.c.	Player to player trading system – to trade with other players on the market.

There are games that already exist on the market and that are direct inspirations of this game. Firstly, the concept of 'Lemonade Stand' is a very common game that has seen many different interpretations over the decades of gaming development and how technology has evolved.

Figure 2: coolmathgames' Lemonade Stand



Shown in Figure 2, coolmathgames.com (Coolmathgames.com, 2021), a very popular practical Mathematics learning website, demonstrated their interpretations that are time-limited and very simple. However, the game features all the core elements found within a 'Lemonade Stand' game; containing; an inventory state, stock purchasing state, recipe state, selling state, and profits state.

Figure 3: EA's Lemonade Tycoon, Recipe & Pricing

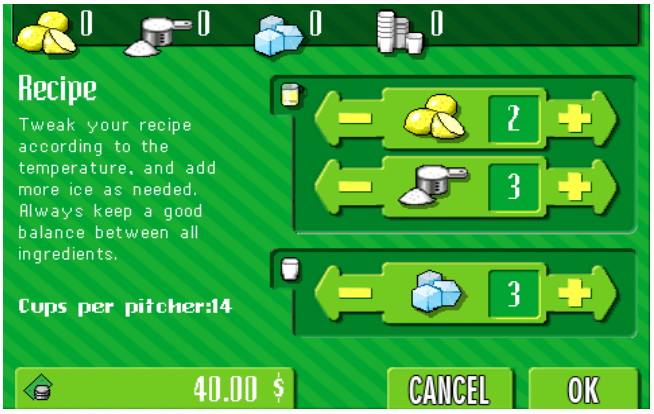


Figure 4: EA's Lemonade Tycoon, Selling

Other great interpretations of Lemonade Stand developed by Electronic Arts (EA) called, 'Lemonade Tycoon' (EA Mobile, 2002). This game's features were more advanced, designed very well and is more of the direction that will inspire how this interpretation of Lemonade Stand will turn out. The design and art of this game has a nice cliche design although it is very cluttered as information on the screen remained static and redundant after many playthroughs. Figures 3 and

4 display the essential and iconic game states found within the game.

Other business simulation game interpretations such as 'Roller Coaster Tycoon' (Chris Sawyer Productions, 1999) or generic Café or Restaurant games. In these sorts of games, the business involved is swapped for a different real-world commodity. Although sometimes, in the case for 'RollerCoaster Tycoon', the gameplay is different, but the objective is the same. Figures 5 displays 'RollerCoaster Tycoon'. Figure 6 displays Armor Games', Coffee Shop game (Armor Games, 2007).

Figure 5: Popular Unique Business Simulation Interpretation, Roller Coaster Tycoon

Figure 6: Café Krazy, Business Simulation

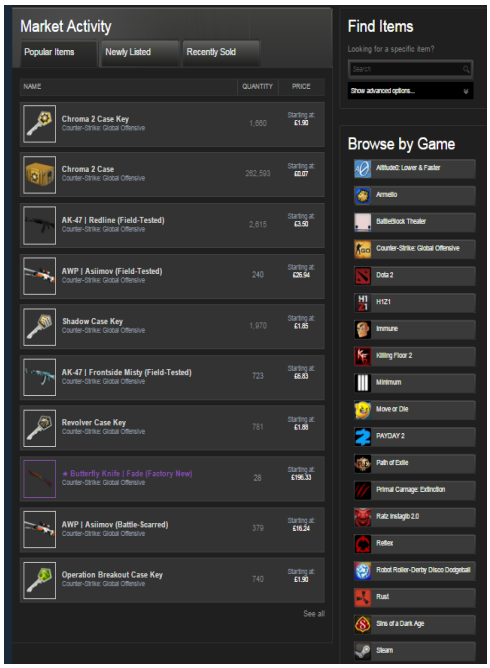
Games that feature virtual economies is a rare game genre to come by as normally, these games would be massive, multiplayer online (MMO) games and have very high scale production. One of the biggest inspirations is RuneScape (Jagex, 2001), which is one of the most popular virtual economies that is active to this day.

Figure 7: RuneScape Trading Interface

Furthermore, a whole system can embed a virtual economy within such as Steam (Valve, 2003). Steam is a video game digital distribution service by Valve and their games have unique, tradable virtual luxury items and people buy, sell and trade with these virtual items on the official steam community market or through a peer-to-peer trading system. These virtual items are

considered to be 'Non-Fungible Tokens' which will be discussed next.

Figure 8: Steam Community Market



A widespread bandwagon has introduced 'Non-Fungible Tokens' or NFTs to the world which is a digital certificate of ownership and authenticity. This market for "NFTs have boomed in the 2020 climbing to a market cap worth ~$338 million from $41 million in 2018" (Wall Street Journal, 2021). However, there are claims against NFTs regarding whether the value assigned to the asset is worth what it is on the market. Fungibility is defined as the ability of an asset to be exchanged or substituted with similar assets of the same value (Wall Street Journal, 2021). Non-Fungible assets are opposite in that they are defined to be unique and may not as easily be substituted for the similar asset. An example in the real world could be an original piece of art where any copy or imitation does not retain the same value as the only original. Cryptocurrency has been huge over the last decade. People, huge amounts of money and very influential companies have been involved with Crypto and born from it is 'CryptoKitties' (Dapper Labs, 2017). One of the first generation-based NFTs that breeds tokens into other tokens. This may inspire how the lemon token may retain a unique value based on the attributes of the unique attribute. The attribute is then able to be bred to create a more valuable token.

Figure 9: 'CryptoKitties' Breeding and Variations

2.2. Summary Of Literature

There are several papers and discussions that are involved in relation to this project as the game genre has proved to have an impact in the world, a virtual economy within a business simulation game. This involves discussion of the concept of 'real-world-trading' and 'gold farming[1]'. Furthermore, the discussion involves correlating topics that are not normally related to mobile application development that causes the development to have a specific approach to its design, 'Mobile game addiction' and 'Designing for mobile game-based learning'. Furthermore, the ethics and laws behind any policies against this genre, 'Legal disputes and legislation policies. The search terms that were included when researching this project: Android, Java, Game development, Project Management, Java Database Server Implementation, Lemonade stand. Many of the sources of these papers and discussions are provided by scholar databases and research literature on from literature from the Institute of Electrical and Electronic Engineers (IEEE.org, 2021), Association of Computing Machinery (ACM.org, 2021), Social Science Research Network (SSRN.org, 2021). It is important to note that scope of technology during the time some of the papers discussing can be different as huge changes have been made over the last decades, however some points are relevant.

This project revolves around the genre of a business simulation and virtual economies so it is important to define the understanding so that we can discuss the impacts and how to create it. As discussed, the main economic aspects that will be discussed and relevant to this project are scarcity, supply & demand, cost, benefits & incentives (Beattie, 2021). Scarcity is defined as to the effect of being in short supply or shortage. This feature will directly be implemented by limiting how much available lemonade stock is in the game and therefore other players will want stock from other players which directly creates the supply and demand. Some players may have an abundance of the basic lemon at a very cheap price and players may purchase or trade from them as it is cheaper and may return higher profits, which is what benefits & incentives are about.

[1] gold-farming - described as the practice of playing the game intensely to amass assets and wealth within the game's virtual currency or any other valuable items found in the game, where the player can trade or sell the item/currency for real money.

In business simulations, the stock is generally unlimited as there are no other players to compete with and the game is not designed in that way. By making the stock scarce, the virtual economy can exist. The paper, 'On Virtual Economies' discusses that developers when a virtual economy is established, should and must use government economic policies to their game economy so that it can retain its game value without being over-inflated as the game lifetime goes on. "And in cyberspace, the coding authority does indeed have the power to create and destroy any good, at virtually zero cost" (Castronova, 2002. p4). Government intervention, in economics, are actions taken by the government to regulate or manage the decisions in an economy about social and economic matters. This may include taxing the people or giving subsidies to different sectors to help benefit the economy as a whole. In this scope of the project, certain policies that have already been implemented to retrain from inflation is the feature of 'Renting' a space in the multiplayer area. This may become increasing competitive and costly and may define its own floors. Other features that may be included is taxes or (when changing server) certain stock becomes more scarce and therefore more demanding and costly in that server market. This is to simulate how moving from different countries causes different ingredients to be available at a time and sometimes more expensive. In terms of business, once the cost is increased, the profit may be decreased significantly. This implementation might be imperative to ensure it is reliable and works because, it may be the determining factor if the game were to be successful and released. Otherwise, it causes players to reach the wealth limit very easily and therefore limiting what potential the game would have as the worth of the game items may lose its value. Castronova describes how value in a game retains a worth, "Given the choice between a puzzle that is mildly challenging (putting together a 100-piece puzzle) and one that is not (put together a two-piece puzzle) …" (Castronova, 2002. p17), and why the game exists where if this game is too easy to reach the wealth limit, there is no fun and there would be no willingness to play the game at all.

Moreover, on virtual economies, it is important to discuss the aspect of selling the stock and how the player makes the money. 'Development Informatics' (Heeks, 2008) uncovers the dramatic and controversial impact of the externalities of virtual economies,

being the concept of ‘gold-farming’. Gold-farming is defined as the practice of playing the game intensely to amass assets and wealth within the game’s virtual currency or any other valuable items found in the game, where the player can trade or sell the item/currency for real money (Oxfordlearnersdictionaries.com, 2021). Th earliest case of a gold farming example was during the 1987 where cash payments were made for virtual items in a text-based basic graphics multi-user dungeon game (Heeks 2008). The issue regarding this practice is for the users that end up spending unethical number of hours in order to make a wage and it is seen as immoral by the developers and any stakeholders to the ‘farmer’. It has become a real-world issue where a voice in the population of Venezuela have been playing RuneScape to survive in such a terrible and corrupt economy (Ombler, 2020).

‘Design User Experience for Mobile Game-Based Learning’ (Shiratuddin and Zaibon, 2011), layouts the fundamental design for mobile-based learning. The theories to approach the most efficient way of teaching mobile-based learning and the methods to apply them. This paper will be effective when applying in requirements modelling as it describes the needs and how the design of the application should be. Although, many of the features it uses to describes on how to the user focused and engaged. It also discusses that it should work with the pleasure reward system and combine that with confirming with the user are learning. It also discusses that research on this learning and design is limited. ‘Computers in Human Behaviour’ (Balakrishnan and Griffiths, 2018), is a paper on Mobile phone addiction and addictive design. It discusses the methods on how mobile phone game applications use addictive design to generate profits and keep you playing. It uses the human reward system by awarding pointless digital points for keeping virtual tasks. This paper is an important discussion in today aspect of mobile phone application design as designing for and against these features difficult approaches. It calls upon ethics, the affects loyalty between the customer and business. This study will help evaluating the design methods used to make the application’s profits, ‘fun’ and manipulative techniques. It slightly goes against how learning-based game is designed as keeping the user engaged may be interpreted as addictive features in games.

Chapter 3: Technical Progress

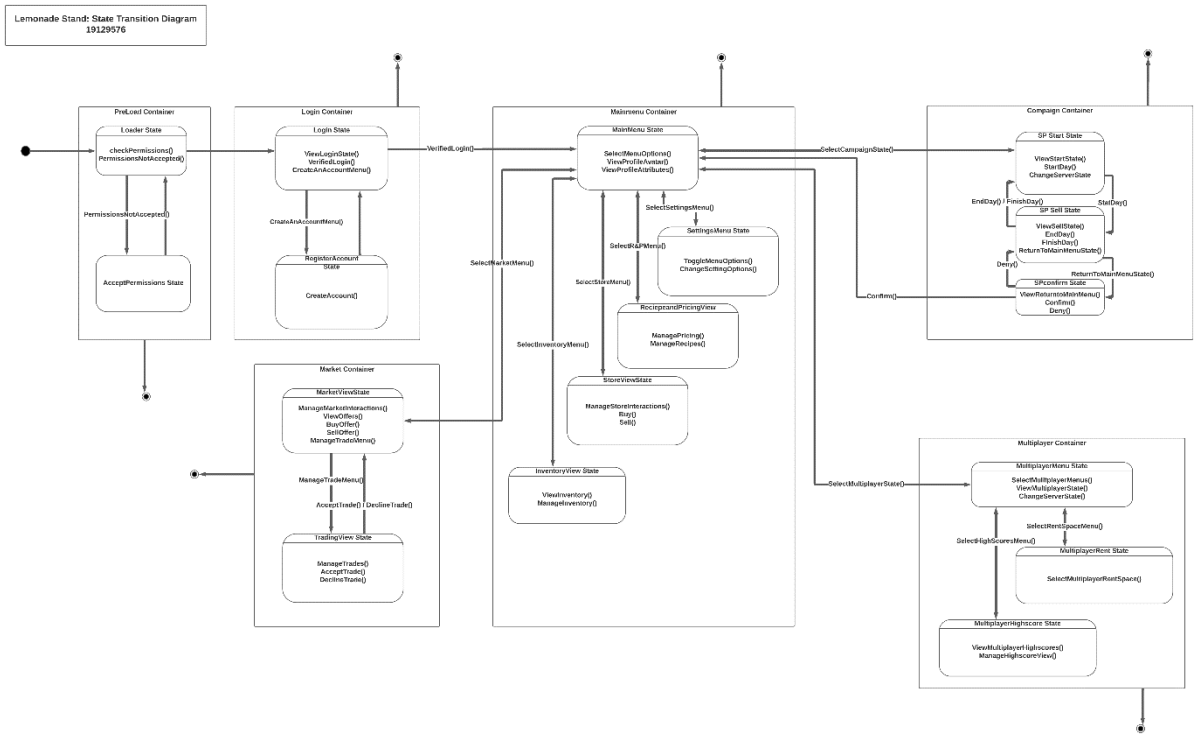
3.1. Technical implementation

The technical progress aims to describe the timeline of the project since it has been developed from its initial stage, from the proposal and progress report. It will be described using the standard methodology, Unified Modelling Language (UML), to show that designs have been abstracted and tested to provide the functionality of the project. Furthermore, a critical analysis of the core components being tested with the diagrams to show intentions and convey the understanding. The elements that make up the project will be illustrated that will represent what objectives and the minimum viable product accomplishes. Moreover, the issues that arisen which led to certain project downfalls but with documented and considerate analysis on what would be improved.

3.1.1. State Transition Diagram

The State transition diagram is an effective method of communicating how the 'Lemonade Stand' client intends to flow between each of its states to provide the different functionalities of the application. It describes each of the states that corresponds to the relevant container to easily describe what the functionality type of the states.

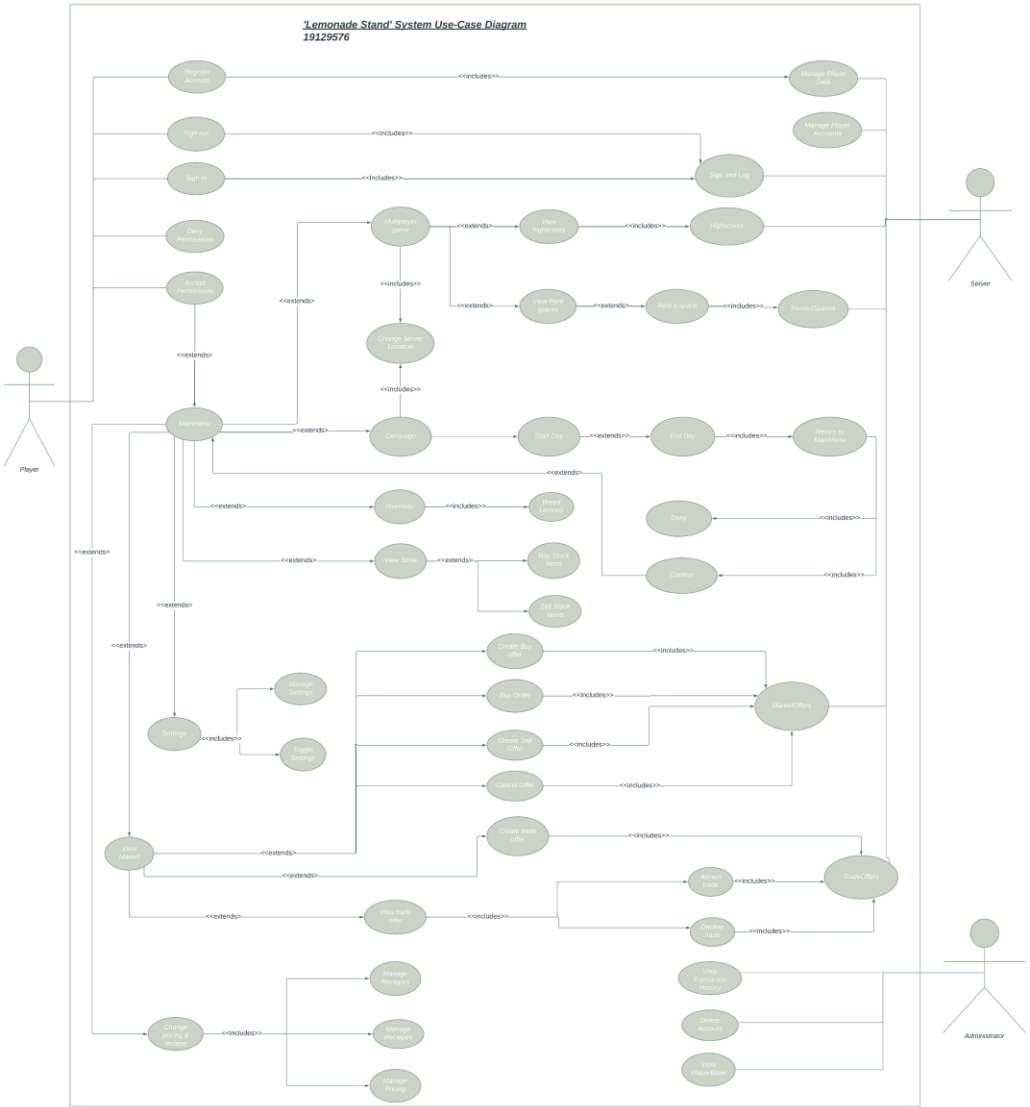
Figure 10: State Transition Model Diagram



3.1.2. Use-Case Diagram

The Use-Case Diagram is an effective method of defining what an actor can do in the system. In this iteration of the diagram, there are 3 actors that can interact with the system: ‘Player’ actor is the core actor that interacts with the system with several actions. The main action step begins with the ‘Player’ ‘accepting application permissions’ and ‘logging in’ to enable usage to the main part of the system. Once logged in, the ‘Player’ is able to interact with the ‘MainMenu’ to allow the ‘Player’ to interact with the different features of the system, such as: the ‘Campaign mode’ or the ‘Store’. ‘Server’ actor is a representation of what the services the database and server provide for ‘Player’ or ‘Administrator’ when they interact with the system. ‘Administrator’ actor is a system management actor that can view the inner details of a system or manage ‘Player’ accounts.

Figure 11: Use-case Model Diagram



3.1.3. Structural Model (Class Diagram)

The Structural Model, or Class diagram describes the different classes found within the system and how they relate and interact. The classes outline the different attributes and operations to allow a better view on what the class can achieve. The two different class diagrams are representing the two different system components in the project, the server system, and the client system.

Figure 12: Server Class Diagram

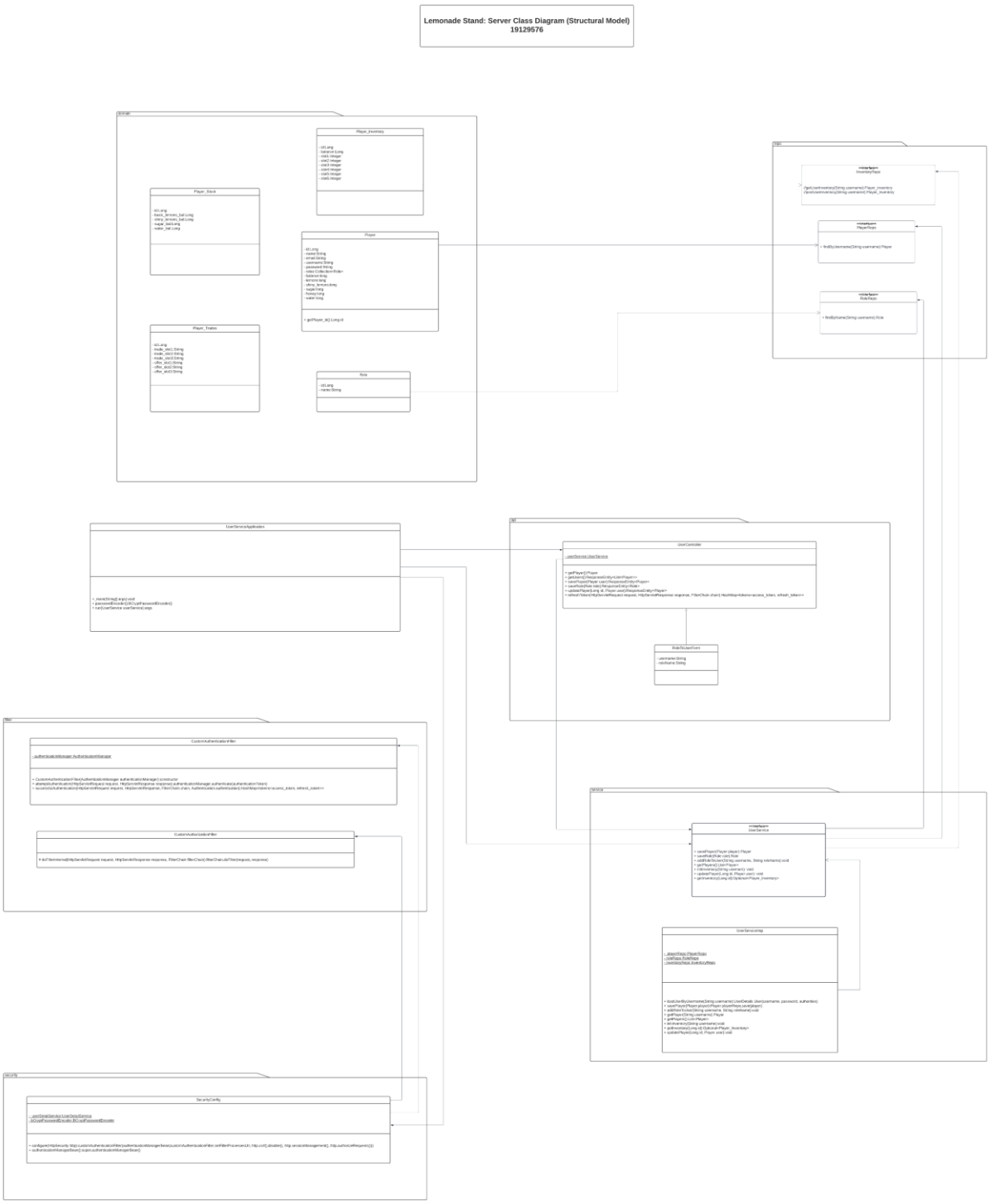
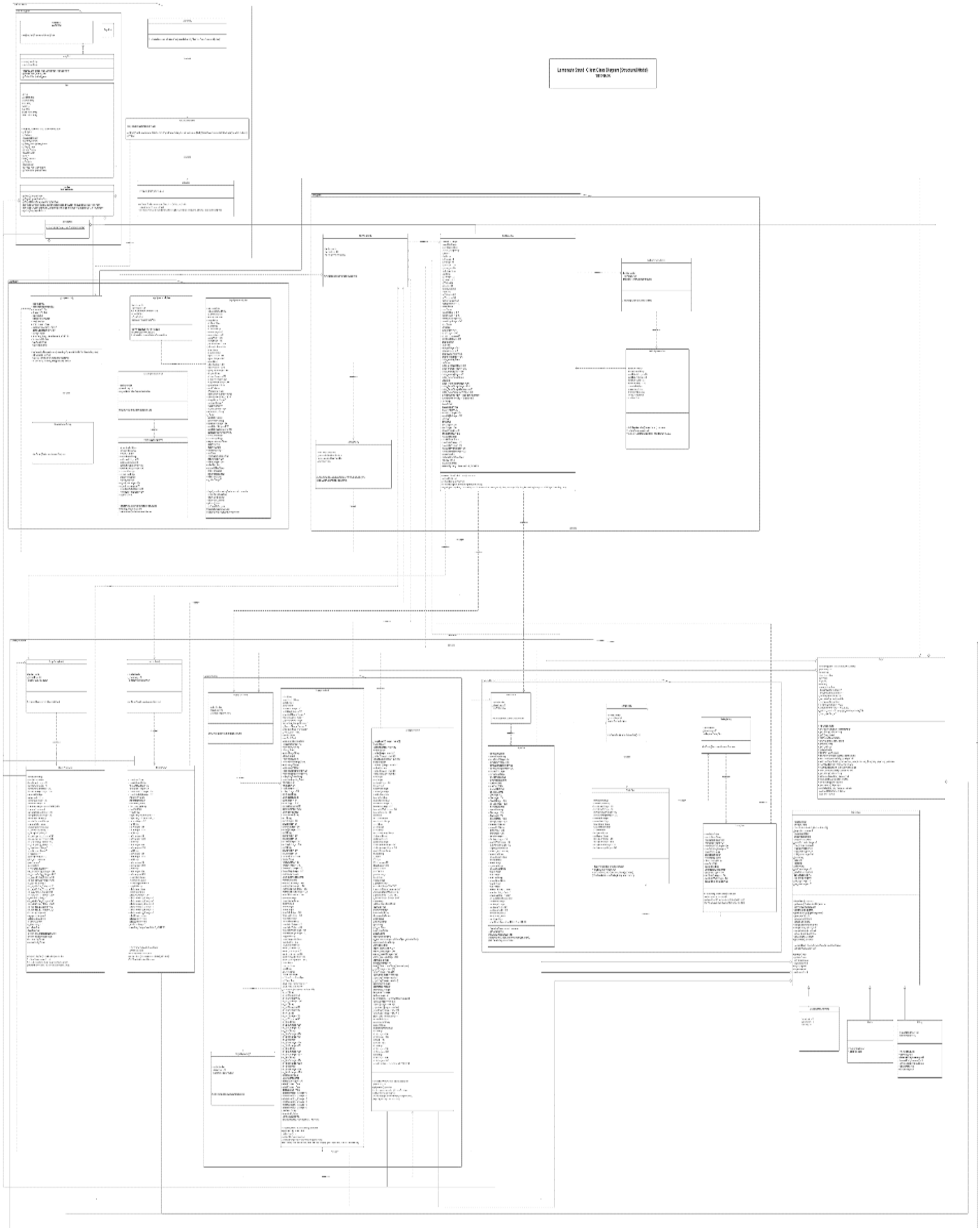


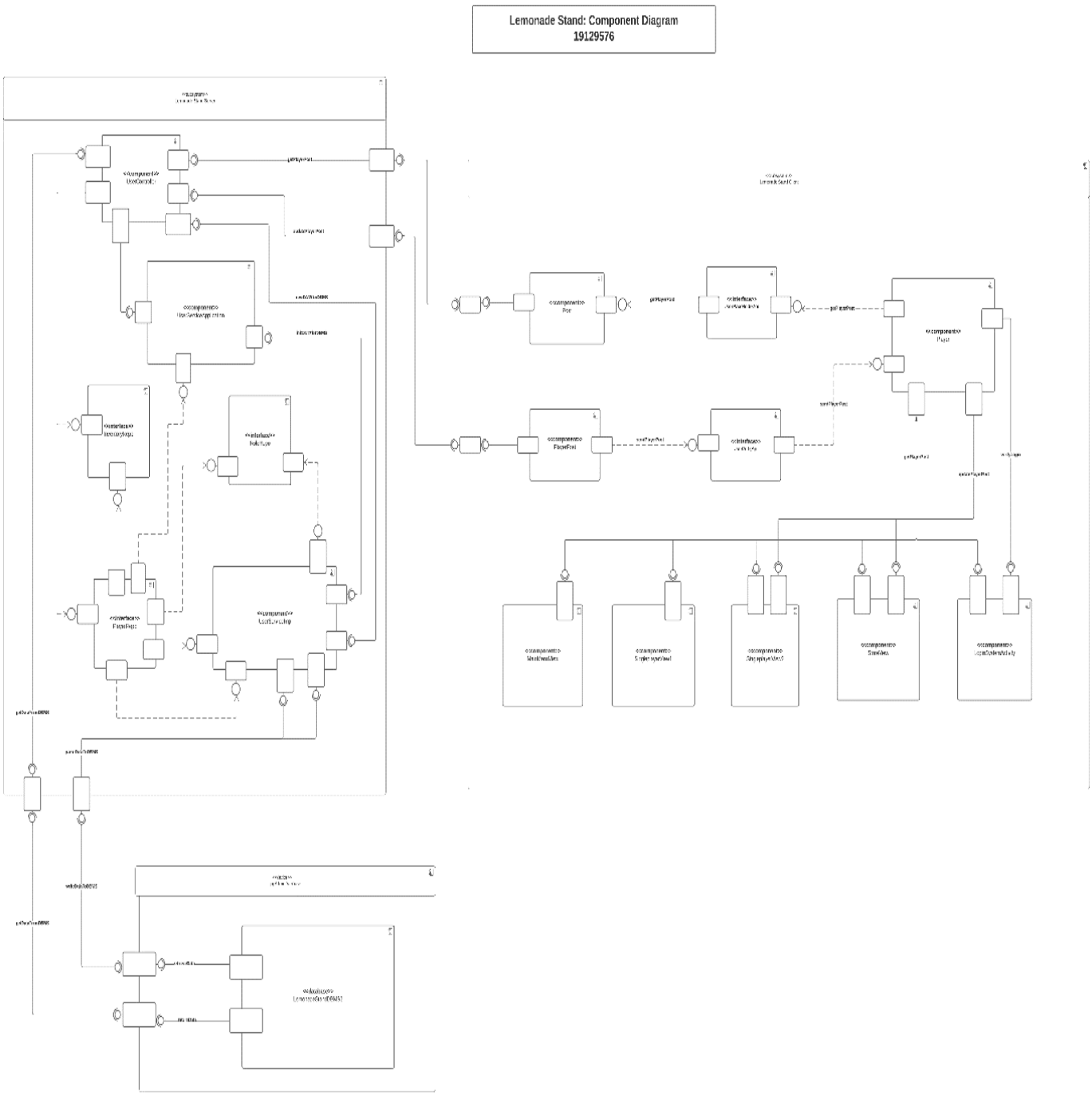
Figure 13: Client Class Diagram



3.1.4. Component Diagram

The Component diagram describes how all the systems will provide the functionality of the project. This includes how the application systems uses the server system component to send and receive data from the database and then allow the client system to communicate with the server to function and provide the given service.

Figure 14: System Component Model Diagram



3.2. Methodology

All tasks and processes in developing this project will require practicing heavy formal documentation, planning, and designing in hopes to describe the overall timeline of the project and how it is made. The meetings with the Supervisor will be planned and logged.

Storage for this project will be held responsible by Google Drive (Google, 2012), GitHub (Friedman, 2008), and main physical drives will hold the project artefacts, source codes, and UML documents.

The repositories contain all the source code for the server and client, furthermore, it contains the relevant artefacts, reports in PDF format, and UML diagrams in SVG format. Moreover, enabling version control management, an important aspect in project management for developing software.[2]

3.2.1 Approach

The approach to smoothly developing this 'Lemonade Stand' application is done by applying high quality software engineering techniques to allow uncertainties in the project to be limited and known. This involves designing proper UML design artefacts that will define the requirements criteria to understand the scope and what the end-product will be. This project will aim to follow the methodologies of Agile software development as it ensures that the development of the project is acerated by being pushed gradually and flows naturally with priority-driven development. As mentioned, several UML artefacts have been built and designed where the requirements of the 'Lemonade Stand' application have been properly defined. While designing the documents and artefacts, constant development corresponding with the artefacts will be done to ensure that features are implemented on-time and with a standard mechanism. Once development has reached a state of being a 'minimum viable product', the project can undergo testing to ensure that the project reaches the requirements of the UML blueprints. Afterwards, the project will go into evaluation and discuss needed development if the project were to be deployed or feature implementations.

The process of following an agile development is understanding the process of a 'Sprint'. A 'Sprint' is a two-week development process where the

[2] Web links and references found in 'Bibliography and References', Chapter 6.

project state undergoes heavy analysis and evaluation on how to develop it further. This is achieved by using artefacts such as 'User-stories' or 'Storyboards' to develop priority-driven tasks. Tasks that are allocated on to the backlog may also be determined through a feature-driven development where a certain core feature is developed such that the system works better or modularly. Generally, when building games, focusing on developing the core methods of the game rather than improving the quality aspects of the application or system.

3.2.2. Technology

Developing this project will be achieved using Java, in Android Studio using Android APIs. This project would connect to a server using the Spring Boot Framework (Spring.io, 2002) built in Java using IDEA (IntelliJ, 2001). This framework provides feasible database networking implementation to be enabled. The database management system provided by pgAdmin 4 (Page, 1996) will allow the server to interface with client so that the multiplayer and player accounts functionality works.

The project will be developed on a Windows 10 (Microsoft, 2015) computer with a 7th generation i7 processor with a graphics card Nvidia GTX 1050, laptop version (Nvidia, 1993). The test development environment will be achieved on a 10th generation i7 Processor with Intel Iris Plus Graphics (Intel, 1968).

The computer systems will be emulating an Android operating system environment to run the game application. The Android Emulator is running a virtual device, Pixel 5 API 28, with 4 cores and requires at least 2 gigabytes of RAM. A computer that will be used to run the database and server would be sufficient with 2 to 4 gigabytes of RAM. The smartphone requirements will include any smartphone made within the recent five years or running Android 9.0 or later.

3.2.3. Version control management

Version control management is an important aspect when ensuring a safe and high-quality application development in any project. It begins with ensuring backups and version management of the project in all stages of development. This creates a clear understanding of the different repositories and the development timeline. Version control for this project will be maintained using GitHub (Friedman, 2008), a popular software and project management tool. Cloud storage and physical storage management is also very important. All artefacts and documents will also be shared on a Google Drive folder (Google, 2012) and backed up to physical drives. This allows sharing documents between important project stakeholders and developers and the security that the project will not be lost.

3.3. Project Plan

Project management is formally understanding and defining how to achieve progress within the project. Conveying the objectives into small tasks, that needs to be completed for the objective to be achieved. It also allocates the efforts for the tasks and layouts the deadlines for the project. It achieves to convey what plans to be developed, what was originally planned and what has been completed. Potentially, it will also cover the event that a task is not completed on time.

3.3.1. Objectives Described as Activities

Objective 1 - 2: Ethics forms & Project Proposal

A1.1	Background Review and Analysis of existing or similar approaches
A1.2	Review Ethics and legal aspects for this project.
A1.3	Develop Project Proposal.
A2.1	Prototyping the product.
A2.2	UML design, planning and development
A2.3	Review and reformat due dates to complete the project.

Objective 3: Designing & Development and Progress Report

A3.1	Develop a minimum viable product application.
A3.2	Design a 'Progress Report'.
A3.3	Develop Server code and attempt database implementation.
A3.4	Design and develop an implementation of database servers.
A3.5	Design State transition structure
A3.6	Design UML documents and artefacts
A3.7	Create Software Artefacts and User Stories
A3.8	Define formal dates, goals and back log.
A3.9	Submit a progress report

Objective 4: Product Post-development

A3.1	Create a greater-minimum viable product application.
A3.2	Design the interface of the Java code with the servers.
A3.3	Develop database code and implement it.
A3.4	Develop an algorithm for the virtual economy and implement it.
A3.5	Iterate features, innovate on the product.
A3.6	Create a shippable viable application or alpha.
A3.7	Start documenting the final report.

Objective 5: Testing Analysis and Evaluation

A4.1	Test the application and analyse.
A4.2	Iterate features or innovate the product
A4.3	Submit a beta or working product.
A4.4	Evaluate the progression and timeline.

Objective 6-7: End of the Assignment

A5.1	See evaluation and review.
A5.2	See plans for release and maintenance.
A5.3	Final report, video and poster.
A5.4	Present the work.

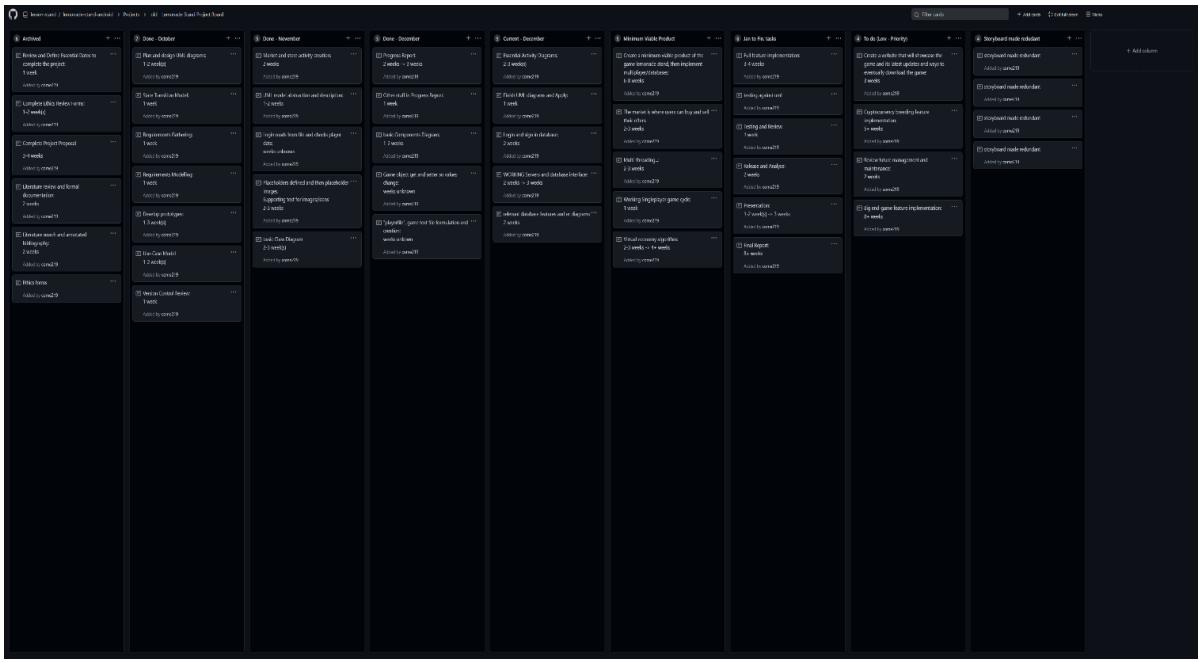
3.3.2. Schedule

An important method in ensuring that the project progresses is building a timetable with due dates. GitHub offers an effective project storyboard, which this project will be using to describe the tasks and effort needed for the project. Moreover, all the storyboards will be put into the relevant repositories with the links provided.

The project storyboard underwent through several iterations as the perspectives of the project changed. The project has three iterations of the storyboard, the first storyboard describes the overview and aims when the project first began. The second storyboard represents the storyboard that was submitted for this dissertation task. Finally, the last storyboard includes any further developments that were added to the project but were not feasible for the given time or appropriate.[3]

Totalling Time allocation:	
October:	4 to 5 Weeks.
November:	6 to 9 Weeks.
December:	12 to 15+ Weeks.
January to Finish:	7 to 11+ Weeks.

Figure 15: Initial Iteration Storyboard



[3] Several tasks can be done in the same week and may overlap each other and not be true to when the task is started. The tasks may also be described in an exaggerated state as it describes the difficulty of the task when it is acceptably completed.

Figure 16: Dissertation Submission Storyboard

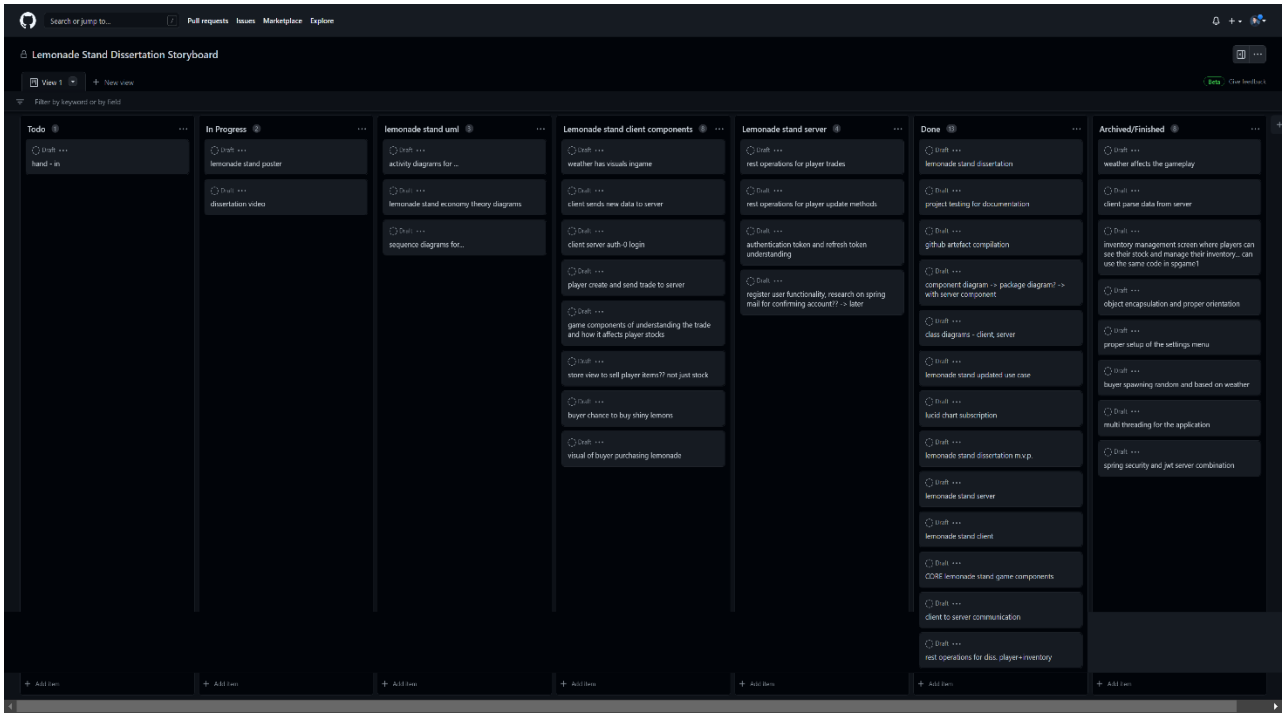
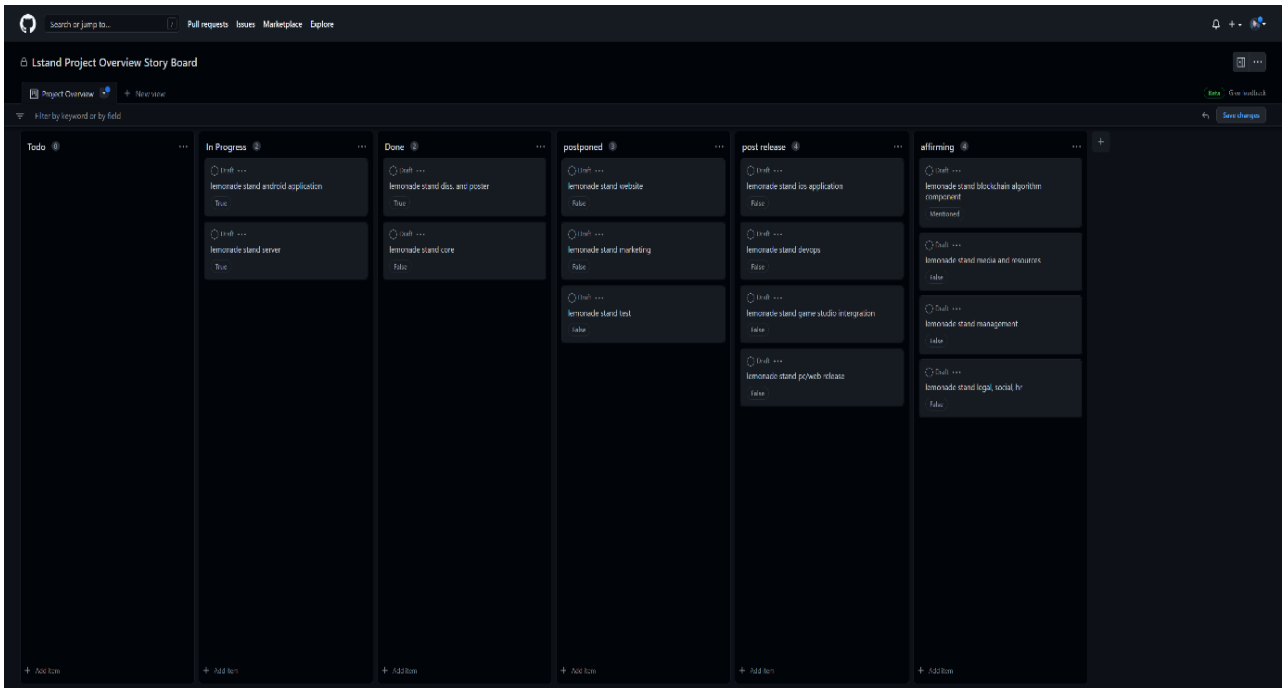


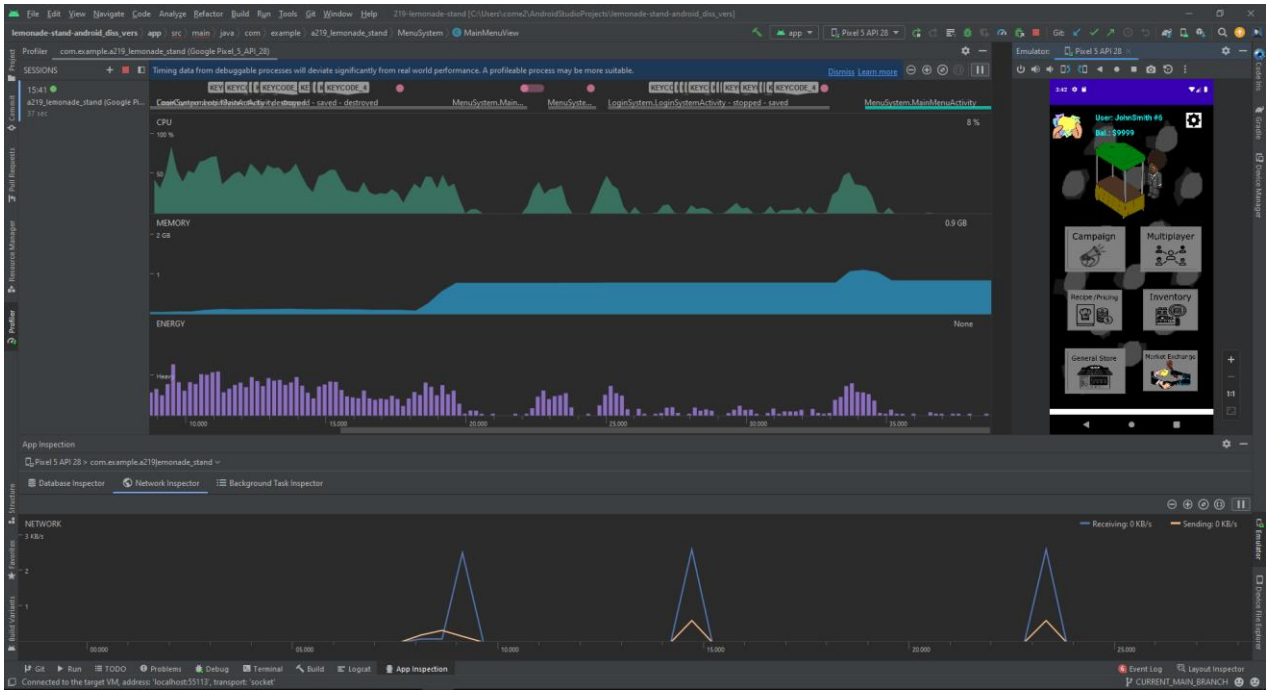
Figure 17: Future Implementation Storyboard



3.4. System Requirements

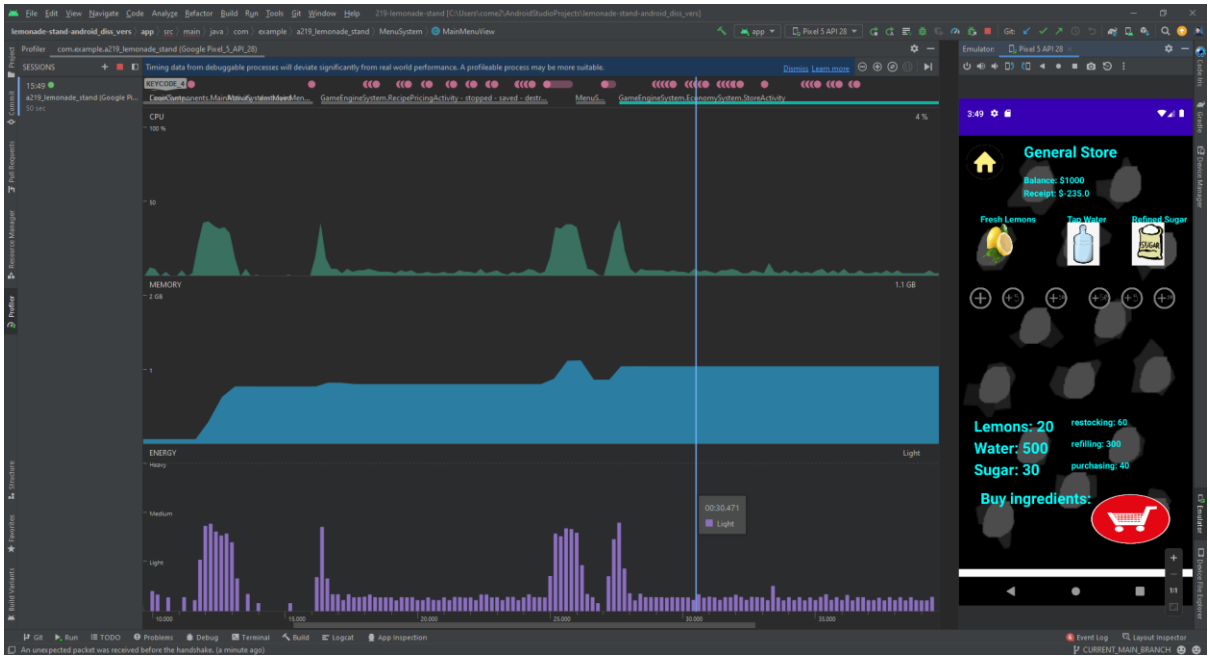
Defining the entire application system requirements at this current stage is difficult be able to be defined and predicted. The Android Studio IDE has a resource usage feature that allows to test the application while being simulated and see the values of usage. The ‘Profiler’ can also measure the performance on the physical device. The simulated system will be tested with each scenario: logging into and out of the ‘MainMenu’, selecting each core features: buying stock, managing pricing, and the different gameplay states. The following figures describes the resource usage of each scenario using the ‘Profiler’:

Figure 18: Scenario: User Logging In and Logging Out



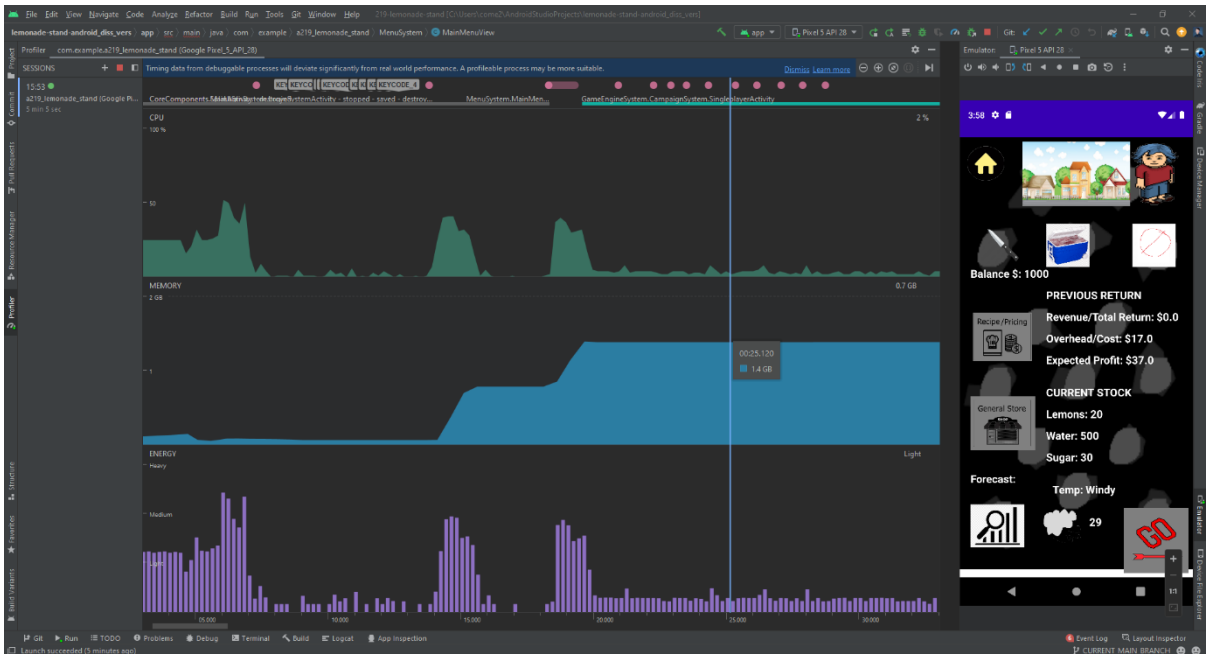
In this scenario, it describes the ‘Bob’ player logging in and out and then the ‘JohnSmith’ player logging in. It also depicts the states of when the networking components are being used and the usages of the resources during the various states.

Figure 19: Scenario, Logging In and Using the Recipe and Pricing State



In this scenario, the user logged in and used the various features and menus in the game. As shown in the diagram, the usage does not go beyond 1 gigabytes of RAM. However, tends to be fluctuating when interacting with the features or switching activities.

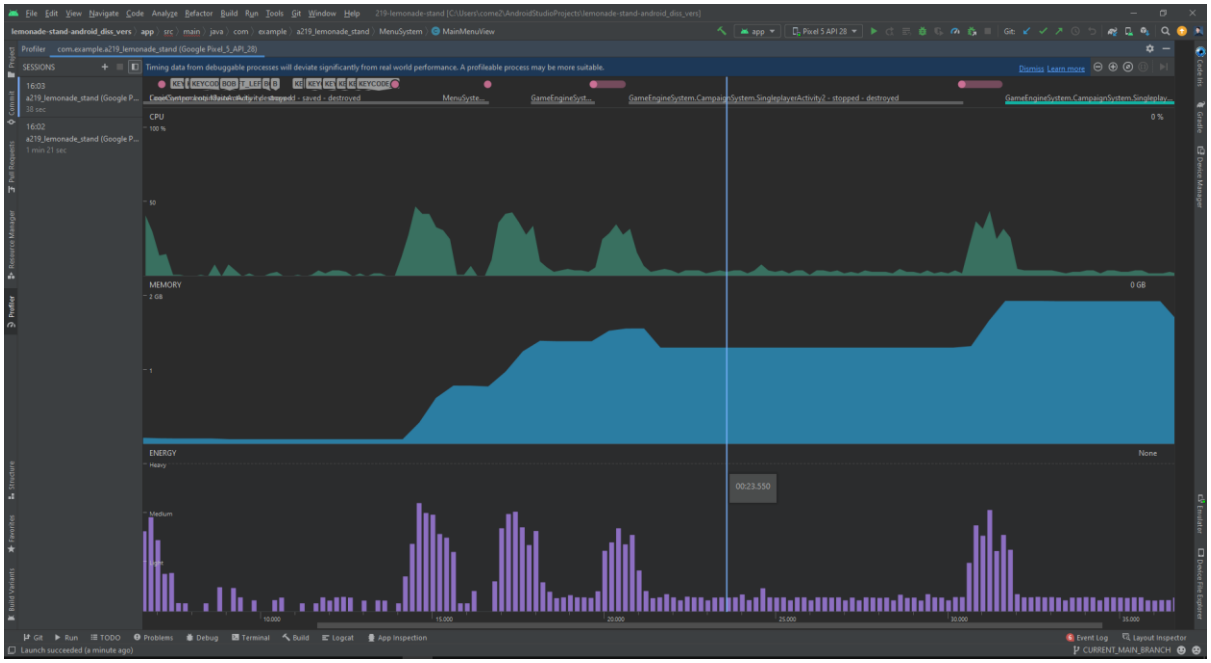
Figure 20: Logging In and Using Campaign View 1 State



In this scenario, the user logged into the player ‘Bob’ and used the various features found in the first game state. As shown in the diagram, when entering and interacting with the different components, the usage goes up towards 1.4 gigabytes of

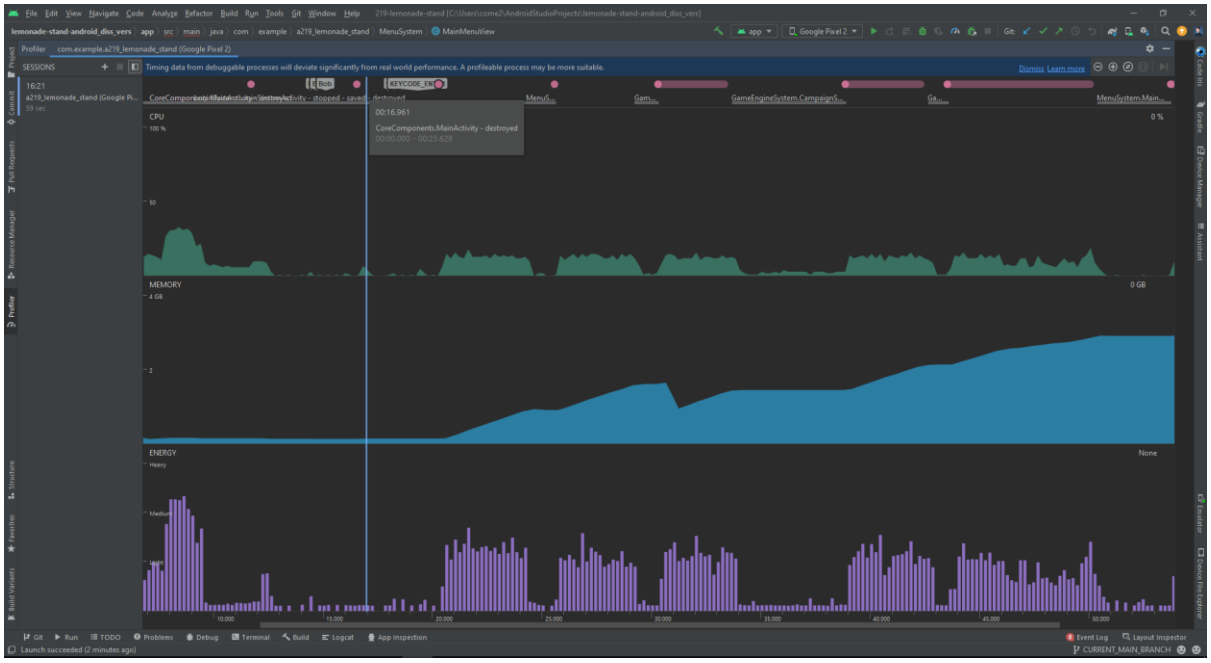
RAM, which is expected as the game states are more intensive in graphical and resource components.

Figure 21: Logging In and Using Campaign View 2 Cycle State



In this scenario, the user went through the game cycle of going between the game states. The trend found in the previous scenario is found here. However, upon switching into and out the most graphic intensive view requires around 2 gigabytes of RAM.

Figure 22: Profiling A Physical Device



In this scenario, the physical device was measured using the ‘Profiler’. As shown in the diagram, a general expectation of increase usage of RAM is described. However, the trend shown is shown to be on an upline and goes beyond the RAM usage of 2.5 gigabytes.

The different components in the figures describe the different types of resources the virtual device is using. As shown in the figure, RAM usage increases as it goes through the different scenes. However, it maintains around 1.5 gigabyte of RAM and may only need to be required to up to 2 gigabytes, but it would be recommended and required to have at least 3 gigabytes which is quite standard. As the system moves through the more demanding scenes, more of the CPU gets used but only up to 50%, which is to be expected as there are not that many demanding resources in the game. However, once multiplayer and networking are used in the system, it requires a bigger portion of the CPU and resources to sustain a reliable usage of the application but is still not demanding compared to drawing 3D graphics or using complex animations.

Finally, it seems that, by design, the load and high resource allocation is at the start, that is due to loading the application and its features. The other parts of where the system seems to use a lot of energy is when it changes between the different screens. This is probably due to the poor implementation of the threading between activities or efficiency of how the different activities interact with each other is poor.

3.5. Test Planning and Testing

This section describes what is implemented with the client and the server. What was developed for the task and the issues that exist and it describes the features that are not included in the project.

3.5.1. Features Not Included

This section describes the features that were developed for the project but ended up being excluded from the final submission as it conflicted with the code and ran into limiting feature issues. These features were tasks for a ‘Sprint’ that ended up being neglected and solved with a different method for the dissertation. To convey these features, a useful application that was used in development and testing, Postman (Postman, 2022), API platform.

The major feature that was not implemented is utilising a web server security standard where the server would inherently contain role-based integration, authentication and authorization filters and other privacy network security standards. Role-based integration would allow users to log into a ‘admin - player’ that may have extra features describe in the ‘Use-case Diagram’ and permit a user to corresponding role features. The authentication and authorization filter allows specified data to be hidden from the network and allows a user to only communicate with proper access and refresh tokens provided by the service. Without this implementation, the current method uses ‘GET’ methods to return data and is not able to ‘POST’ new data as too much time went into implementing a more difficult ‘POST’ for the filters and tokens.

Figure 23: Uncommented Security Configuration Feature

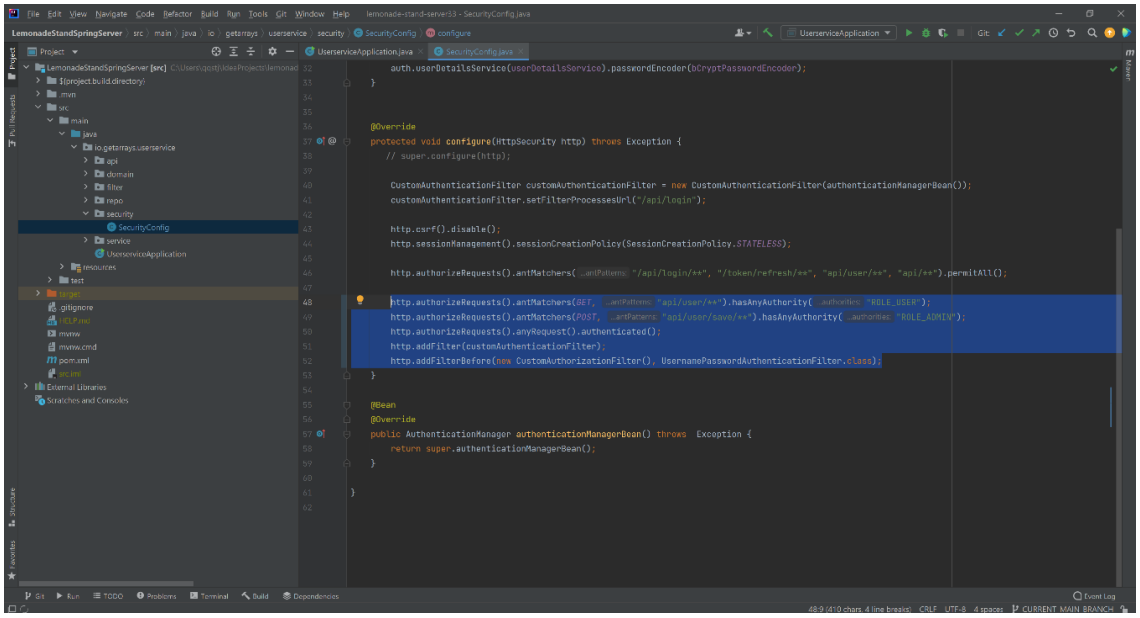


Figure 23 describes the security config section that is commented out in the original project but was developed for the task. When uncommented, it implemented proper spring web security standards. However, it required the client to communicate differently to the server and proved to be difficult.

Figure 24: Authentication Logging In With Authorization And Refresh Tokens

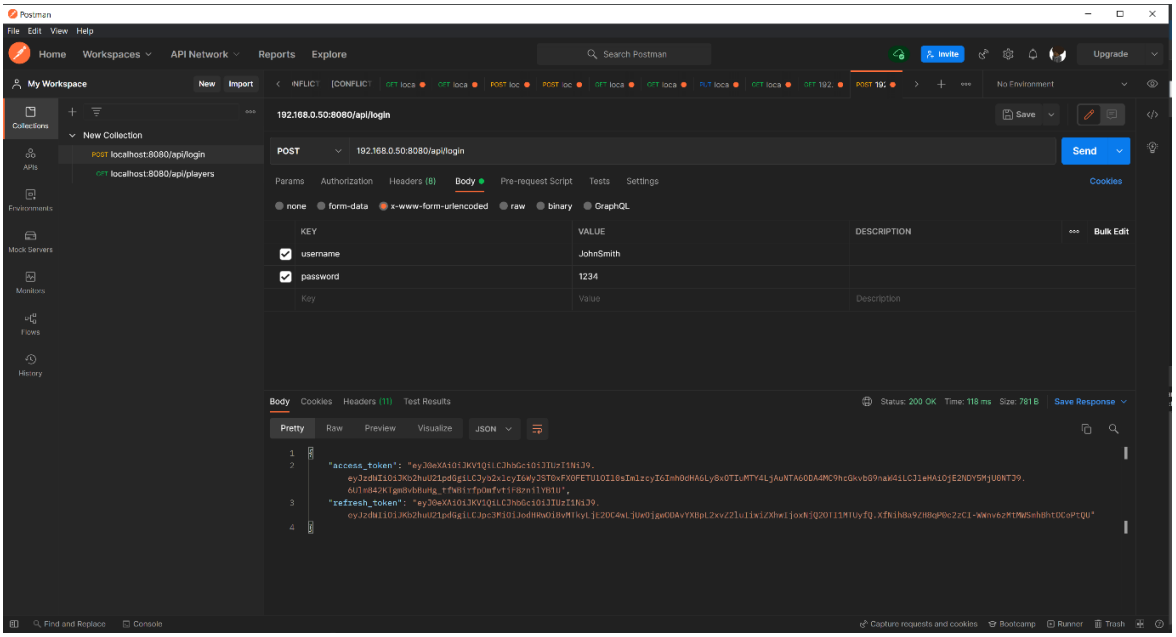
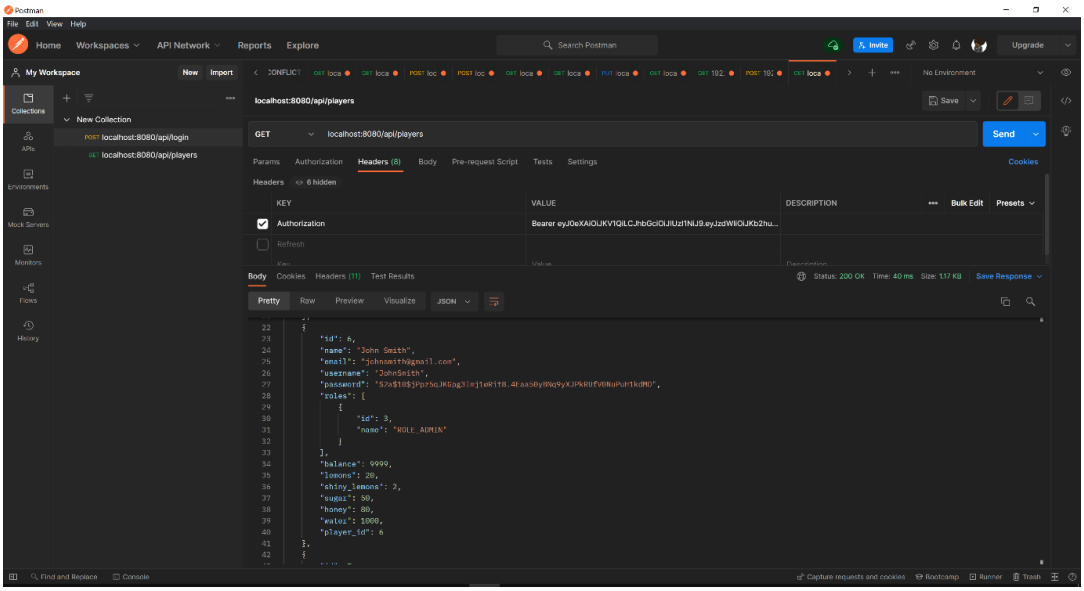


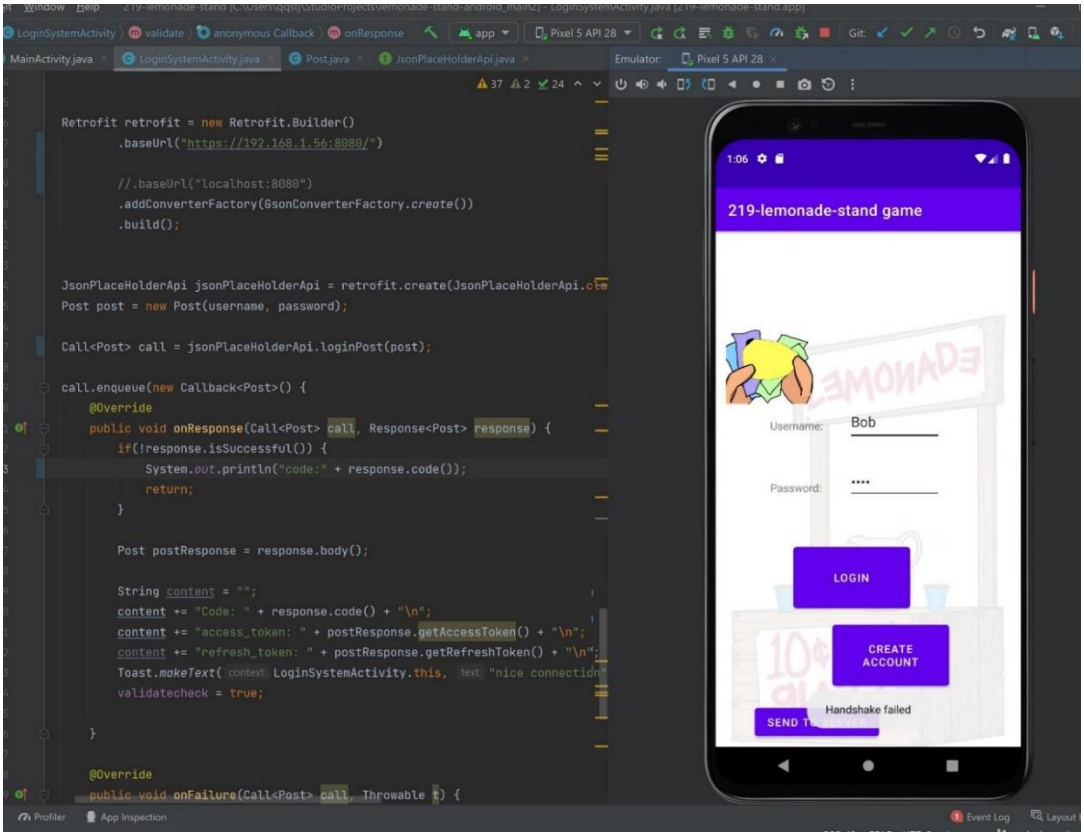
Figure 24, 'Postman' is showcasing the usage of the authentication and authorization filter and the data returned that allows the given user to interact with the system. As shown, the user 'JohnSmith' logged into the client and retrieved the access and refresh token from the server.

Figure 25:Retrieving Data Using Authentication Tokens



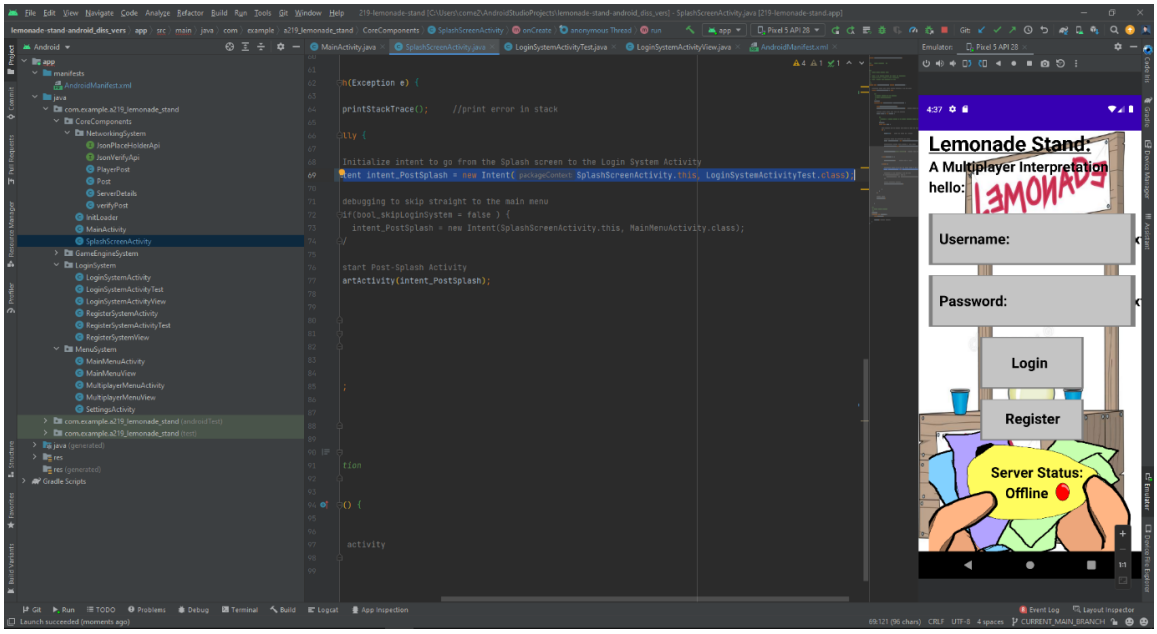
In figure 25, the access token key is used as a parameter with 'Bearer ' to retrieve and communicate with the server securely.

Figure 26: 'Bad Handshake' Error When Trying To Implement Login Authorization



This figure showcases the implementation attempt of the authorization mechanism to communicate with the server in the game client. However, I ran into issues with 'Handshake failed' and could not finish completely in time which is why the current implementation retrieves and communicates with 'GET'.

Figure 27: Alternate Login View Developed Programmatically



This figure describes the development for the ‘login screen - version 2’ which ran into issues with defining a keyboard programmatically verses how it is achieved using android activity layout method. A task that seemed simple yet took about a week to understand it was going to take another week to solve. Therefore, uses an older, less impressive method.

3.5.2. System Testing

This section describes testing the system for any feature errors, bugs or implementation. Generally, the system works as defined by the UML blueprints and is able to perform the given task to a minimum viable state. However, some of the values aren’t parsing properly this is probably due to how quickly each feature of the project was developed. This leads to the project crashing due to improper implementation of android code/inefficient activity handling and view construction.

Figure 28:Touchscreen Input System Issue

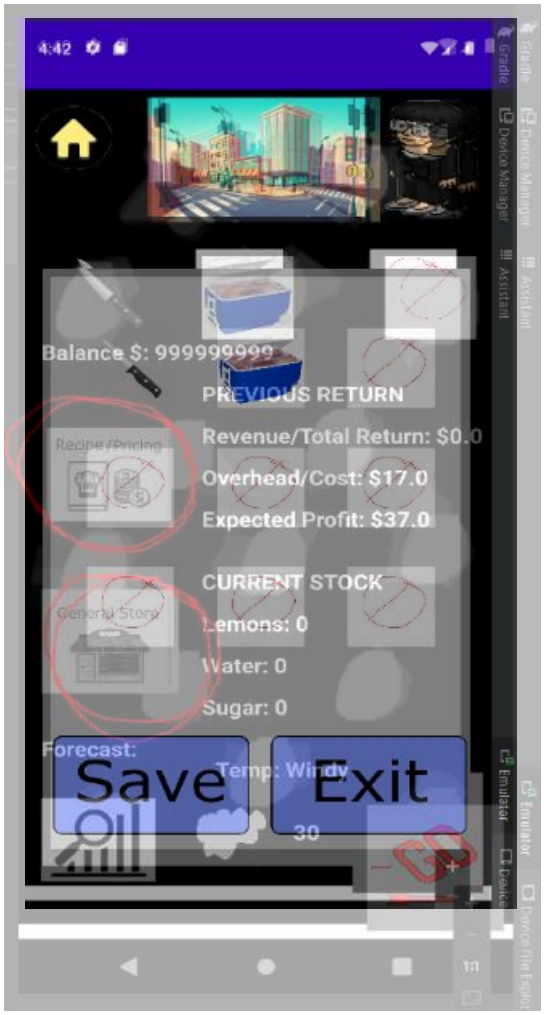
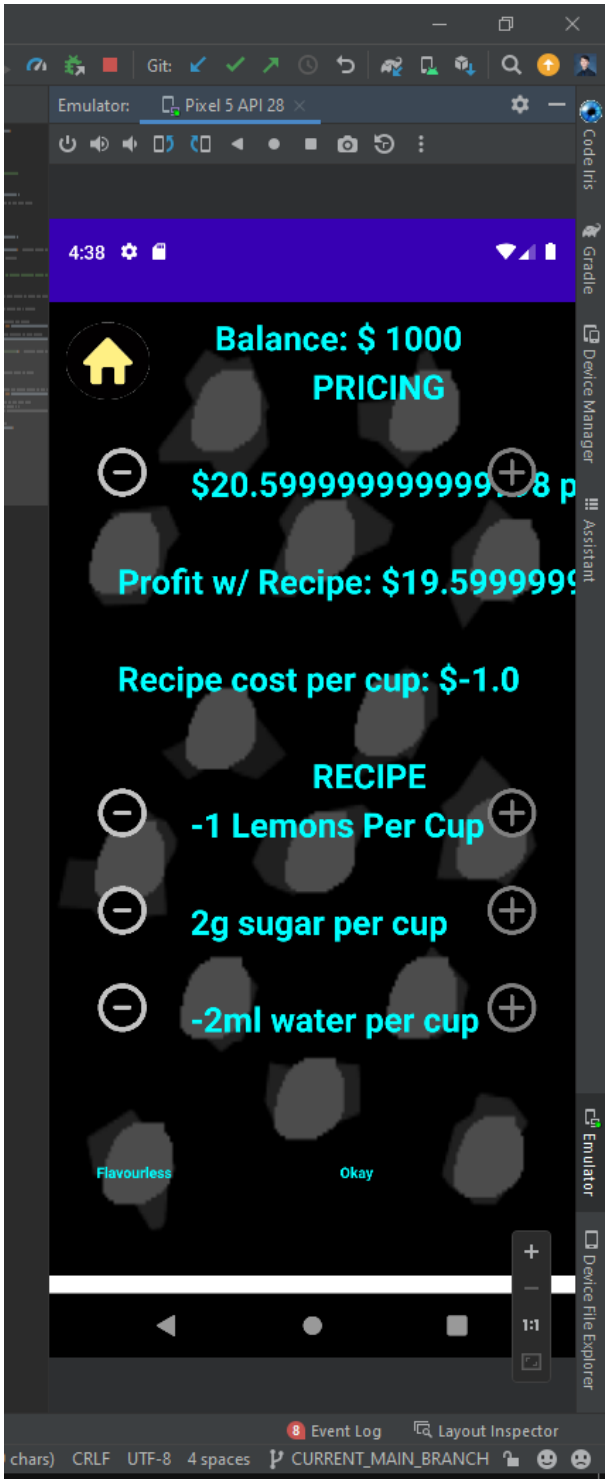


Figure 28 depicts a systematic ‘Touchscreen Input’ issue where not neglecting areas of space that feature commands to other parts of the program can be triggered from other parts of the program. This is only one example where the issue will go to a different menu because the current tool that is being displayed is simply drawn on top.

Figure 29: Long to Integer Conversion Drawn To View

Figure 29 describes when certain values that are long or doubles being drawn without conversion to integer problem. When discovering this issue, it was not really a problem. However, when understanding that the values being used here (and defined by the server) would be naturally irrational, it led to this task being postponed for other tasks to be completed.



3.5.3. Unit Testing

This section describes the inherent unit functionality that is performing the task, at least intends to. However, once networking was implemented into the client, the source of data was different and how it would pass new data was different and required an extra implementation to do it properly. Mostly, all the values that can be passed around and used in the client are effective in the task, however, do not have any limits or error catching for null cases. To further describe the issues, overflow errors or drawing to screen errors or values going below zero when it shouldn't, such as lemons or water stock or player balance.

Figure 30: Values 'Purchased'

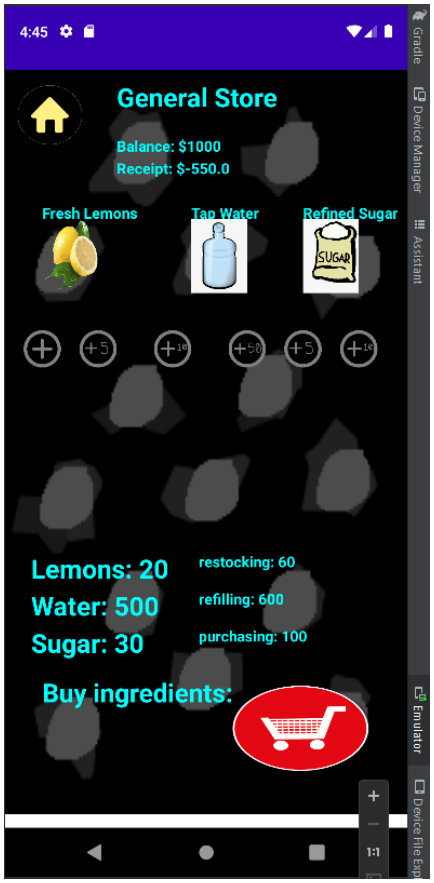
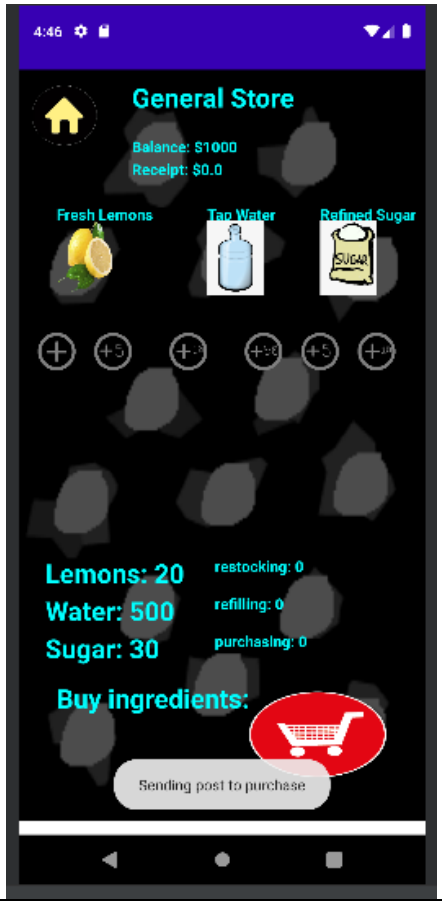


Figure 31: Values 'Sent' Without Posting



Figures 30 and 31 describes the missing functionality to 'POST' new data to the server to update the current 'Player' fields. This functionality is also missing in the core functionality of the 'Campaign game' states where it is essential to update new data.

3.5.4. Physical Demo Test

This section describes the physical devices demo testing which is an important aspect in testing to understand how the application performs in a real scenario implementation. The three physical devices used in the test were: Oppo F1f (OPPO, 2016) running ColorOS, Huawei DUB-LX2 (Consumer.huawei.com, 2019) running EMUI, and a Google Pixel 2 (Android, 2017). The best device to run the application flawlessly was the Pixel 2 and ran into less errors than when it was simulated. All devices were able to connect to the server. Excluding the Oppo, the devices are able to log into the different accounts and use the 'Admin' to test offline when the server was not available.

Figure 32: Three Physical Devices Running the Client with Connection

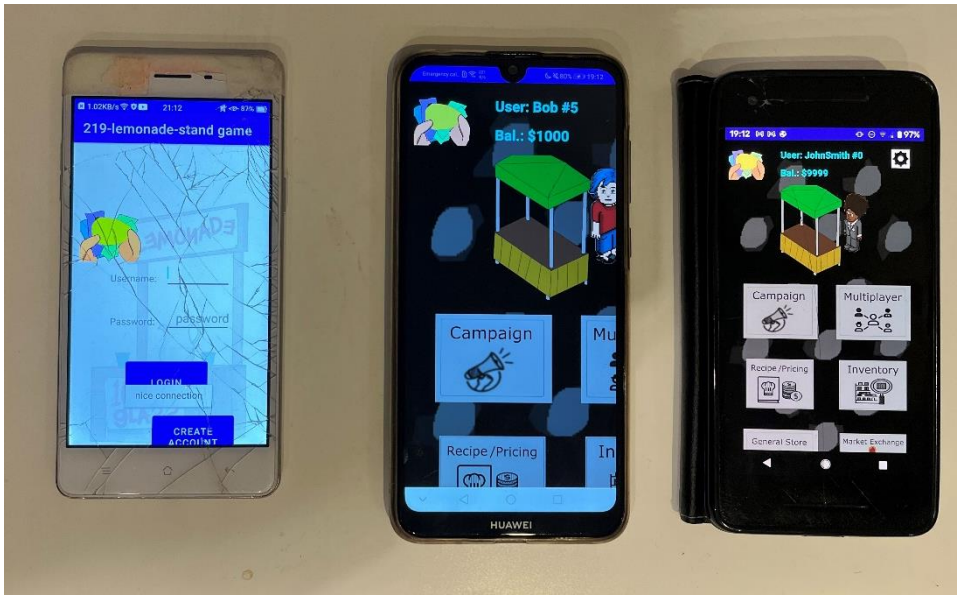


Figure 32 displays the three devices in action where the Huawei and Pixel are logged into the different accounts and the Oppo has at least established connection to the server.

Figure 33: Discovered RAM and Displaying Issues for the Oppo Phone

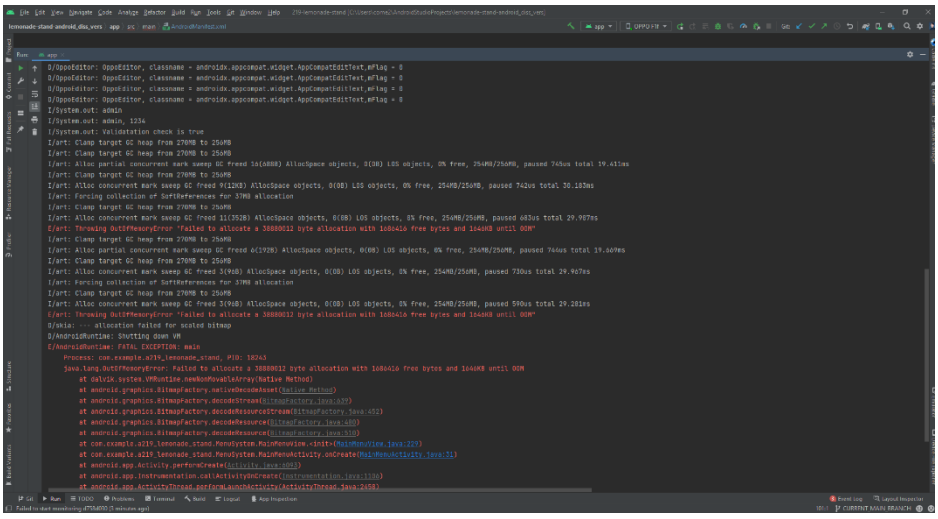


Figure 33 depicts the error occurred when running the client application beyond the login screen. Essentially, the client is requiring more RAM for displaying but where the resources would be sufficient for the phone, it is being held by the Operating System. However, this error describes issues for the client if it were to be implemented and reach a larger audience with lesser powerful devices. However, this would require rigorous changes to the project.

Figure 34: Fully Functional Game on the Pixel 2

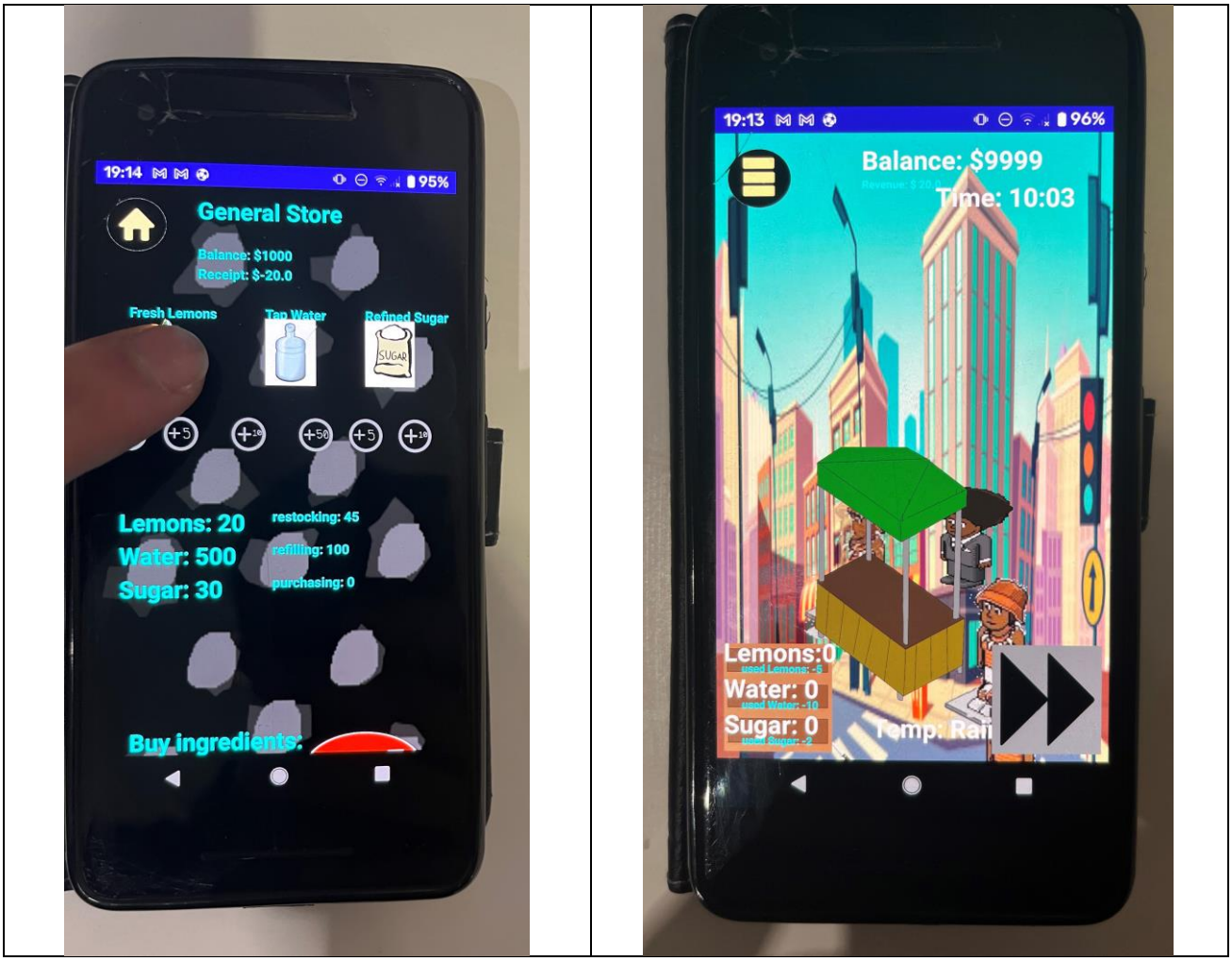


Figure 34 describes the Google Pixel 2 performing the features of the core game in action. The pixel was able to run the application and perform all the task on the simulated machine, potentially even faster when switching between activities. The Pixel 2 also ran into fewer crashes and was able to connect to the server and log into different accounts. It was very effective in offload testing and showing the performance of the project.

3.6. Implementation and Demo

The project implements an Android mobile game application. That will use a database that will hold the game server. The server will host the database and allow the user in the mobile application to connect to the server, create an account, sign in and use the game. The game will feature a take on the classic ‘Lemonade Stand’ game which will have multiplayer features to allow players to compete and gain wealth in this virtual environment.

Figure 35: Spring Boot Server Start and Security Configuration

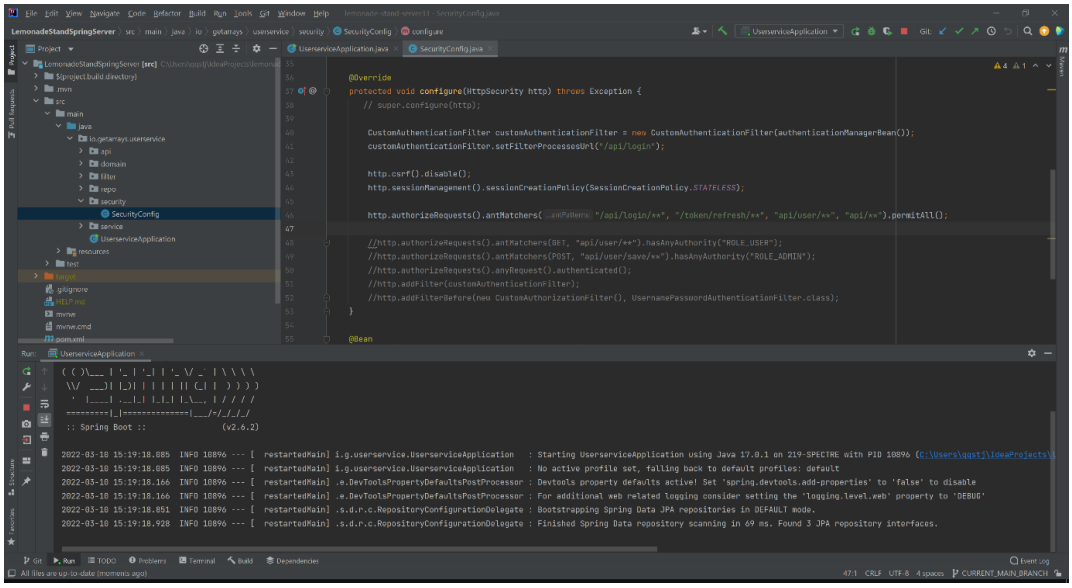


Figure 35 showcases the Spring Boot server at its initialisation phase and also showcases the current implemented security configuration which excludes the authorization and authentication filters.

Figure 36: Spring Boot Initialization

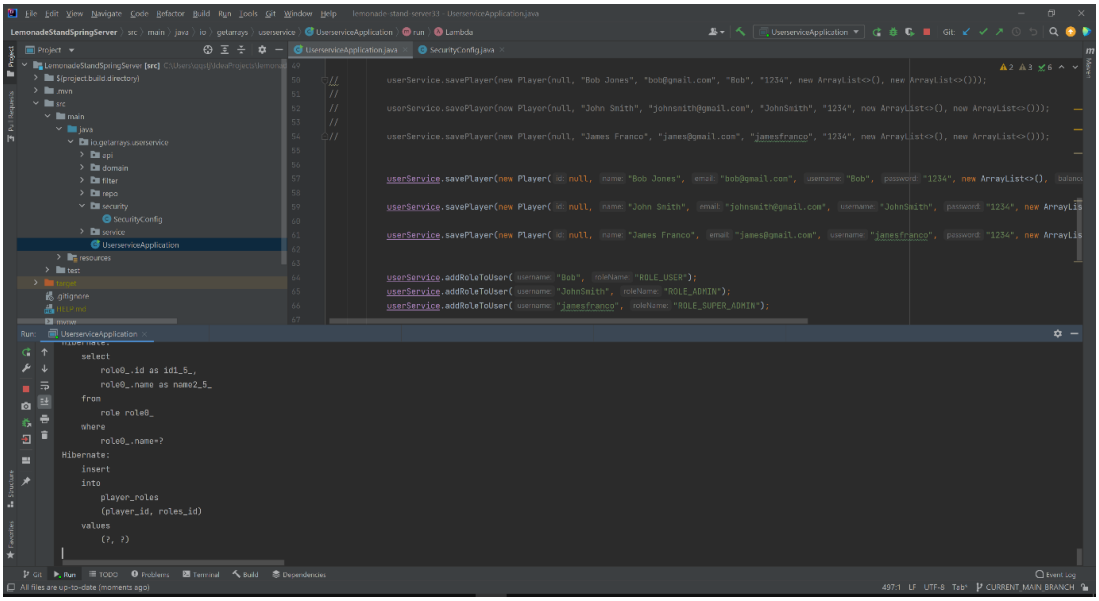


Figure 36 showcases the server when it starts providing the service and the predefined ‘Players’: ‘Bob’, ‘JohnSmith’, ‘jamesfranco’; defined with the parameters of the unique fields of the ‘Player’.

Figure 37: pgAdmin PostgreSQL Database Management System

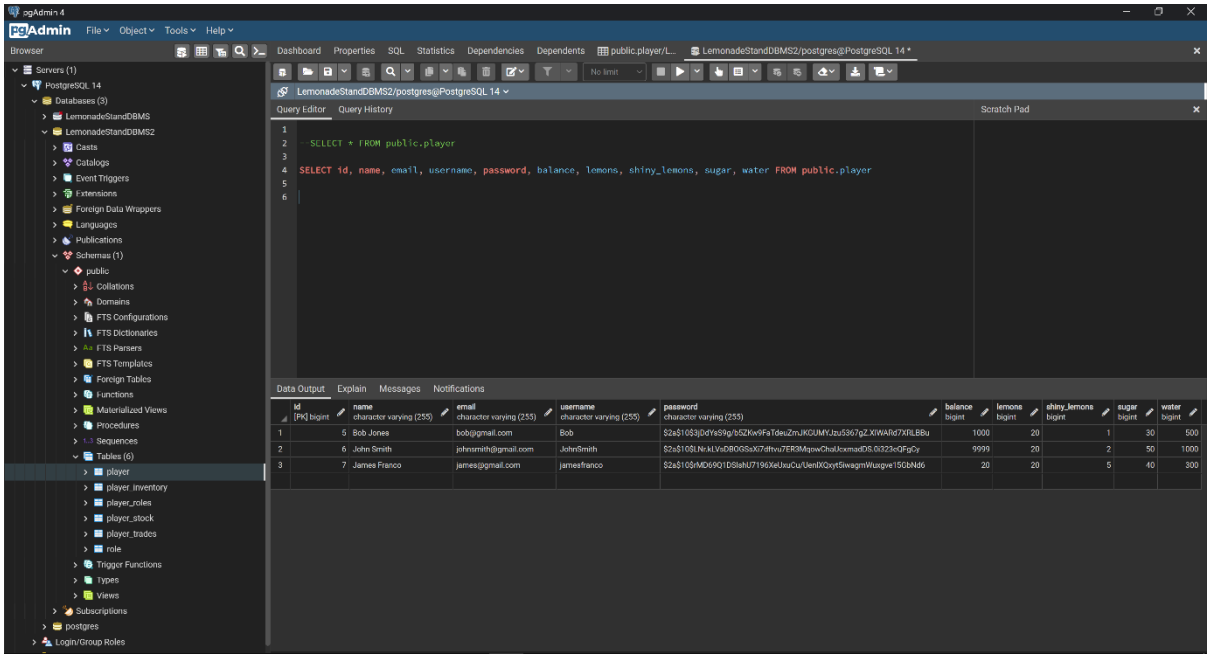


Figure 37 showcases pgAdmin 4, the database management system running concurrently with the server on the same computer. It displays a query of the ‘Player’ table and its corresponding columns.

Figure 38: Admin Testing Client Logging In

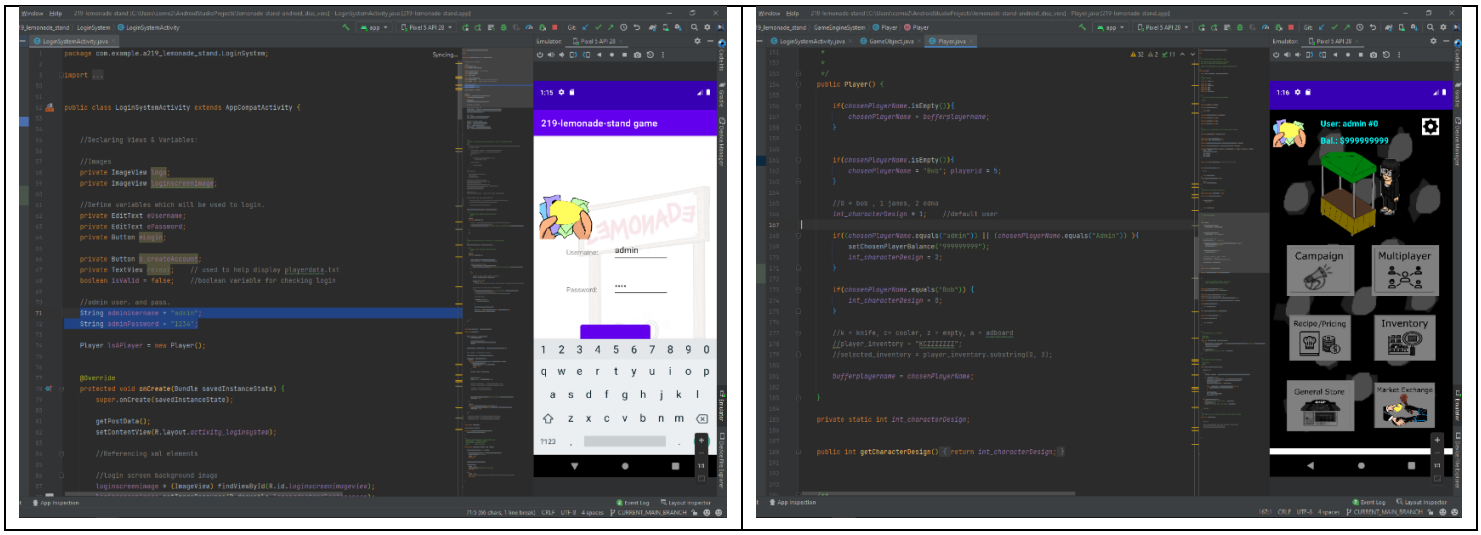


Figure 38 showcases when the ‘Admin’ player login. This is a backdoor account system that is used to test the client application without the connection of the server.

3.6.1. Core Demo Explanation

This section will describe what the project is attempting to achieve in the state. It will describe what function is does well, what needs improving and what features may be limited from developing further to limit time restraints.

Figure 39: Spring Boot Fetching Request For The Client

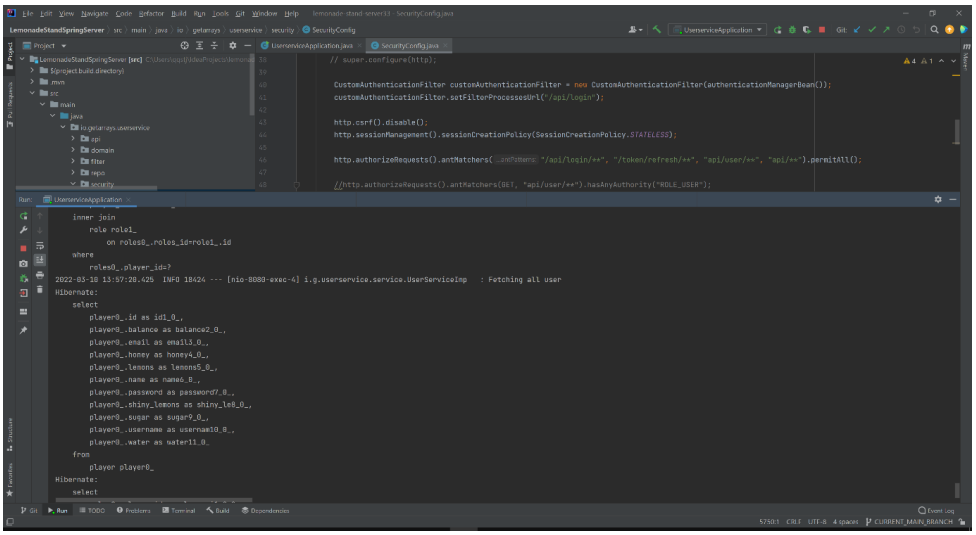
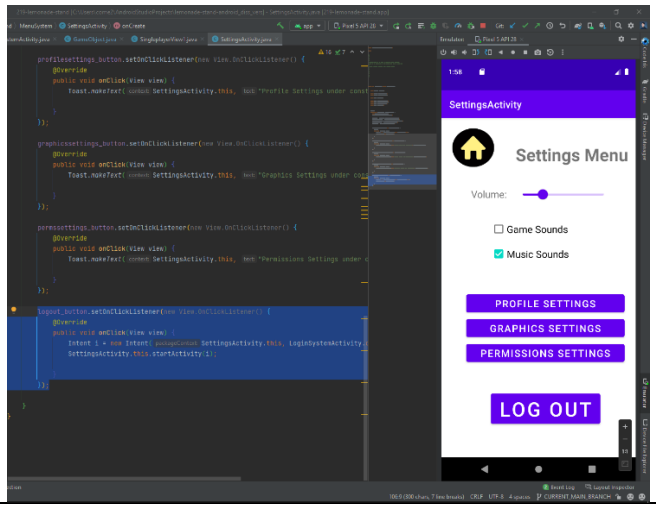


Figure 39 showcases the Spring Boot server application when the client application creates a fetch request to retrieve information from the server. The server provides the client with a 'json' data tuple which is then used to parse the data to the corresponding game 'Player' fields. The user will be logging into the 'Player' 'JohnSmith' but is also able to log into 'Admin' or any of the other predefined 'Players'

Figure 40: Logging In As 'JohnSmith'

Figure 41: Client Settings and Logging Out Feature



In general, the game and menu states working as intended and defined by UML diagrams. The 'Mainmenu' going into the corresponding menus and the menus exist.

Figure 40 showcases ‘JohnSmith’ user logging in and figure 41 showcases the user logging out of the account. Shown through the program are different logins, typically ‘Bob’ or ‘Admin’.

Figure 42: Campaign State 1 Features

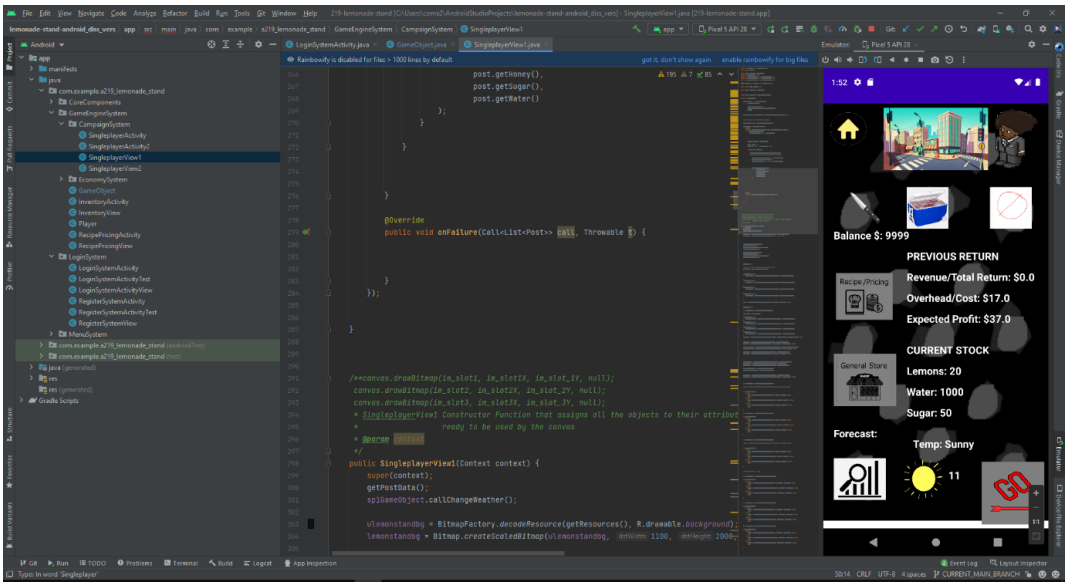


Figure 42 describes the features found in the first game state of the ‘Campaign state’ The use is able to see their inventory values and quick menus to core features that manage the business of the user. Moreover, it has an inventory organiser management system and a location selection system. The player can also see the weather before starting their ‘game day’.

Figure 43: Campaign State 2 - Bad Weather, Fewer Customers

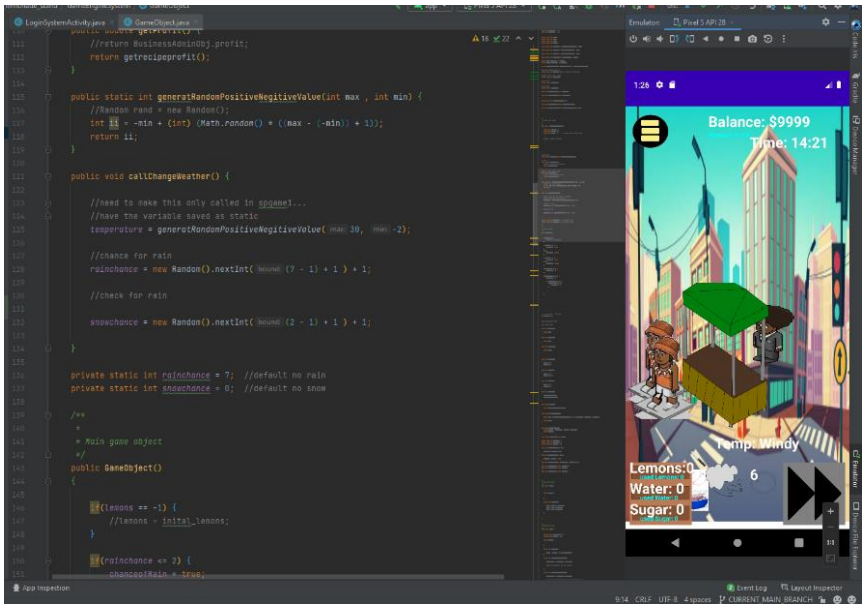


Figure 44: Campaign State 2: Good Weather, More Customers

The core game functionality that allows the 'Player' to sell their stock and gain revenue works and to enhance this. The variation on how much a 'Player' can profit from a 'day' change based on a working game 'temperature' mechanism that affecting the amount of customers present on the screen. Shown in figure 43, there is 'Bad weather' for this day and there are only 2 customers present in the day. In contrast, figure 44 shows a 'Good weather' which has 4 customers and up to a maximum 6 customers.

Figure 45: Recipe and Pricing State Functional Flavour and Values

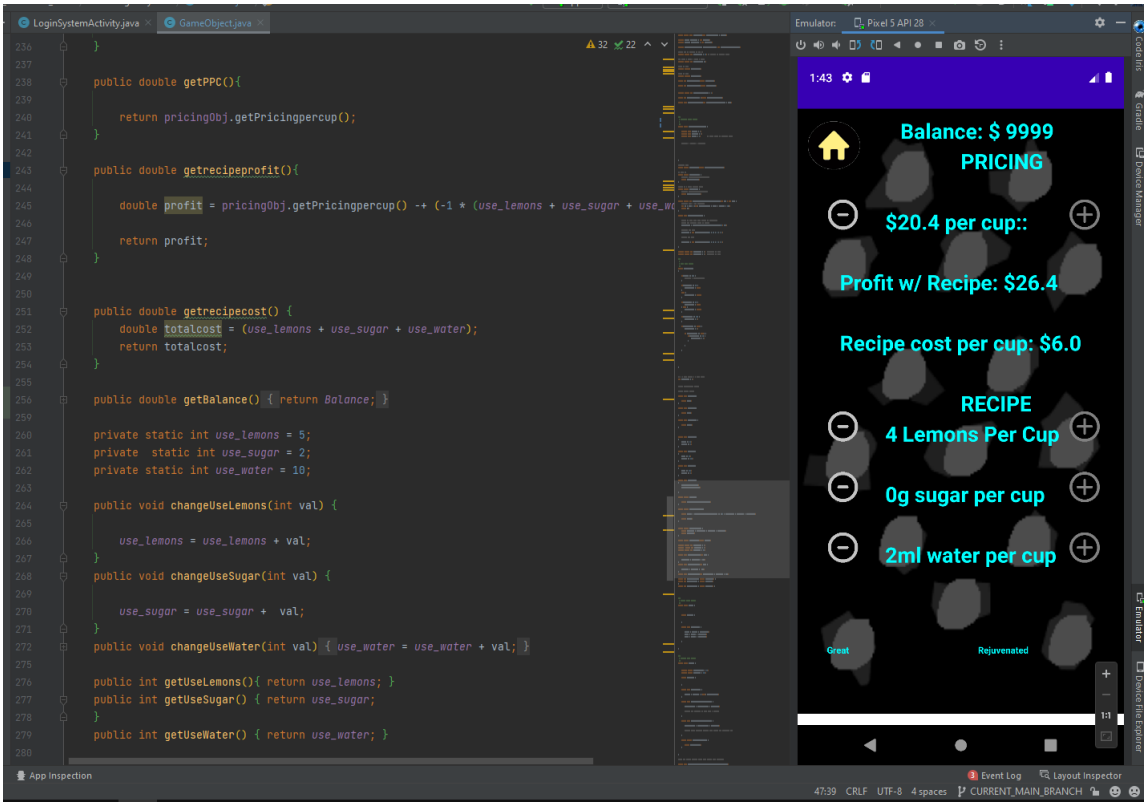


Figure 45 showcases the user changing values found in the ‘Recipe and Pricing state’. As shown, the flavour has improved to ‘great’ and ‘rejuvenated’ as less sugar is used in the recipe. As well as displaying the pricing of a cup at an increased $20.4 and shows expected returns.

Finally, all the functionality is shown to be functional on the physical device and multiple devices as well which proves that it is portable and has reach a state of being a minimum viable (passable) product.

Chapter 4: Professional Issues and Risks

This chapter aims to explore the professional issues and risks involved when developing the project. Understanding the overview of these issues ensure that a developer can understand the externalities and potential mishaps that may be caused through developing this project. '4.1 Professional Issues' will discuss and identify any relevant legal, social, ethical, and environmental issues as a by-product of this project. Several professional codes of conduct have been defined and pressed over the years such as the BCS, ACM and IEEE. Moreover, these professional issues are inherently a risk for the project. A risk is an uncertain event that may have a positive or, in general, negative impact on the project. '4.2 Risks' will undergo understanding the management of risk by identifying and mitigating these risks. It is important to carefully manage and predetermine the risks of the project as all aspects including, the schedule, scope, or budget, will be affected.

4.1. Professional Issues

The objective of identifying and discussing the professional issues is to lead with professional competence and integrity so that this project remains in the legal limits of the law and ethics to what society deems acceptable.

One of the major issues of this project is understanding copyright and its use. 'Lemonade Stand' may be an intellectual property and may have rights and owned by an entity. However, the main issue only is derived if the project is highly popular and gaining revenue in any way that the original owner has no part in. This paper, 'An Analysis of the Scope of Copyright Protection for Application Programs' (Menell, 2021) and discussions, its notoriously hard to create a legal offense on a game concept or idea as arguably there are many clones on the market not just of this game but game genres in general suffer from this.

Furthermore, this project will be represented as a research project and will not be within the scope of any copyright or legal issues. However, it is still informative to have presented any articles regarding the topics.

The discussion of the general computer ethics of acts. The Data Protection Act 2018 (GOV.UK, 2021) allows data subjects to have a greater control of their data and enforces data controllers to highly secure that data. The General Data Protection Regulation (GDPR, 2021) ensures that companies will follow the regulations of the policies as they would face severe punishment. The Institute of Electrical and Electronics Engineers (IEEE.org, 2021), has defined several ethical laws and procedures that companies must abide by. The BCS and IEEE has stated several computer laws and ethics that every developer must abide by. The relevant ones that are related to this topic are: data protection, security, and privacy. This is due to the players are

using data from the users and must be stored and processed securely and encrypted or this project could face problems if it were to be released and problems arise. This project will take the approach to solving this problem by inherently encrypting all private data files relevant to account information and securing properly. Furthermore, there are components or privacy features implemented so that it remains with in the laws, ethics and security standards are met.

As mentioned with virtual economies, there have been some legal disputes and laws that have been impose directly because of these games. As these game items may retain value and be taken advantage of, game companies have conceded from destructive legal disputes by stating that the game items are owned by the games company and not the user (tip.it, 2012). This means that items are not owned by the player and do not have any rights towards these virtual items. However, in other cases, game items are directly owned by the user (Partner.steamgames.com, 2022). This means that the developers have to ensure that these assets that users hold can be held reliably because some players that lose a lot of their assets may take legal action against the company. Moreover, 'On Regulation of Virtual Economies' discusses the impact of virtual economies and how government are requirement and understanding through reporting and revenues so that they can understand this virtual wealth and how it can impact the real-world economy. However, discussed in 'On Virtual Economies' (Castronova, 2014) and 'Development Informatics' (Heeks, 2008), restricting features from users playing the game may be a demerit implementation and negatively impacting; although it is also discussed that it is a strong feature.

This project will not need to describe what approach it will take as it will not be within the scope. However, to avoid issues, the items will not be owned by the player and hopefully real-world money will not be directly involved. Also, as this project is very early in its development, it will not define any restricting elements that may be unmoral or unethical as it attempts to achieve the solving. This is because the solutions to these problems can be solved later when it is within the scope.

4.2. Risks

The risk analysis will be informed by current state and progress of the project. This section regarding risks will hope to resolve risks and develop a successful mitigation strategy for any of the risks identified. Furthermore, it may define changes to project plan because of risks and any future risks that the project might lead into.

4.2.1. RISK: Risk Management

Risk management involves following several steps that will specify all the risk criteria and how to act on it. The following table identifies the 'Risk Management' steps:

Risk Step	Risk Description
1. Risk Identification	Defining all the relevant hazards and threats to the development or release of the project.
2. Risk Assessment	Identifying the project vulnerability by quantifying the risk, with a higher scale being a more damaging risk.
3. Risk Response	Aims to reduce or mitigate the risk factors by defining clear steps and actions for the risk.
4. Risk Control	Involves prioritising the risk mitigation strategies to keep focus on potential high attention risks.

4.2.2. Risk Breakdown Structure

Understanding how risks affect the project system, we will use several risk resources that helps identify what to do with risks or identify or define them. The Risk Breakdown structure goes as follows:

Technical	Requirements, Technology, Complexity, Interfacing, Performances, Reliability, Quality, Memory, Leaks, Dangling pointers, Deadlocks, Noise for data, etc.
External	Subcontractors' suppliers regulatory, Market, Customer, Weather, Pandemics, etc.
Organizational	Project dependencies, Resources, Funding, Prioritization, People-related issues, etc.
Project Management	Estimation, Planning, Controlling, Communication, etc.

A famous American Software Engineer discusses software project risk factors using the Boehm top ten software project risk factors (B. W. Boehm). These risks will be used to provide some potential ideas to risk that needs to be evaluated and mitigated.

Boehm's Top Ten Software Project Risk Factors

1. Personnel shortfalls:	Talent, job matching, personnel agreements.
2. Unrealistic schedules and budgets:	Problems in cost and scheduling, design to cost.
3. Developing the wrong functions and properties:	Problems in organisational analysis, operation-cost formulation, performance failure, quality failures.
4. Developing the wrong user interface:	Problems when not prototyping, no external participation for feedback.
5. Gold-plating:	Creating or allocating too much time on unnecessary features but makes it seem better.

6. Continuing stream of requirements:	Major changes to the project, information hiding, stall in incremental development.
7. Shortfalls in externally furnished components:	Benchmarking, inspections, compatibility analysis.
8. Shortfalls in externally performed tasks:	Reference checking, audits, competitive design.
9. Real-time performance shortfalls:	Simulation, modelling, benchmarking, tuning.
10. Straining computer-science capabilities:	Cost-benefit analysis, prototyping problems, reference checking.

4.2.3. Quantifying Risks

Generally, a risk has a likelihood or a probability of it occurring and the consequence or the impact of the failure or risk. A formula to describe the Risk expose can be defined using these as variables and calculated to product a value that ranks on a numeric scale to determine its severity. The severity of the risk at hand can determine how the risk can be prioritized. The formula for determining a value for Risk Exposure is:

$$Risk\ Exposure = Likelihood * Impact$$

Where ‘Likelihood’ or ‘Impact’ may be ranked on a scale of one to five and multiplied and referred to the Risk Severity Matrix. The value and colour will correspond to how tolerant a developer may be towards the risk or how the project’s tolerance towards the risk.

Risk Severity Matrix:

	Likelihood					
Impact		1	2	3	4	5
	1	1	2	3	4	5
	2	2	4	6	8	10
	3	3	6	9	12	15
	4	4	8	12	16	20
	5	5	10	15	20	25

Legend:	
	acceptable
	Tolerable
	High Risk

Figure 10: Risk Management Table

Potential Risk	Potential Causes	Severity	Likelihood	Risk	Mitigation
Missed deadline	Illness	1	3	3	Apply for exceptional circumstances
	Poor Time Management	4	2	8	Apply schedule and time management skills and work ahead.
	Other Obligations	3	2	6	Keep ahead of other obligations.
Feature Creep	Over-ambitious Project Feature or Specifications	2	3	6	Keep scope at the current state and manage time efficiently to develop core/high priority tasks.
Poor Artefacts	Poor motivations, unnecessary by perspective.	1	3	3	Constant design and development of the artefacts.
Missing Artefacts	Neglected.	2	1	2	Create a basic artefact to be able to implement it later, perhaps.
Software problems	Bad code; Poor implementation of component.	4	4	16	Consistent development and learning. Prevention by solving the problem before it becomes a risk.
Software minor bugs	No software unit checks; Poor implementation.	3	5	15	Not entirely terrible as the variables seems to be implemented and working but if it doesn't use or show the right data it must undergo tests so that the component is considered functional.
Software major bugs	Failure of implementing feature.	5	2	10	Ensure that the minimum viable product is achieved at least so that this risk can never be reached.
Stalls in development	Other errands, procrastination, stress.	2	3	6	Ensure there is some development or planning for some development such that this doesn't occur.
Computer Ethics/Professional Issues	Failure to meet the standards.	4	1	4	Checking and meeting the requirements based on research and findings.
Virtual economy regulations	Failure to implement standards for these games.	4	1	4	Ensuring that the regulations are discussed and sought to when relevant.

Copyright laws	Failure to meet the standards.	3	1	3	Ensuring that the issue are discussed and sought to when relevant.
Product inefficiency/performance issue	Unable to write efficient code that is reliable and fast for the scope.	4	3	12	Making sure that the issue at hand is dealt with as soon as possible. Proper use of multi-threading as it is a complex issue to solve later.
Compatibility issues.	Failure for the application to reach the objective of being compatible	4	4	16	Testing and prototyping the product on the objective device early.

Chapter 5: Final Extrapolations and Conclusion

5.1. Review and Retrospective

The project featured all the core components of the proposed game, selling lemonade and managing a business. Furthermore, it was able to integrate the server and database components to allow multiplayer and account-based systems, although not properly using security standard techniques and properly being able to parse data to and from the server properly. The downfalls that led to the limitations of the project is due to the features required and the technique and methodology used for the project.

The client features were implemented quickly and without the approach with advanced object-oriented programming which in effect caused the server to be difficult to define proper integration and had to be reflected off what was initial propose in the prototype. Therefore, any feature that was required from the server because of the client would require, would lead to changes in the server code and then the client complying with major reworks to the networking code implementation.

5.2. Future Implementation

Although the project proved to be inefficient in the effective code and how the client communicated with the server, it is important to note that there were major components that were developed and need to be encapsulated or further improved.

With that being said, the project should have the server and its requirements defined and perfected before the client was developed. Ensuring that the server functionalities were achieved as it would not cause any lag or stalls in development for the server to provide the service that the client needs. Therefore, greater planning for the server and the composing systems would be a better approach.

Furthermore, there are a lot of issues that the code has and the flaws that lie under these issues. Therefore, any further development or plans for release would require sufficient means of ensuring quality of service improvements such that these issues do not exist. However, these issues are abundant and exists in several components of the system. This means

it would probably require a team or roles dedicated for the components so that it is maintained properly and providing a good quality service or code technique.

5.3. Conclusion and Reflection

This project is an effective method for an aspiring software engineering graduate or person to show what different components of a computer science degree is able to create as a product. From including the understanding of creating a high-quality designed game application system for a smart mobile operating system, with complex object-oriented techniques and algorithms to networking and security and blockchain and economics theories. All undergoing through an individually planned and formalised process and documenting and practicing the standard methodologies of Agile and UML and expectations of the industry. Also, insight and discussion into the social, legal aspects for this given game software system.

Chapter 6: Bibliography and References

This section will include all the necessary and required references for this project. It will also include any links towards the artefact and the project.

6.1. Bibliography

[1] Wikipedia. (2021). *Lemonade Stand*. [online] Available at: https://en.wikipedia.org/wiki/Lemonade_Stand [Accessed 15 Oct. 2021].

[2] Nesta. (2021). *The Innovation Foundation*. [online] Available at: <https://www.nesta.org.uk/> [Accessed 15 October 2021].

[3] Coolmathgames.com. (2021). *Lemonade Stand - Play it now at CoolmathGames.com. [online]* Available at: <https://www.coolmathgames.com/0-lemonade-stand> [Accessed 15 October 2021].

[4] Friedman, Nat. (2008). *GitHub*. Web Browser. Microsoft.

[5] Page, Larry. *Google*. (1998). Everything. Google.

[6] Google. (2012). *Google Drive*. Web Browser. Google.

[7] Google. (2008). *Android*, Windows. Google.

[8] Google. (2021). *Android Studio*, Windows. Google, JetBrains.

[9] Microsoft. (2015). *Windows 10*. Windows. Microsoft.

[10] Nvidia. (1993). *Nvidia Graphics cards*. Nvidia.

[11] Intel. (1968). *Intel Processors*. Intel.

[12] EA Mobile. (2002). *Lemonade Tycoon*. PC, Mac and iOS. Hexacto, Airborne, EA Mobile.

[13] Chris Sawyer Productions. (1999). *RollerCoaster Tycoon*. Microsoft Windows and Xbox. Chris Sawyer Productions.

[14] Armor Games. (2007). *Coffee Shop*. Available at: <https://www.coolmathgames.com/0-coffee-shop> [Accessed 15 October 2021]. Armor Games.

[15] Jagex. (2001). *RuneScape*. PC, Web Browser, Mobile. Jagex, Jagex.

[16] Valve. (2003). *Steam*. PC, Web Browser, Mobile. Valve.

[17] Dapper Labs. (2017). *CryptoKitties*. Dapper Labs.

[18] IEEE.org. (2021). *Institute of Electrical and Electronics Engineers*. [online] Available at:

<https://www.ieee.org/> [Accessed 15 October 2021].

[19] ACM.org. (2021). *Association for Computing Machinery.* [online] Available at: <https://www.acm.org/> [Accessed 15 October 2021].

[20] SSRN.com. (2021). *Social Science Research Network*. [online] Available at: <https://www.ssrn.com/index.cfm/en/> [Accessed 15 October 2021].

[21] Castronova, Edward. (2002). *On Virtual Economies.* Social Science Research Network.

[22] Heeks, Richard. (2008). *Development Informatics: Current Analysis and Future Research Agenda on "Gold Farming"*. Development Informatics Group.

[23] Castronova, Edward. Knowles, Issac. Ross L. Travis. (2014). *Policy Questions Raised by Virtual Economies*. Elsevier Ltd.

[24] Shiratuddin, Norshuhada. Zaibon, Bahrin. (2011). *Designing User Experience for Mobile Game-Based Learning*. IEEE.org.

[26] Balakrishnan, Janarthanan. Griffiths D. Mark. (2019). *Computer in Human Behavior*. Elsevier.

[27] Menell, S. Peter. (2021). An Analysis of the Scope of Copyright Protection for Application Programs. HeinOnline.

[28] GOV.UK. (2021). *Data protection*. [online] Available at: <https://www.gov.uk/data-protection> [Accessed 15 October 2021].

[29] GDPR. 2021. *General Data Protection Regulation (GDPR)*. [online] Available at: <https://gdpr-info.eu/> [Accessed 15 October 2021].

[30] B. W. Boehm, "Software risk management: principles and practices," in *IEEE Software*, vol. 8, no. 1, pp. 32-41, Jan. 1991, doi: 10.1109/52.62930.

[31] Wall Street Journal, 2021. NFTs Are Fueling a Boom in Digital Art. Here's How They Work | WSJ. [video] Available at: <https://www.youtube.com/watch?v=zpROwouRo_M&ab_channel=WallStreetJournal> [Accessed 10 December 2021].

[32] Beattie, A., 2021. Four Economic Concepts Consumers Need to Know. [online] Investopedia. Available at: <https://www.investopedia.com/articles/economics/11/five-economic-concepts-need-to-know.asp#:~:text=Four%20key%20economic%20concepts%E2%80%94scarcity,many%20decisions%20that%20humans%20make.> [Accessed 10 December 2021].

[33] Oxfordlearnersdictionaries.com. 2021. gold-farming noun - Definition, pictures, pronunciation and usage notes | Oxford Advanced Learner's Dictionary at OxfordLearnersDictionaries.com. [online] Available at:

<https://www.oxfordlearnersdictionaries.com/definition/english/gold-farming> [Accessed 10 December 2021].

[34] Ombler, M., 2020. How RuneScape is helping Venezuelans survive. [online] Polygon. Available at: <https://www.polygon.com/features/2020/5/27/21265613/runescape-is-helping-venezuelans-survive> [Accessed 10 December 2021].

[35] Klewitz, M., 2021. Code Iris. www.codeiris.com: JetBrains.

[36] OPPO. 2016. OPPO Global OPPO F1 Selfie Expert | OPPO Global. [online] Available at: <https://www.oppo.com/en/smartphones/series-f/f1/> [Accessed 11 March 2022].

[37] Consumer.huawei.com. 2019. HUAWEI Y7 Pro 2019. [online] Available at: <https://consumer.huawei.com/kh/support/phones/y7-pro-2019/> [Accessed 11 March 2022].

[38] Android. 2017. Google Pixel 2. [online] Available at: <https://www.android.com/intl/en_uk/phones/google-pixel-2/> [Accessed 11 March 2022].

[39] Spring.io. 2002. *Spring Boot*. [online] Available at: <https://spring.io/projects/spring-boot> [Accessed 11 March 2022].

[40] JetBrains. 2001. IntelliJ IDEA: The Capable & Ergonomic Java IDE by JetBrains. [online] Available at: <https://www.jetbrains.com/idea/> [Accessed 11 March 2022].

[41] Page, D., 1996. pgAdmin - PostgreSQL Tools. [online] Pgadmin.org. Available at: <https://www.pgadmin.org/> [Accessed 11 March 2022].

[42] Postman API Platform. 2022. Postman Application. [online] Available at: <https://www.postman.com/> [Accessed 11 March 2022].

[43] tip.it. 2012. Does Jagex Own In Game Items?. [online] Available at: <https://www.tip.it/runescape/times/view/795-does-jagex-own-in-game-items> [Accessed 11 March 2022].

[44] Partner.steamgames.com. 2022. User Authentication and Ownership (Steamworks Documentation). [online] Available at: <https://partner.steamgames.com/doc/features/auth> [Accessed 11 March 2022].

6.2. Artefacts and Source Code Links

I. Source code GitHub Links:

- GitHub Organisation: https://github.com/lemon-stand
- Android client: https://github.com/lemon-stand/lemonade-stand-android
- Spring server: https://github.com/lemon-stand/lemonade-stand-server

II. UML Diagrams Link: https://github.com/lemon-stand/docs/tree/main/UML%20_Diagrams

III. Artefacts Links: https://github.com/lemon-stand/docs

IV. Google Drive Folder Link: https://drive.google.com/drive/folders/17uE8x1u6mzwm8xwVpWKuQQ5mHhpFRQE2?usp=sharing

V. GitHub Storyboard Links:

- Initial Storyboard: https://github.com/lemon-stand/lemonade-stand-android/projects/1
- Dissertation Storyboard: https://github.com/orgs/lemon-stand/projects/2
- Future Storyboard: https://github.com/orgs/lemon-stand/projects/1/views/1?layout=board