Lemonade Stand | Multiplayer, Economy-based Android Game Concept

'coolmathgames.com Lemonade Stand'

'EA Games Lemonade Tycoon'

Original Lemonade Stand Game on Apple II

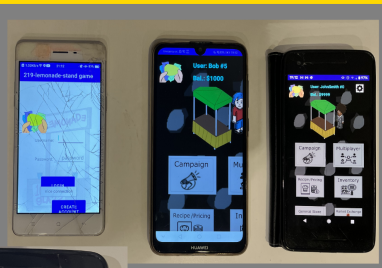
Introduction:

This project aims to provide a web application service represented as a multiplayer, Android game. It attempts this by defining a problem and choosing a topic to base the problem against. The topic at hand is: 'Money'. To broaden this issue, money was decontructed into understanding and teaching the systems behind economic and business theory. A popular game genre called 'Business Simulations' are a popular technique that encapsulates the solution to this problem. A classic business simulation game is known as 'Lemonade Stand' which is a popular game that was intergrated into many people's lives if they explored a bit of the web. Many interpretations of 'Lemonade Stand' have been developed over decades beginning with the Apple II. Another take on 'Lemonade Stand' has been created by popular education, Mathmatics, website 'coolmathgames.com' and 'Electronic Arts'.

The task defined was to develop 'Lemonade Stand' but to add a multiplayer component to the core game application to feature a concept known as a 'Virtual Economy'. To extrapolate this, a 'Player' may trade their 'items' for a virtual currency. However, this definition alone does not create an economy; As discussed in the dissertation, several methods are defined to establish this. In summary, a 'Player' may obtain a unique or special 'lemon' or 'item'. Utilising the 'unique lemon' or 'item' in their recipe may increase profits and therefore retain a higher demand value. Furthermore, these 'items' can be intergrated on a blockchain system to create a new mechanism of value and economy outside the application. Moreover, 'items' are scarce so that the items have a supply chain rather being abundant.

This dissertation also discussed the security standards required behind an account-based logging in system. This feature that was attempted to be implemented in the system, that led to the delays to other features or causing tasks to be pushed back.

Fully functional game working on a Pixel 2



Three different devices connected and running the game

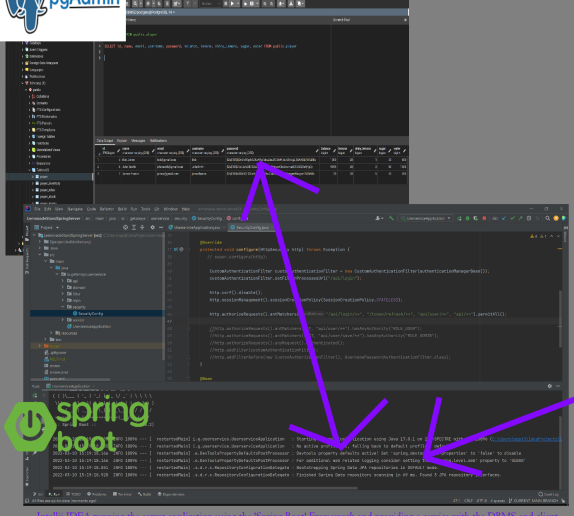
User buying stock in the game on a Pixel 2

Implementation & Demo:

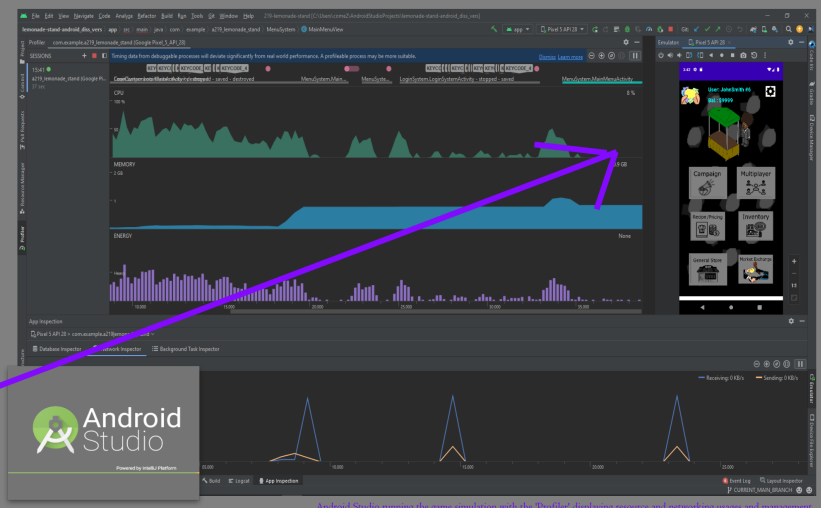
This section showcases the game application running in action and real-time. It features an image where three devices are connected and playing the game on different accounts. It shows that the application is functional and performs the task. Furthermore, it was essential in understanding the testing and demonstrating the core elements of the game.

Physical devices used:
Googel Pixel 2 (2017), Android 11
Huawei DUB-LX2 (2019), Android 8.1.0, EMUI
Oppo F1f (2016), 5.1.1, ColorOS

pgAdmin 4, Database Management System providing the game service for the server and displaying a query



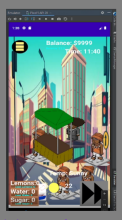
IntelliJ IDEA running the server application using the 'Spring Boot' Framework and providing a service with the DBMS and client

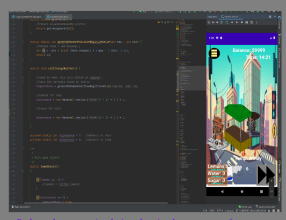


Android Studio running the game simulation with the 'Profiler' displaying resource and networking usages and management

Technical Methodology & Approach:

The comprising project was mainly developed in the Android Studio IDE using Java creating the client system application. The server system was developed using IntelliJ IDEA, Java, using the Spring Boot framework to provide the intergration and functionality for the database and web communication service. The Database Management System was provided by pgAdmin 4, PostgreSQL. As shown in the relationship diagram above, the client is logged into the 'Player' 'JohnSmith', loaded with all the corresponding fields provided by the server. Moreover, the Android Studio diagram features the 'Android Profiler' which showcases the system resource usages and the networking with a graphic describing the communication when the server requests it. The pgAdmin diagram features a query describing the 'Player' table and the corresponding fields. Finally, the Spring Boot diagram showcases the initialisation of the server with the corresponding web security configuration.





Great weather game simulation showing higher count of customers VS Bad weather game simulation showing lower count of customers

References, Repositories & Links:

- GitHub Organisation: < github.com/lemon-stand > <.../lemonade-stand-server> <.../lemonade-stand-android> <.../docs>
- Google Drive: < https://drive.google.com/drive/folders/17uE8x1u6mzwm8xwVpWKuQQ5mHhpFRQE2?usp=sharing >
- UML Diagrams: < https://github.com/lemon-stand/docs/tree/main/UML%20_Diagrams >
- Storyboards: < https://github.com/orgs/lemon-stand/projects?type=beta >
- Video Demo: YouTube: < https://www.youtube.com/watch?v=JI1anwq5n3c&feature=youtu.be >

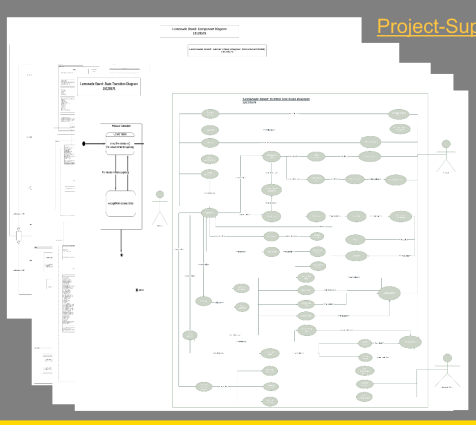
Panopto:

< https://brookes.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=9a4f17f3-a205-47e5-a8b7-ae57015c8da4&start=0 >

Game Weather Feature Demo Description:

The game features a game weather mechanism within the system that fluctuates the profit of a 'Player' on any given day. This game weather is randomized and when the weather is poor, the population of the customers is reduced. This means that the user may reevaluate if they want to sell for that day or how they should charge their customers. If the weather is good, above (15 degrees and not raining) there will be a chance to have a larger population in that day. This allows up to 8 customers to buy lemonade compared to 2-4 if the weather is windy or below 14 degrees.

To further describe what features are in the game, a good lemonade recipe will allow the user to recieve more profit for their lemonade. However, As mentioned and described in the dissertation, several components that would normally be checked and cleared from the system and application are missing from the program. Some values may go below zero or null. Some values that are affected by a 'long' or 'double' data type are drawn to a screen when it should be converted properly into an integer representation. Finally, systematic issues may cause the system to crash unexpectedly.



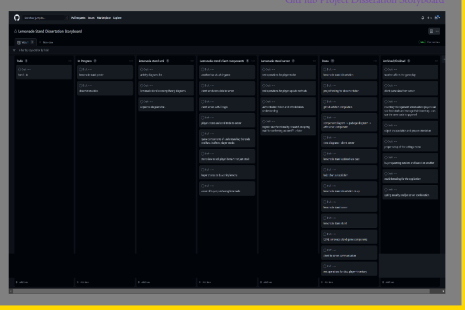
Project-Supporting Artefacts:

This project underwent through rigious planning, documentation and using standard methodologies to approach the task. The Unified Modelling Language (UML), created requirements for the system and was essential in describing and defining the core functionalities of each component and how they interact with each other. Furthermore, several artefact documents were concieved as a result of UML to create an understanding of the tasks as 'Sprints' and 'Priority-driven' tasks. Other essential documents include: Requirements Modelling, which helped defined what was required; Log book, establishing a timeline and documenting what occured and how it was achieved.

The UML Diagrams include a Use-case, State Transition, a Component model (comprising of the entire system), and two Class diagrams for client and server system. The Use-case diagram helps model the features a user, described as a 'Player', may be able to do in the system and helps define what components needed to be in the present. The State Transition diagram conveys the application states and the processes that the system go through. It is effective in providing navigation and what each state is capable of. The Component model is essential in describing how the data is communicated in the intergrated system, through describing the components relationships and connections of the client to the server and to the database. The 'Client' Class diagram describes all the features and functionalities behind the game system application; it lays out the classes in their corresponding packages and how they use other classes. The 'Server' Class diagram describes the functionality behind the server and how it intergrates the Spring Boot service to provide the 'Player' repositories.

The UML and agile methodology requires standard planning techniques. Using GitHub's provided feature, several storyboards were designed to accomodate the 'Priority-driven' tasks and any issue on the back log. The storyboard presented showcases the storyboard when the project was reaching the completion stage. As shown, several tasks have been moved towards the 'completion' column. Yet there are several tasks that ended up being pushed back as discovered and discussed in the dissertation.

Several UML Diagrams designed for planning and defining tasks and systems



GitHub Project Dissertation Storyboard