ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

Summer Practice Report

CENG400

STUDENT NAME: Berat Furkan Koçak

STUDENT NUMBER: 2380665

FACULTY – DEPARTMENT: Faculty of Engineering – Computer Engineering

SUPERVISOR NAME: Prof. Dr. Pınar Karagöz

REPORT TITLE: Optimized Game Development with Clean and Structured Code

ORGANIZATION NAME: Simsoft Bilgisayar Teknolojileri Ltd. Şti.

ADRESS: Üniversiteler Mah. 1596. Cadde 6.AR-GE E Blok 2.Kat Ofis No: 12 Beytepe - Çankaya Ankara

START DATE: 18.07.2022

END DATE: 26.08.2022

TOTAL WORKING DAYS: 30

STUDENT'S SIGNATURE **ORGANIZATION APPROVAL**

TABLE OF CONTENTS

Contents

1 INTRODUCTION

A game is a specialized software, using user interactions in order to create enthusiasm and fun. In that sense, a developer should be aware of his responsibility to follow the best practices as in any other software development. The most important of these responsibilities is to provide reliable, secure, safe and scalable code.

Creating a mobile game as a member of a developer team in a game development company requires a well-established communication between all units, clear goals and profession. In this internship, my goal was to develop my existing engineering skills and learn new concepts that would make me a reliable and efficient developer in the industry. In that perspective, I decided to choose a well-known, successful and interdisciplinary company, Simsoft Computer Technologies. The company is focusing on doing research and development on simulation systems, military software and serious game applications.

My internship was conducted face-to-face, giving me the opportunity to experience an agile working software company's daily routines. I have learnt several beneficial concepts from my colleagues and my supervisor during my 30 work day internship period.

From the beginning of my internship, upon my request, I am treated like a junior developer instead of an intern. This specific request was to make the best out of my internship period, resulting in gaining the knowledge and skills that a junior developer must have.

At the beginning of my internship I am asked to share a detailed review on companies previous made most successful game projects considering different aspects of a quality software like reliability and scalability. After the evaluation of my review, I got the chance to choose which project I want to work on. My decision was to work on the biggest game of the company, Mega Capital. This game is a product of at least 3 years of development and contains hundreds of classes with thousands of well written code.

My internship tasks can be partitioned into these tasks: Inspecting and understanding all scripts in the project, Fixing the not working legacy database connection and fixing the in app ad scripts using a new ad provider's SDK. Apart from this project, I also implemented a complete hyper-casual mobile game from scratch working with design and art teams.

The company had specific standards in how the codes should be written and game files and namespaces should be structured. I adapted myself to these standards and delivered clean and structured code with following several design patterns.

In this report, I will explain the tasks that I performed and associated analysis, design, implementation and testing phases. They will be organized in weekly manner. While explaining the tasks, I will also discuss the problems I faced and how I managed to solve them. Afterwards, I will introduce the organization, its structure and the methodologies used by the organization. In the summary part, I will provide the conclusion and a discussion of my internship period.

2 REPORT

2.1 INTRODUCTION AND MOTIVATION

Being involved in development of a complex software requires certain understanding and skills in order to achieve a standard quality. As an intern who willingly participated in the biggest project of the company, I aimed to qualify myself to that certain point where my work would fit and scale the effort that previously put on the project. Taking this into consideration, I decided to read most of the code previously written, understand and learn the structures and patterns used in the project and standards of the company. As a result, my focus was providing a better and more optimized software (a game) than I found.

2.2 FIRST WEEK

2.2.1 ANALYSIS PHASE

2.2.1.1 LEARNING GAME ANALYSIS

In order to create a good and reliable game, I needed and encouraged to learn how to analyze successful games considering the most important concepts and metrics. For that purpose, my first internship task was to analyze the top three games created by the company. The purpose of this was to understand the essentials things that makes a game “good” for the user and apply those to my work throughout my internship period.

As a student ambassador of Voodoo, a known Paris based game company, I had access to the essential hyper-casual game development courses they provide. After carefully taking those courses, I learnt that these factors have critical importance in the success of a game:

Depth: This is a measure of whether the player has different experience with the same content each time he plays. This concept boosts the overall retention and playtime.

Unlimited Upskill Potential: This is a concept whether a player can get better at the game and challenge his skills as he keeps playing. This concept increases long term retention.

Clear Choices: This is a concept whether difficulty in the game is an option. It is important that user shouldn’t feel being forced, instead he should be provided with different options.

Reward and punishment: Punishment in a game is usually frustrating and creates a bad user experience. Rewards must be a direct consequence of high risk. In that sense, user is given the chance to gain more with taking higher risk.

Instinctive and Fast: Mobile games that have a fast pace, short game sessions and contain instinctive moves are likely to be played more. These kind of games are named "System 1" in the hyper-casual game industry.

Taking these parameters into consideration, I was ready to analyze the games by playing and reading their GDDs (Game Design Document). I spent first three days of my internship on these topics.

2.2.1.2 LEARNING NECESSARY DESIGN PATTERNS

When I started my internship, first and most important advice I received from my supervisors was to write clean and optimized code. This has several benefits like making the project scalable and reliable.

I was lucky that at the beginning of my internship, my supervisors in the company had just initiated a program where they gave lectures on the important principles and patterns. It was an everyday and two hours of lecture program which I attended throughout my internship. I have learnt several design patterns that I implemented later in the company's projects such as message bus, object pooling, singletons, etc.

2.2.1.3 RESEARCH ON HOW TO DIVE INTO A LEGACY CODEBASE

Milecia McGregor from FreeCodeCamp states that "Working as a developer means you need to know how to dive into existing code bases"[1]. In this perspective, I performed a research on how to dive into a codebase containing hundreds and thousands of line of code and several scripts with large amount of classes, structures and different anming standards for the company.

As a result of my research I gathered many useful tips that I used further in my design process for a clearer understanding on how to contribute to an existing project.

[1] McGregor, M. (2020, May 17). *Legacy Code Tips – How to Take Over an Existing Project and its Codebase*. freeCodeCamp. Retrieved from https://www.freecodecamp.org/news/taking-over-an-existing-project/

2.2.2 DESIGN PHASE

First thing to do was reading all the documentation that exist on the Mega Capital project and talk to the ones who participated in the project. It was very useful that the documentation of the project was well prepared and ready; hence, I directly gained an overall understanding.

On the other hand, it was both lucky and unlucky that I had only my supervisor as a person who worked on this project since the last improvement was made in 2020 and most of the contributors moved on to either different departments or different companies in time. However, I was lucky that I am assigned to correct database integration and ad scripting which are the exact areas where my supervisor contributed in the project, so I always had someone that I could ask my questions throughout my implementation.

2.2.2.1 DRAWING DIAGRAMS TO UNDERSTAND CODEBASE

During my research on how to dive into a legacy codebase in analysis phase, one of the methods caught my intention the most. It was drawing diagrams, similar to UML diagrams, in order to gain an overall understanding of the structure of classes, especially the inheritances and dependencies. This method is mainly about looking at the most central concept of a software/product and then wandering out from there. Having the basic functionality in my mind, it was much easier to understand the minor/helper functionalities in the project, which also makes it easier realize the dependencies between classes. On the sixth day of my internship, I created a diagram of the structure of the codebase that helped me keep myself on track throughout the project.[2]

[2] See appendix 1 for codebase structure diagram

2.2.2.2 USING IDE FUNCTIONALITIES TO DESIGN THE STRUCTURE

IDEs provide with great functionalities that help programmers when they get lost in large codebases. One of the features I used all the time was finding all occurrences of a variable or method throughout all the scripts in the project. This makes it super easy to have an understanding of dependencies in the project.

The other feature that helped me a lot was debugger. While line by line debugging helps finding bugs, it also helped me to understand how the code flow between the methods works.

As it is widely used and liked by many unity developers, I used Ryder as an IDE throughout my internship.

2.2.3 IMPLEMENTATION PHASE

In the first week, using what I learnt from the design and analysis phases, I created a review report for the top three games of the company[3]. This is a broad review categorizing different aspects of the game into “Pros”, “To Change” and “Suggestion” areas. The report also contained in-game screenshots, images and comparisons with similar games to illustrate my points clearer. The aim of the report was both to provide feedback to the company for future improvements on the projects and creating a thinking framework for my own responsibilities in my future work in the company.

I also used a method called exploratory refactoring, which is refactoring a function or class to have an insight of what it actually does by breaking it into smaller blocks. It helped me a lot when I came across a method not named or commented well and have a structure making it hard to understand its functionality and side effects.

[3] See appendix 2 for game review report

2.2.4 TESTING PHASE

When I finished investigating the codebase and made sure that I understand the project enough to move on and contribute, I decided to learn the fundamental reasons of specific usages and structures. I started asking some questions like “Why did you implement this part this way?”, “What made you choose this pattern over this one?”. The reason behind this was I was testing my knowledge and the companies decisions for certain implementations.

2.3 SECOND WEEK

2.3.1 ANALYSIS PHASE

After acquiring the fundamental concepts and creating a roadmap in first week, time has come to make the legacy project Mega Capital back to a working condition. This project was a final product of 3 years of work and the last improvement was made in 2020.

After analyzing the current situation of the project and discussing it with my seniors, I realized that there are lot of things in the project needed to be upgraded if we want it a proper working and smooth project.

I noted the necessary things to fix and they were as follows:

- Unity version of the project was 2019, it needed to be upgraded to the latest LTS version of 2021 available at the time.
- There are bugs and errors resulting from unity version upgrade from a very old version, especially database SDK inconsistencies exist.
- Dependency manager of the project needs to be upgraded

In the meantime, I followed the lectures on necessary concepts given by my seniors such as NavMesh controls and Coroutines throughout the week.

2.3.2 DESIGN PHASE

After discussing the problems I pointed in analysis phase with my supervisors, I decided the strategy was to first to upgrade the project to the Unity's latest version of 2019, then to 2020 and 2021 in order. This way, it would be easier to resolve the errors in each turn and predict to causes.

Moreover, I was assigned to a new hyper-casual game that I would create on my own that has a due date of 2 weeks. For this purpose, I read a book on game design to create a better and reliable game. The book is Theory of Fun for Game Design by Raph Koster.

2.3.3 IMPLEMENTATION PHASE

Upgrading the unity version of the project required some steps that needed to be followed in order, so that no dependency conflict would occur. Following these steps was especially crucial with the integration of Google packages such as firebase and external dependency manager.

After upgrading the project version to the last LTS version of 2021, I upgraded the external dependency manager to the appropriate version (1.2.169). Afterwards, I integrated the packages such as firebase.app and firebase.auth.

Moreover, in the last day of the week, I was assigned to resolve a bug with another project of the company, Snakey Maze. The problem was related to the NavMesh, which I refreshed my knowledge by listening to the lectures of my seniors. This helped me resolve the bug quickly so that I delivered to project in a condition “ready to develop” to the responsible team.

2.3.4 TESTING PHASE

When I was integrating the necessary SDKs and upgrading the project packages, I continually faced new errors. I made my testing by cleverly committing each resolve to the source control and trying different versions that can be compatible with the latest changes I made. When something failed, I could always go back to my commits.

Also, with the bug I resolved on the game Snakey Maze, I made my testing through debugging. This helped me easily find out which code line makes the game act unexpected.

2.4 THIRD WEEK

2.4.1 ANALYSIS PHASE

In this week, my task was to develop a new game I was assigned from scratch. I had the GDD of the game and started to plan my development process accordingly.

I decided to use the reference images in the GDD as a template for my game. This was an ASMR hyper-casual genre game, where user input is limited to one touch and aim for pleasant feeling of accomplishment by filling empty spaces. For privacy reasons, I cannot provide more information about the game.

After creating the basic template, I decided to create a basic code structure for the game.

Since the platform for the game was aimed to be mobile, I needed to learn some touch input structures.

Lastly, I discussed different level design ideas with the design team and wanted some correction from the art team because of some weird looking of the object provided in the first place.

2.4.2 DESIGN PHASE

One of the core mechanics of the game was a specific movement pattern of object filling. In order to achieve that, I had to create an indexing algorithm. I used my previous knowledge from my algorithms and data structure classes. I drafted a solution that indexes the necessary object depending on their row and column place in a specific manner.

I also decided that accuracy control would be a good aspect of the game. In that sense, I needed to implement an accuracy calculator algorithm to the project. I drafted an idea that might achieve this by comparing some aspects of user action with the desired one.

In order to achieve “Clear Choices” feature of the game, I needed to implement a mechanism where users can redo their actions. I imagined a solution where I can use linked lists, so that I can manage jumping back and forth easily.

Lastly I designed a simple first scene in Unity that would provide me a testing environment for my algorithms mentioned above.

2.4.3 IMPLEMENTATION PHASE

I started to implement the game beginning from the core mechanic, which is filling. In order to do so, I practiced the linked list idea I created in the design phase and then combined it with the indexing algorithm.

At first, the indexing algorithm worked flawlessly. However, after running some unit tests in the test phase, I realized that my algorithm is causing some unexpected indexing when grid shape is changed. Therefore, I came up with a generic solution where my algorithm would not depend on the shape of the grid, and only required the knowledge of the size and starting point of the grid.

Then I implemented a simple UI bar to test whether the accuracy algorithm worked. As I filled the grid, accuracy bar also filled accordingly with the corresponding percentage. Hence, this feature was also implemented correctly.

After a discussion with the design team, we decided to add a touch feedback whenever user interacts with the screen and takes action. For that purpose, I made my research and learnt this feature is called “haptic feedback”. Then I implemented the feature and left the test until the build phase since it is not possible to test this feature on computer.

Lastly, I added the necessary animations for camera and gameplay and sent the first draft for a test to the game tester of the company.

2.4.4 TESTING PHASE

Firstly I tested the haptic feedback feature that I mentioned in the implementation phase after I created a development/testing build. The result was fine, and the feature was working correctly.

Most of the algorithms that I implemented was reflected on the game as expected without any strange behavior; however, my supervisor suggested a better inheritance of the manager classes I wrote in the project and some better SOLID principles, especially single responsibility. Hence, I refactored my code according to the advice I got. I separated the functionalities into their own methods and similar methods and attributes to their own classes. I also created some interfaces to manage future classes that can implement similar logic.

In this way, my code became much more readable, optimized and following SOLID principles.

2.5 FOURTH WEEK

2.5.1 ANALYSIS PHASE

After the core mechanic is complete and working without errors, this week I needed to design new levels to provide a better game experience to the user. These new levels also would contain the original art prepared by the game artists.

I decided the indexing algorithm to operate more autonomous.That is, it would index the game objects at the beginning of the scene no matter how they are structured in the hierarchy. I also decided to add new features to the game such as a bonus mechanic, a menu implementation and showing the final result to the player by implementing an in-game screenshot taking mechanism.

2.5.2 DESIGN PHASE

At first, my grid manager was indexing the object that it contains in the hierarchy depending on the condition of having them as its children. I planned to remove this dependency and make this more autonomous.

I also discussed the design of the game menu which would provide access to the different levels that are unlocked. This discussion was about how the menu should appear and disappear on the screen.

Finally, I decided to implement a bonus mechanic, where the user is awarded after making a certain amount of correct decisions one after another. I suggest a solution using different events that would fire when the user makes a correct or incorrect action respectively.

2.5.3 IMPLEMENTATION PHASE

I achieved the autonomous indexing of the grid by adding a new class that the objects to be indexed would have, containing a public property that can be manipulated outside by grid manager. With this logic grid manager finds all the objects that has this class and then index them accordingly; hence, removing the dependency of them being children.

For the menu, I merged different level icons on to one panel and give them a button logic. I also make this button logic active dependent to whether the level is reached by the user.

As I planned, I implemented the bonus logic using events. It basically works when certain amount of successful action events are fired by the InserManager class without being interrupted with an uncuccessful action event. Then the player is awarded accordingly depending on the level logic.

2.5.4 TESTING PHASE

The autonomous indexing of the grid worked flawlessly. However when I tested the menu logic, I faced with several problems. The buttons for all the levels were active no matter what. I solved this problem by getting the active scene index count from Unity and then by comparing it to the button order.

Bonus logic worked as expected and rewarded player in each unit test I provided.

As a result, the game was ready for the company to send for a metric test and my task was completed for the week.

2.6 FIFTH WEEK

2.6.1 ANALYSIS PHASE

After the completing the development of a new game, I returned to contribute more to the Mega Capital project. This week, I am assigned to remove the old ad provider for the game (Unity Ads) with a new one (AppLovin).

This process requires the change in the following:

- Integration of new SDK.
- Scripting of AppLovin ad class and keeping the coding structure that previously written.
- After achieving a working structure, improving the abstractions for the previously written ad logic.

2.6.2 DESIGN PHASE

Firstly, I consulted to my supervisor Samet Kurumahmut, who wrote the ad scripts for this project. He suggested me to keep the same standards and design patterns written in the previous version until I achieve working test results and then to improve the logic for a scalable code.

I started reading the previous code which was providing the ads from Unity Ads, and understand the usage of different structures. Then I read the documentation for the AppLovin SDK to find the corresponding implementations.[4]

In order to create AppLovin Ads and test them, we needed a company account on AppLovin. For that purpose, I consulted to my supervisor and created the necessary ad units for the company.

[4] See appendix 3 for Applovin SDK initialization provided in the documentation

2.6.3 IMPLEMENTATION PHASE

I followed the same pattern that was previously written for UnityAds for the project and created another class for AppLovin Ads. After using the corresponding and suggested SDK calls and event handlers, ads were ready to test.

2.6.4 TESTING PHASE

I tested the ads both on my phone and on the test phones of the company and succeded in showing the test ads. This meant that after providing the real ads and playing the game downloaded from a store, user would see the actual ads.

2.7 SIXTH WEEK

2.7.1 ANALYSIS PHASE

After the completing the AppLovin ad integration of the game successfully, my last task for my internship period was to provide an analytics sdk integration and a cleaner asynchronous ad call structure for the game. For analytics, we decided to use AdjustSDK. In order to create asynchronous ad calls, I decided to use async/await structure of C#.

I read the documentation of Adjust and booked a meeting with a former developer of the project in order to provide a clean integration.

2.7.2 DESIGN PHASE

The purpose of the asynchronous ad calls is to provide a better and cleaner event handling of the ads in different circumstances such as when ad failed to show or successfully shown to the user.[5] To achieve this usage, I created another class that would contain the ad result. I also decided to use UniTasks, which is a structure that returns an object of a class that you can later check and perform actions asynchronously.[6]

For the analytics part, in order to keep track of the events in the game, I needed to create global events on the website of Adjust. I consulted this to the former developer of the project and he suggested we leave this for future and only to integrate the SDK to the project and test if it works correctly.

[5] See appendix 4 for example async ShowRewardedAd() method
[6] See appendix 5 AdManagerBase.cs using UniTask

2.7.3 IMPLEMENTATION PHASE

I implemented the new AppLovin ad script to the project with the new async/await structure of C# as discussed in the design phase. I also implemented the edge case scenarios after the advice of my supervisor and the project had a complete and working clean ad structure with a new ad provider.

I successfully integrated the analytics SDK provided by Adjust to project by using its Unity integration.

2.7.4 TESTING PHASE

After the implementing the asynchronous ad logic, I had to test the ads one more time to check if they still work. The result was successful.

I left the testing of Adjust Analytics SDK to the developer team as adviced since I came to the end of my internship period.

3 DESCRIPTION OF THE COMPANY

Company Name: Simsoft Bilgisayar Teknolojileri Ltd. Şti.

Company Location: Üniversiteler Mah. 1596. Cadde 6.AR-GE E Blok 2.Kat Ofis No: 12 Beytepe -Çankaya Ankara

Website: https://www.simsoft.com.tr

Organizational Structure of the Company: hierarchical organization

Number and Duties of Engineers Employed: 1 Senior Game Developer, 1 Manager

Main Area of Business: Simulation systems and serious game applications

3.1 A BRIEF HISTORY OF THE COMPANY

The company was founded in 2006 in order to develop military software, simulation systems and serious game applications. Simsoft was enlisted as one of the "50 Fastest Growing Technology Companies of Turkey" by Deloitte in 2015,2017,2019 and 2020. The company was also in the list "500 Biggest Service Exporter of Turkey" by Turkish Exporters Assembly in the years 2017 and 2018. Since 2016, Simsoft is a "Strategic Solution Partner" of ASELSAN,a Turkish defense corporation and one of the major contractor of Turkish Armed Forces.

Simsoft's head office is at METU Technopolis in Ankara Turkey and also has offices in Bilkent Cyberpark, Hacettepe Technopolis, Technopolis İstanbul and Florida USA.

3.2 METHODOLOGIES AND STRATEGIES USED IN THE COMPANY

Since my internship was conducted face-to-face, I observed how the company spends a business day throughout six weeks. I performed my internship in the game development department of the company, which operated similar to a startup company. The communication between the game designers, artists and programmers is usually direct and supervised by the lead game designer. Hence, it creates a warm working environment with fast and effective communication. Company uses the general game design processes while developing a new game and online issue tracking tools like Jira.

4 CONCLUSIONS

4.1 SUMMARY

My summer internship period lasted six weeks, starting with a discussion on how I can make the best out of my internship period. The company was familiar and expert on developing games after years of experience and I stated my interest in working on a large and complex project in order to improve myself by performing more than an ordinary intern.

My main aim was to learn how to write optimized and readable code as a professional developer while both extending and improving the previously written code and providing a clean workspace for future developers. In that perspective, I spend my first weeks on understanding the professionally written code on the biggest project of the company, Mega Capital. In the third and fourth week, I applied my knowledge in a completely new game and delivered a finished game from scratch as required. In the last weeks I had the chance to apply new patterns that I learnt by improving the ad scripts of the Mega Capital game and integrated third-party analytics and ad SDKs into the project.

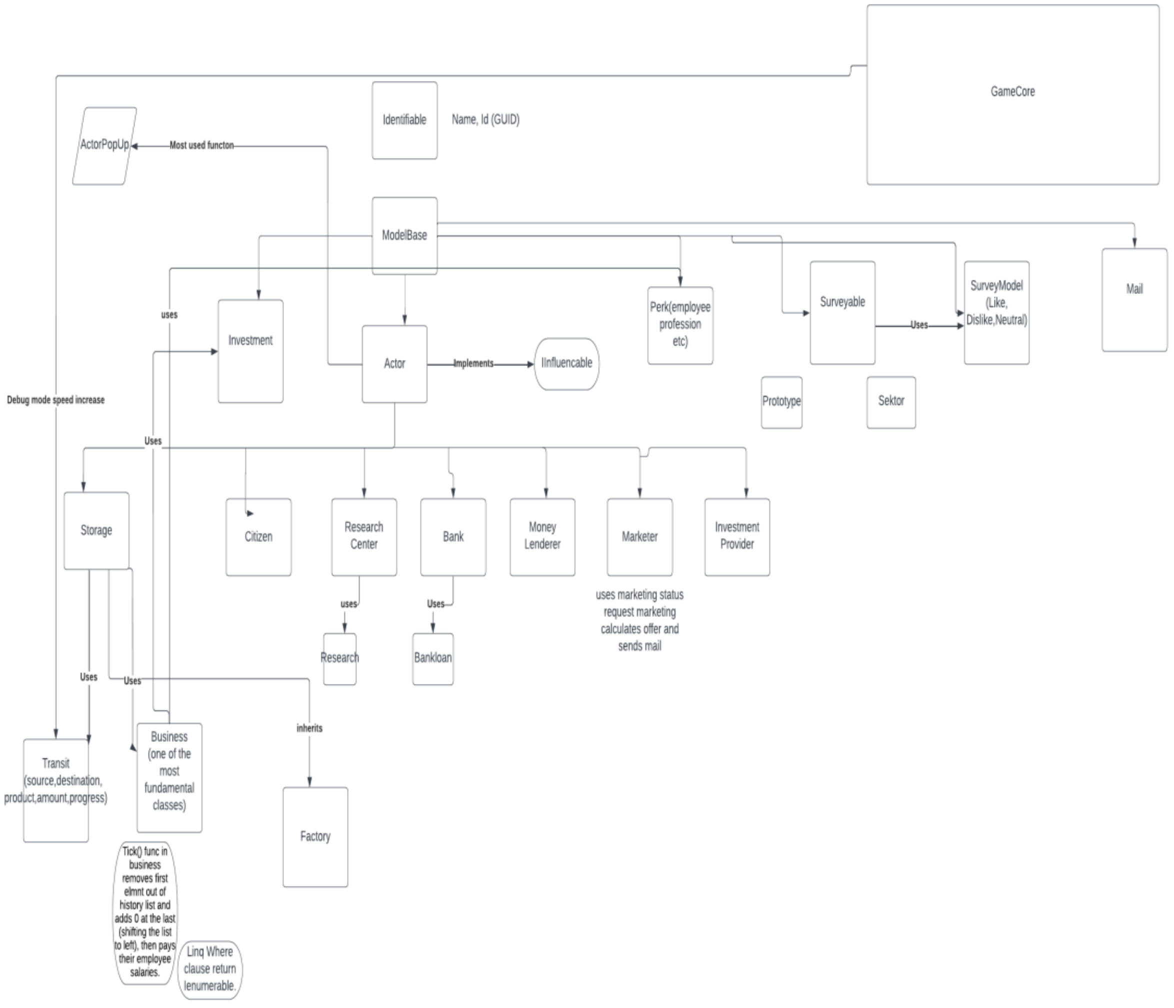
At the end of my internship, I was a competent developer who understands the agile software development environment and ready to apply my knowledge in the business area.

4.2 DISCUSSIONS AND COMMENTS

Throughout my internship, my supervisors helped me and encouraged me anytime I had a question. It was a real privilege to work with those people and contribute to the great things they create. At the end, I was more of a junior developer than an intern, which was my aim from the beginning and Simsoft created this environment for me.

5 APPENDICES

[1] MEGA CAPITAL CODEBASE STRUCTURE DIAGRAM


GameCore
Identifiable
Name, Id (GUID)
ActorPopUp
Most used functon
ModelBase
Mail
uses
Investment
Actor
Implements
IInfluencable
Perk(employee profession etc)
Surveyable
Uses
SurveyModel (Like, Dislike,Neutral)
Prototype
Sektor
Debug mode speed increase
Uses
Storage
Citizen
Research Center
Bank
Money Lenderer
Marketer
Investment Provider
uses
Uses
Research
Bankloan
uses marketing status request marketing calculates offer and sends mail
Uses
Uses
inherits
Transit (source,destination, product,amount,progress)
Business (one of the most fundamental classes)
Factory
Tick() func in business removes first elmnt out of history list and adds 0 at the last (shifting the list to left), then pays their employee salaries.
Linq Where clause return Ienumerable.

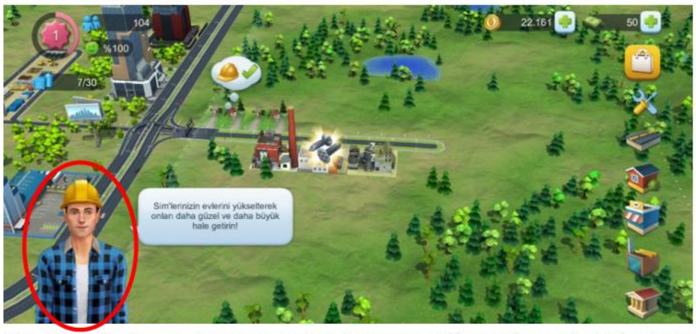
[2] GAME REVIEW REPORT

Author: Furkan Koçak

Mega Capital Review

✓ **Pros:**

- Player has different experience with the same content (different products, satisfaction calculation of a new product through survey and different decision making)
- Not too complicated to understand, good balance between thinking and playing.
- Choices are optional, risk and reward depends on user decisions. Player can progress through casual playing or play better by his skills. This increases retention.
- There is no punishment as far as I played. This is good especially we don't want user to be frustrated. Instead reward must be scaling according to the risk and this should be clearly shown to the user.
- There is a good walkthrough at the beginning, making it easier to understand and have fun from the game. As a suggestion instead of an image, an animation of a secretary/advisor would be better. See the example below from SimCity.



Sim'lerinizin evlerini yükselterek onları daha güzel ve daha büyük hale getirin!

- The idea of using an arrow to switch between different households' opinions is a really good idea for playability. There can be an overall satisfaction of product added for ease of play. Moreover, I liked the detail of different doorbell sounds for satisfied/unsatisfied households.
- There are no lags and transitions are smooth.
- There are reminders/notifications to alert the user they haven't been producing for a while. This helps the user to take the correct action.

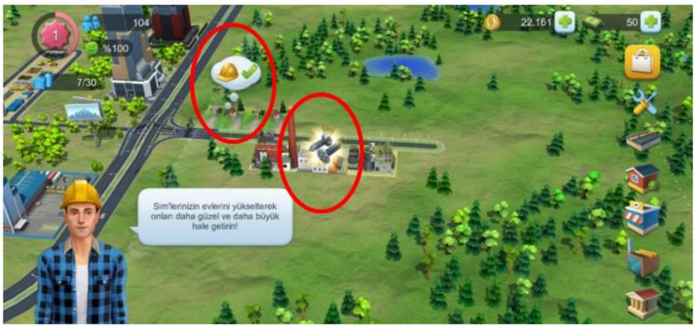
✗ **To Change:**

- After the walkthrough we need a clear first goal, what should I do now? I paused for a couple of seconds trying to understand what is next.
- Too many ads while trying to understand a game like this. Retention must be checked through statistics whether the ads frequency is at a good level between revenue and retention.

Author: Furkan Koçak

Suggestions:

- Household survey can contain less text, in fact dislike part can be completely removed and replaced by a happy face/upset face and a number. For example 5 like,2 neutral,1 dislike for a household can be replaced with +4 happy face.
- Hint video has low quality, we might consider fixing it.
- When a prototype is ready, instead of sending it to the mailbox, there can be an icon popping up on the building showing its ready, and through clicking it the interface opens. See the example below from SimCity.


1
104
%100
7/30
22.161
50
Sim'lerinizin evlerini yükselterek onları daha güzel ve daha büyük hale getirin!

- When we gain money (which I believe the most satisfactory part for the user) and experience, this should be clearly shown to the user through animations and sounds so that user would be more satisfied to continue.
- The PlayStore thumbnail of Mega capital is pretty good. As a suggestion, similar games such as Landlord, SimCity, SimCompanies etc. have the name of the game as a text on the thumbnail which I believe attracts more interest to download. Mega Capital text can be added to the thumbnail.
- Game speed increase should be an option for skillful players who can control multiple things at a time and they should be rewarded. (just like in Sınıfta simulation)

Author: Furkan Koçak

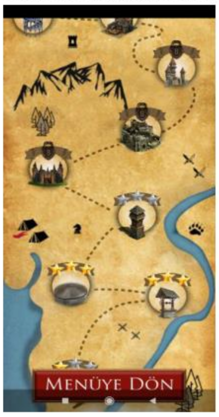
Archers: War of Anatolia Review

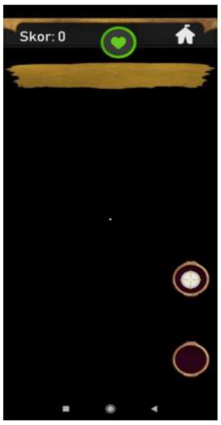
✓ **Pros:**

- Player has different experience with the same content (different levels, continuation of a story)
- Skillful players are rewarded through multiplayer games by winning. In singleplayer, they should be rewarded depending on precision and timing.
- Easy to understand.
- Choices are optional, risk and reward depends on user skill and decision. Player can progress through casually hitting the body or play better by using proper arrow type and aiming to head. This increases retention.
- There is no punishment as far as I played. This is good especially we don't want user to be frustrated. Instead reward must be scaling according to the risk and this should be clearly shown to the user.
- There is a good walkthrough at the beginning, making it easier to understand and have fun from the game.
- The idea of using different arrow types is good.
- There are reminders/notifications to alert the user they haven't been producing for a while. This helps the user to take the correct action.

✗ **To Change:**

- There are lags and some animations are not smooth (Especially when camera is following the arrow after release).
- My game got stuck at around level 4. See the images below. These kind of bugs are extremely critical for a game's retention.

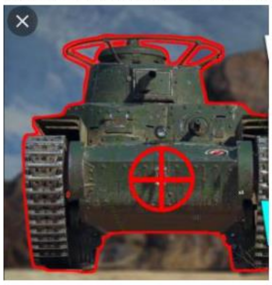




Author: Furkan Koçak

Suggestions:

- Environmental assets can be better to reflect the ambience.
- Instead of only clicking a button, it would be better for me to press hold and release to get the sense of releasing an arrow.
- When we get the dodging an arrow training, there should be a moment of freeze when the arrow is coming to us, saying click this button to dodge. This way user would better understand that he needs to dodge it just before the arrow is released by the enemy.
- There can be an indicator for the user showing normal arrow cannot penetrate the material being aimed such as a red cross. See the example below from World of Tanks.



Author: Furkan Koçak

Sınıfta Review

Pros:

- Player has different experience with the same content (different behaviors from students each level, free movement and eye contact options)
- Game speed increase is an option for the user.
- Not too complicated to understand.
- Choices are optional, user decisions are directly affecting the lecture result. Player can progress through casual playing or play better by his skills.
- Tutorial is fairly easy to follow, making it easier to understand the simulation
- I liked the idea of having 3 different parameters namely concentration, entertainment and knowledge and trying to maximize them.
- There are no lags and transitions are smooth.
- There are reminders/notifications to alert the user.
- Student animations are realistic and looking good.
- Different classroom environments are designed meticulously.

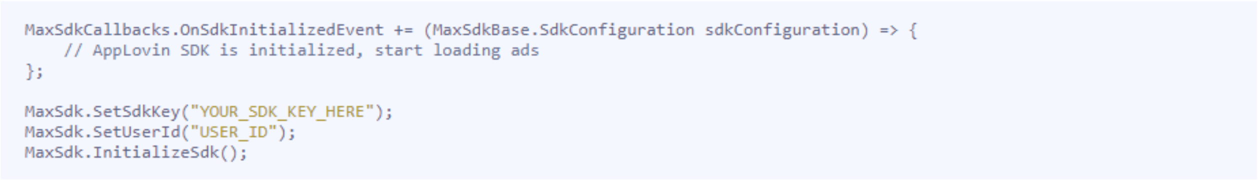
To Change:

- Tutorial must be encouraged more (maybe compulsory) at the first play. Because it is extremely hard to understand what to do without gameplay tutorial.
- Head movement is too sensitive. I think vertical movement (head up and down) should be much slower.

Suggestions:

- There is a Turkish language setting but it is hard to find. Through location, this must be set default for the user if she/he is Turkish.
- Tasks should appear in a better looking style. I couldn’t see the tasks appearing at the right because they have too little font and have an extremely simple UI.
- There should be different sound effects for student moods when they appear on top of their heads.
- There should be approving sound effects when the teacher takes the correct action.

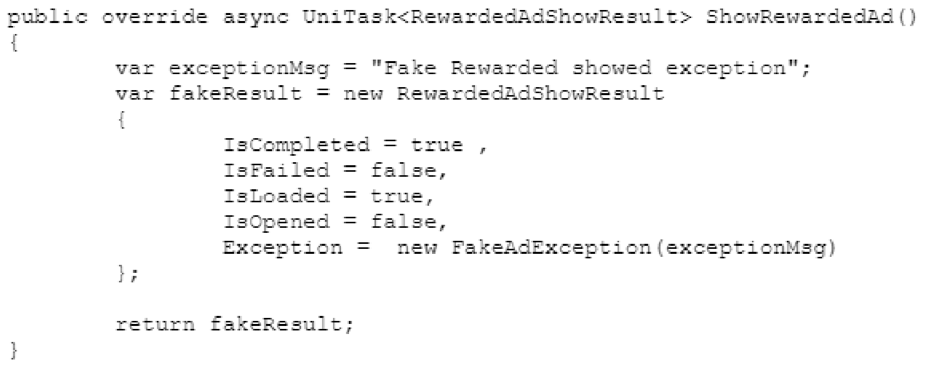
[3] AppLovin SDK Initialization



```
MaxSdkCallbacks.OnSdkInitializedEvent += (MaxSdkBase.SdkConfiguration sdkConfiguration) => {
    // AppLovin SDK is initialized, start loading ads
};

MaxSdk.SetSdkKey("YOUR_SDK_KEY_HERE");
MaxSdk.SetUserId("USER_ID");
MaxSdk.InitializeSdk();
```

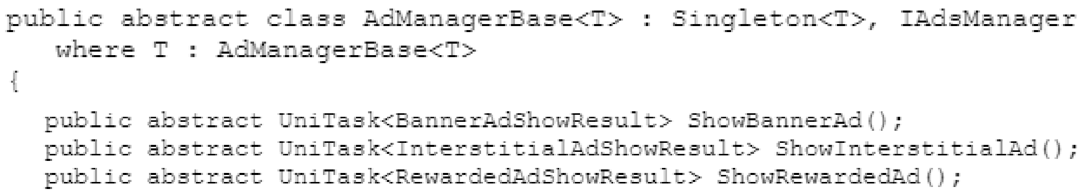
[4] An example for async ShowRewardedAd method



```
public override async UniTask<RewardedAdShowResult> ShowRewardedAd()
{
        var exceptionMsg = "Fake Rewarded showed exception";
        var fakeResult = new RewardedAdShowResult
        {
                IsCompleted = true ,
                IsFailed = false,
                IsLoaded = true,
                IsOpened = false,
                Exception =  new FakeAdException(exceptionMsg)
        };

        return fakeResult;
}
```

[5] AdManagerBase.cs using UniTask structure



```
public abstract class AdManagerBase<T> : Singleton<T>, IAdsManager
   where T : AdManagerBase<T>
{
  public abstract UniTask<BannerAdShowResult> ShowBannerAd();
  public abstract UniTask<InterstitialAdShowResult> ShowInterstitialAd();
  public abstract UniTask<RewardedAdShowResult> ShowRewardedAd();
```