Çankaya University

Department of Computer Engineering

Project Report

for

CAMPUS KART RACING GAME

(CKRG)

Version 1.0

Prepared By:

Esra ŞAHİN

Ozan ÇETİNER

Buğra DOĞAN

N. Cem ALTUNBULDUK

Due Date: 04 January 2021

Change History

This document include the first version Software Requirements Specification document of the project named Campus Kart Racing Game. It was created on December 3, 2020.

This document include the first version of SDD document. The document was prepared on 24 December 2020.

Table of Contents

Figure List of SRS

Figure List of SDD

1.LITERATURE SEARCH

Abstract

Today, the video game industry has become one of the biggest sources of entertainment for people. The gaming industry has surpassed many industries and has generated $ 150 billion in revenue in 2020. This means a significant increase of 9.3% according to 2019 data.[1] Many game development companies want to keep their income level at the highest level by releasing their games with cross-platform and online features. Today, hundreds of games provide cross-platform support. [2] 56% of the players prefer to play games online. And they spend an average of 7 hours playing with others online – an hour more than playing in person.[3] In this review aims to inform multiplayer racing games, artificial intelligence in games, game engines and cross platform games.

Keywords:
Computer and Mobile games,Car Racing Games,Cross Platform Games,Online Games,Multiplayer and Singleplayer Games,Game Engines,Artificial Intelligence,Networking

Öz

Günümüzde video oyun sektörü insanlar için en büyük eğlence kaynaklarından biri haline geldi. Oyun endüstrisi birçok sektörü geride bıraktı ve 2020'de 150 milyar dolar gelir elde etti. Bu, 2019 verilerine göre % 9,3'lük önemli bir artış anlamına geliyor. [1] Pek çok oyun geliştirme şirketi, oyunlarını çapraz platform ve çevrimiçi özelliklerle yayınlayarak gelir düzeylerini en üst düzeyde tutmak istiyor. Bugün yüzlerce oyun çapraz platform desteği sağlıyor. [2] Oyuncuların% 56'sı çevrimiçi oyun oynamayı tercih ediyor. Ve başkalarıyla çevrimiçi oyun oynamak için ortalama 7 saat harcıyorlar - tek oyunculu oynamaktan bir saat daha fazla. [3] Bu inceleme, çok oyunculu yarış oyunlarını, oyunlarda yapay zekayı, oyun motorlarını ve çapraz platform oyunlarını bilgilendirmeyi amaçlamaktadır.

Anahtar Kelimeler:

Bilgisayar Ve Mobil Oyunlar,Yarış Oyunları,Çapraz Platform Oyunlar,Çevrimiçi Oyunlar, Çok Oyunculu ve Tek Oyunculu Oyunlar,Oyun Motorları,Yapay Zeka,Ağ İletişimi

1.1. Introduction

The racing game is a type of video game in which players compete with land, air or sea vehicles. The history of racing games in the world dates back to the 1980s (such as Spy Hunter series (1983), Autoduel (1985)) and has always been the focus of attention. With the advent of computer technology, computer games have also gained an important place in our lives. It has even become an integral part of everyday life for many people. As computer languages, internet and visual effects developed over time, computer games were made realistic and more complex.These games, which started to be of higher quality, attracted the attention of wider audiences. A game that comes to mind when it comes to computer games and has the largest player mass in the world is racing games. Among these games, the most popular and one of the oldest games is the car racing game.

The Turkish game market is developing slowly and opening up to the world. A few top selling companies have built their expertise on combat, RPG or action. However, there has not been much work done on racing games yet. By evaluating this gap, we want to bring an untouched genre to the sector and to create a fun competition environment among universities for domestic users.

This article explores the dynamics of racing games and important sub-issues that need to be developed to make a racing game. At the same time, the article highlights the reasons for choosing the right game engine and why cross platform games are preferred.

1.2. Cross Platform Development

When you build a native app you have to construct a separate one for Android and a separate one for Windows(PC) for this project, each using a unique language. For example if you want to code a program for Apple/iOS; programming language must be Objective C or Swift. For Android you have to use Java etc. For Windows Phone you have to use C# or XAML. As a result for each platform you have to use different languages and that might cause few problems in a one particular project. [4]

1.One of these problems is financial consumption. For each platform buying, creating and keeping it sustainable will cause a lot of expenses.

2.Another problem is programming native games requires at least 2-3 different programming languages.

3.Third, and lastly, another big problem is the ability to consistently maintain stability and uniformity. Since each platform has its own user interface, standardized components and features, applications will not be uniform from platform to platform and will create differences in user experience depending on the device.

Most people have more than one device type, so their experience will be different when they switch to an Android device using your app on iOS.

Gamers today have a wide variety of devices, from popular smartphones to high-end consoles. The best way to ensure your game reaches the widest possible audience is to create games that run on all major devices used.

1.3. Game Engines

Game engines are one of the most important tools for game developers. Game engine is a type of software that includes various tools for game development on various platforms. These tools are 2D / 3D Renderer, Physics Engine, Scripting Support, Audio Engine, Artificial Intelligence etc. can be listed as.[5]

The Most Popular Game Engines

- Unreal Engine
- Unity
- Cocos2d-x
- CryEngine
- AppGameKit
- Godot

1.3.1 Why Unity Game Engine ?

There are many reasons why Unity game engine is preferred. Although it is primarily designed for 3D games, it can be developed in 2D games. As it is widely used, there is access to a large number of free videos explaining how to use Unity software. This means you will not pay for the training. Another feature is the cross platform option. The game developed for one platform can be easily converted for other platforms. The main platforms that Unity supports are: Windows, MacOS, Linux, Android, IOS, Xbox One, Wii U, Play Station 4. In addition, the Unity game engine is more convenient than other game engines, consumes less resources. And Unity is free, despite all the possibilities it offers. Unity game engine is used by half of mobile game developers, while more than half of the VR games on the market are developed with Unity.[6]

1.4. Game Mechanics in Racing Games

Important part of racing game is of course driving the car. But diverse games proposes you a different type of racing gameplay experience.

-NFS: Underground (EA Games, 2003) includes a special winning condition. A player must finish the race once from the start to win this game. This is the winning rule even if the multiplayer mode is chosen. This behavior has been developed to increase the difficulty level in the game to a higher level. NFS, as in the name of the game, was used to increase the stress, and different cars were selected from different countries to increase the competition factor. The vehicles in the game are available for modification by the player. [7]

Split / Second (Disney, 2010), on the other hand, caused a frenzy when it launched after NFS, offering many more features. The player faced different modes outside of the race. For example, only drift mode or correct driving mode showed people that there are new areas where they can invest in car games outside of racing. All of these features are presented under the name of 'power play'.

Blur (Activision, 2010) laid the foundations for a different genre that combines racing games and arcade games. The race is also completed by fighting other cars. In this race, you can bleed extra speed, use shortcuts, blow up other players' cars or blow up the wheels of those behind you.

Unlike other racing games, cars that have won previous races can start the race more advantageously. All of these features have made Blur more popular on Playstation since its release.

One of the most important factors in racing games is speed. The more we increase the game speed, the greater the excitement, interest and difficulty for the player. In most racing games, beginner players are not given very fast cars. However, as the player gains certain achievements or levels up, they can use new features and cars. This makes the game more rewarding and increases interest.

Players generally prefer environments in which they can make their personal choices. The customization of the car is an important part of racing games. In addition, car models that have been transferred from real life to the game are another popular feature preferred by the players. In addition to these, the feature of ranking tables, comparison of scores and achievements brings more interest and competition.

Classic style speed racing isn't the number one factor in racing games. Some games provide stunt moves, car crashes, explosions other than racing in the game. For example, Burnout

(Criterion Games, 2001) has many options besides competition, such as these features. Apart from making racing games just about racing, it also increases the fun level of the game. [8]

The product, which occurs not only in racing games but also in all games in general, is a combination of many factors and game mechanics. There is no formula for making the game with the highest sales numbers. But you have a chance to increase your chances of success by combining different types of gameplay.

1.5. Artificial Intelligence in Racing Games

The aim of artificial intelligence in racing games is not to beat the player, but to give the player a challenging and realistic experience.
Artificial intelligence uses a variety of techniques that can simulate the driving of cars to achieve this goal.Waypoint System, one of these techniques, is the simplest way to move the car on a certain road. These waypoints are stored in an ordered list and the car is moved between the positions of these waypoints. Another system is the trigger detection system. It supports detecting a specific game object when it is within the vicinity of another game object. And it can give commands such as turning or accelerating the car according to the detected objects.

1.6. Conclusion

In this literature, history of racing games,overview of cross platform games, encountered problems in cross platform games, game engines, game mechanics used in racing games and their examples are examined with details. Cross platform games continue to develop with high revenues and they must be able to be played via different platforms such as PS4, computer. One of the problems of cross platform games is platforms that have their own user interfaces. Game engines are important for users to develop games and have many properties.There are some techniques of artificial intelligence used in racing games. Racing games may have many game mechanics. If a racing game has many gameplay then the game is likely popular.

2. SOFTWARE REQUIREMENTS SPECIFICATIONS

2.1. INTRODUCTION

The document is a review of the Software Requirements Specification (SRS) document of the CAMPUS KART RACING GAME project.

2.1.1 Purpose

This document provides information about the technical details of the CAMPUS KART RACING GAME (CKRG) project.

It also includes definitions of all functions that run on CKRG, specific requirements, limitations, user characteristics.

2.1.2 Scope of Project

CKRG is a cross-platform racing game with single player and multiplayer modes.The aim of the project is to provide a competitive racing experience among players. Players can join the game on mobile and computer. Also players can compete against artificial intelligence in single player mode or they can compete with other players in multiplayer mode if they wish.

2.1.3 Glossary

Expression	**Meaning**
CKRG	Campus Kart Racing Game
SRS	Software Requirements Specifications
SP	Single player mode is a game mode played by only one player.
MP	Multiplayer mode is a game mode in which multiple players can play together at the same time.
AI	Artificial Intelligence

CPG	Cross Platform Games
GE	Game Engine
Use Case Diagram	A type of diagram in UML that represents the user's interaction with the system.
IEEE	Institute of Electrical and Electronics Engineers
SDD	Software Design Document
NPC	Non Playable Characters
OS	Operating System
PC	Personal Computer
RAM	Random Access Memory

2.1.4 Overview of Document

This document gives a general description of CKRG. Also, this document is prepared in accordance with the IEEE Std. 830-1998, IEEE Recommended Practice for Software Requirements Specifications. [9]

This SRS document of our project consists of 7 sections. In the first part, there is general introduction information of the project. Overall description section is the 2nd section. Detailed explanations about artificial intelligence can be found in section 3. In 4 chapters, the requirements required for this project are explained clearly. Project planning and references information is at the end of the document.

2.2. OVERALL DESCRIPTION

This part defines the general factors that affect the CKRG and its requirements. To simply make it more understandable.

2.2.1 Product Perspective

CKRG is a cross-platform racing game developed with the Unity game engine. The game consists of two modes. These are the single player mode that competes against artificial intelligence. The second mode is the multiplayer racing mode where you can compete with real players. The user uses kart type vehicles. And it can use customizations such as adding new parts and changing colors as desired.

2.2.2 Memory Constraints

Devices must have and have at least 512MB of RAM.

2.2.3 Product Functions

There are many options available to the player in the product. These options are explained in detail with diagrams.

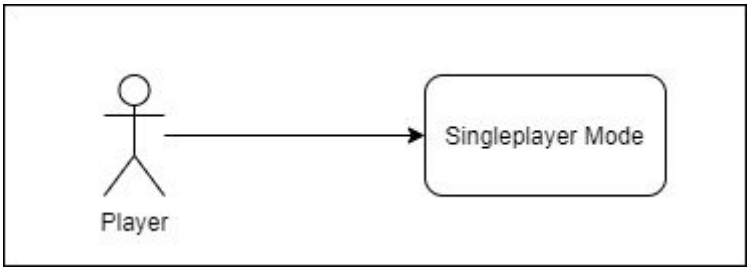

Figure 1: Singleplayer Mode.

Singleplayer Mode: It is the game mode in which the player will compete against artificial intelligence.

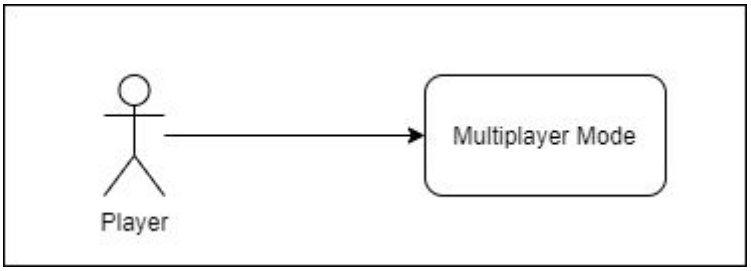

Figure 2: Multiplayer Mode.

Multiplayer Mode: It is the game mode in which the player can compete with other real players.

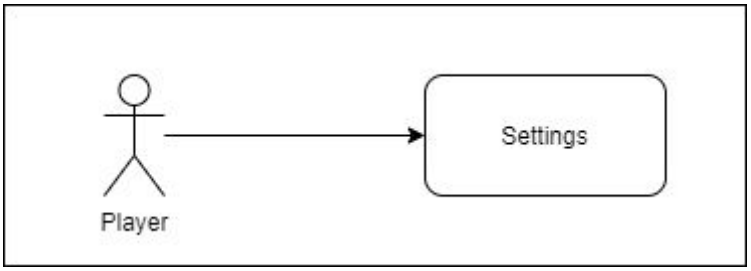

Figure 3: Settings that the user can set.

Settings: It is the menu where the user can make settings such as texture quality, loudness, resolution.

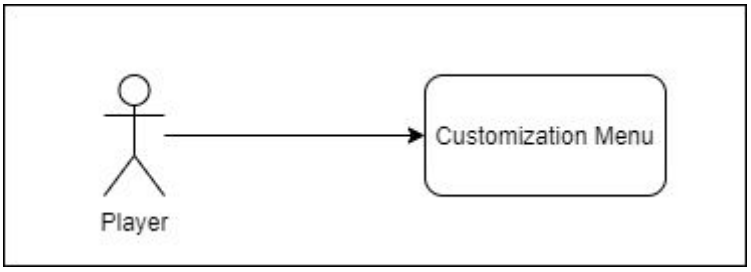

Figure 4: Customization Menu.

Customization Menu: It is the menu where the player can customize their vehicle. It can use customizations such as color change, add parts, etc.

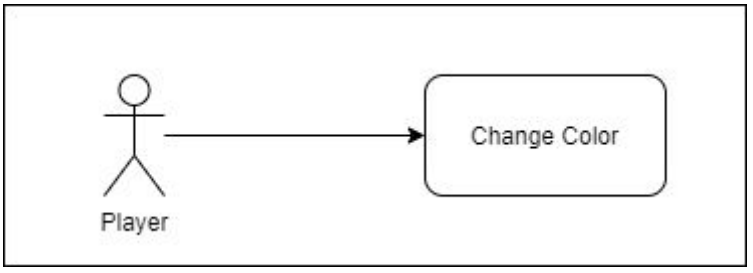

Figure 5: Car Color Changing.

Change color: Players can change the car color.

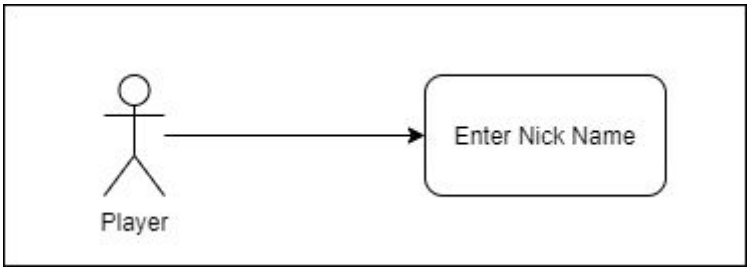

Figure 6: Entering Nick Name.

Enter Nick Name: The player can set the nickname that will appear in the game.

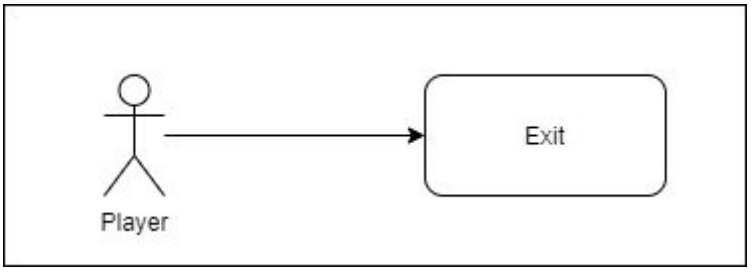

Figure 7: Exiting the Game.

Exit: The player can exit the game.

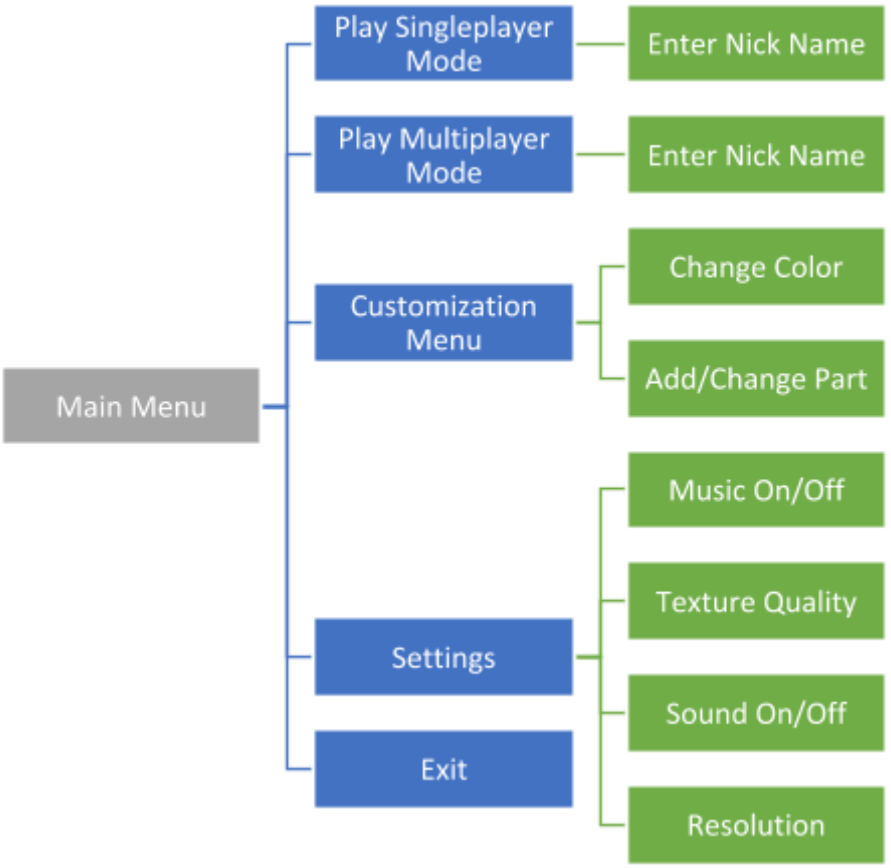

Figure 8: CKRG Functions

2.2.4 User Characteristics

The user who has knowledge about mobile or computer games should also know the English language.

2.2.5 Constraints

In this project, requires minimum Android 4.4+ for mobile devices, Windows 7 and higher for personal computers.Also, an internet connection is required for multiplayer mode.

2.2.6 Assumptions and Dependencies

We assume that the game server can handle at least 1.000 simultaneous connections for multi-user game.

2.3. ARTIFICIAL INTELLIGENCE

2.3.1 Introduction

In this part of the document, you will see how artificial intelligence and NPCs work and how they interact with each other. Artificial Intelligence is getting more and more talked about both academically and technologically. Game artificial intelligence continues to develop depending on all these other developments.

Looking at today, many progress has been made in artificial intelligence. Health sector, voice assistants, e-commerce, autonomous vehicles and communication are among the areas of use. In our age, there is no place where artificial intelligence is hardly used. In digital games, the use of deep learning model artificial intelligence and rule-based artificial intelligence, which are among the artificial intelligence techniques, has become widespread.

To understand how artificial intelligence will become more intertwined with games, it is necessary to look at the history of the artificial intelligence and the game industry. From the earliest days of the game industry, developers have been striving for artificial intelligence to act like a human and create a game world from scratch without the need for a real person. Because it is thought that an active learning artificial intelligence will create games of superhuman beauty. For example, in a war game, enemy soldiers who are constantly hit on the head see this situation and come wearing more robust helmets in the following sections. In another example, the game automatically increases the values of the players of the opponent team and offers you a more challenging game according to your playing well in the football match game. Another contribution of artificial intelligence to the game industry is in graphic design. It can analyze the pictures recorded on the computer and process them on the game and thus reveal more realistic games.

To explain the artificial intelligence technology in games in a simpler form, all the events that occur outside of you while you are playing are made by this technology. For example; When playing games, a character passes by us or enemy soldiers see and shoot you in a war game.

2.3.2 Non Playable Characters

NPCs will be active in single player mode and will be managed by artificial intelligence. In online mode, real-time users will compete instead of NPC. So no matter which mode you enter the game, you will be able to find a character that can compete.

2.3.3 NPC Interaction with Players

It is the name given to the characters that we have seen in online games, moving from time to time and standing still from time to time. "NPC" are "BOT" services that are used for item purchases and placed by game developers so that you can receive the rewards offered by the task and the game.

In single player mode, the user will compete against these NPCs. With the help of AI, NPCs will detect when a car arrives near them and act accordingly. For example, when the user's tool NPC gets too close to the vehicle, the NPC may understand this and try to move away.

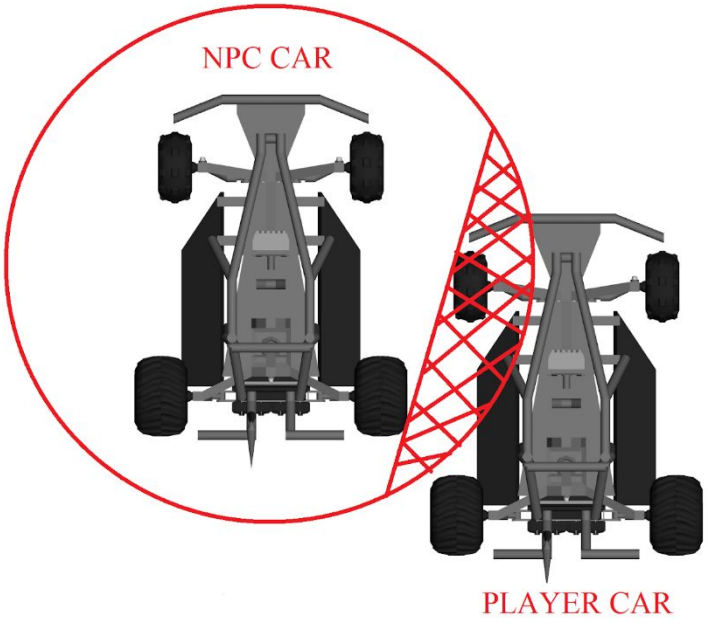

Figure 9:Non Playable Character Car vs Player Car.

2.3.4 Pathfinding

The Pathfinding method is the method that you can move freely between two points. It gives you more options for determining the destination and feels freer.

When the race starts, each NPC car will first calculate the path to a destination and then follow it. The first part of the algorithm is called 'pathfinding', the second part is called 'path tracking'. The scope of our changes was mostly limited to the latter and we avoided making changes (other than minor adjustments) to the pathfinding algorithm. This is because these methods are shared among all the different object classes in the game.

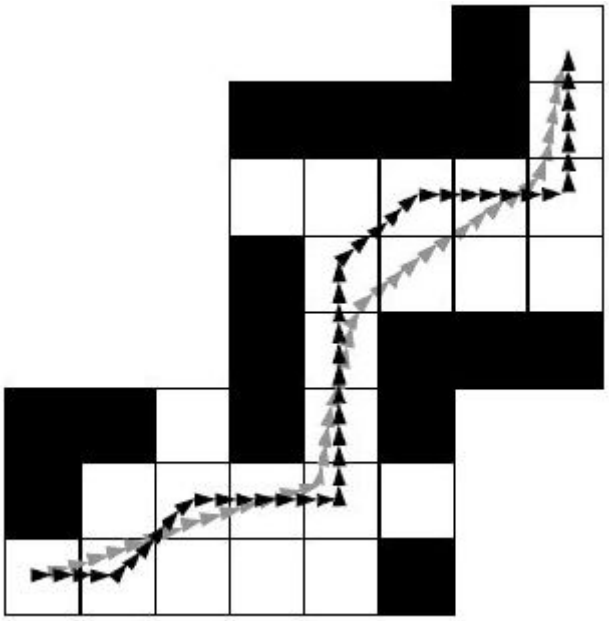

Figure10:Pathfinding in grid system.

2.3.5 Curvatures and Bumps

Curved areas and realistic bumps in car racing, along with bends, maximize the gaming experience. If the control is not maintained correctly when driving at high speed, the vehicle may become unusable. For this reason, the curvatures and bumps on the racing fields are minimized. While doing this, real go-kart tracks have been researched and examined in detail.

2.3.6 Overtaking the Car

One of the important factors to win in racing games is successful overtaking. For this, we organize systems that will create overtaking situations for the player from the race track to the structure of their cars. In the next stage, the places of the checkpoint and final lines are chosen properly for overtaking. It is referred to as "the Field of Opponents", one of the subfields of artificial intelligence.

2.3.7 Driving the Car without Going off the Road

To drive in the track direction without any collision with the track edges, we must have repelling as well as attractive forces spread all over the track to keep the car on track. The repelling forces are concentrated on the edges of the track while the attractive forces are spread along the preferred navigational path. For example if our controller wants to follow the middle of the track throughout the race, the attractive forces are placed in the middle of the track along the length of the whole track.

2.3.8 Improvements and Additional Details

- A menu that will launch the game quickly. Two simple options: Single Mode and Multiplayer mode. After the mode is selected, the countdown for the race starts.

- Sound effects for acceleration, braking, crash, race start countdown and finish.

- One way to keep number of laps and show them during the race so that we know how many laps passed while the race was going on and when it ended.

- A timer that shows how many minutes and seconds have passed since the race started.

- Win / Loss screen and ranking chart after the race is over, based on what the player or computer has won.

These features must be available in the final publishable version of the game.

2.4. SPECIFICATION OF REQUIREMENTS

2.4.1 External Interface Requirements

2.4.1.1 User Interfaces

The user interface will be worked on Windows and Android operating systems.

2.4.1.2 Hardware Interfaces

Minimum system requirements that can run the game:

1. Graphic processor supporting minimum OPENGL ES 2.0 or higher for mobile devices.
2. Graphics processor that supports minimum DirectX 9.0 and higher for personal computers.

2.4.1.3 Software Interfaces

Since this application is a cross platform application, it will need an Android version 4.4 or higher or Windows 7 or higher version.

2.4.1.4 Communications Interfaces

Wired or wireless internet connection is required for multiplayer game mode.

2.4.1.5 Performance Requirements

At least 30 FPS is aimed for a smooth gaming experience. Also, for a competitive multiplayer mode, the internet ping values should be below 100ms.

2.4.2 Software System Attributes

2.4.2.1 Portability

- The game has been developed for Windows and Android operating systems.
- The game supports both 32 and 64 bit processor architecture.

2.4.2.2 Performance

- The game frame rate must be 30 frames per second.
- Instead of real time lighting system, baked lighting system should be used.
- Objects outside the camera angle should not be rendered.
- Level of detail of objects should be changed according to distance between object and the participant.

2.4.2.3 Usability

When a player will open the game,he/she will see a simple interface.The game is easy-to-use.

2.4.2.4 Adaptability

The game can be played on computers and smartphones only.

2.4.2.5 Safety Requirements

Playing games for a long time (looking at the screen) can cause illness such as eye strain.

2.5. PLANNING

The following sections give information about this project planning.

2.5.1 Team Structure

The four members of the CKRG team are Buğra Doğan, Esra Şahin, Ozan Çetiner and N. Cem Altunbulduk.

TASK	**MEMBERS**
3D Modelling	Team
Game Logic	Buğra DOĞAN, Cem ALTUNBULDUK
Game Music	Esra ŞAHİN
Development of Artificial Intelligence	Buğra DOĞAN, Cem ALTUNBULDUK
Game Animations	Ozan ÇETİNER
Lighting	Buğra DOĞAN, Esra ŞAHİN
Sound Effect	Team
Menu Design	Buğra DOĞAN, Ozan ÇETİNER

2.5.2 Estimation

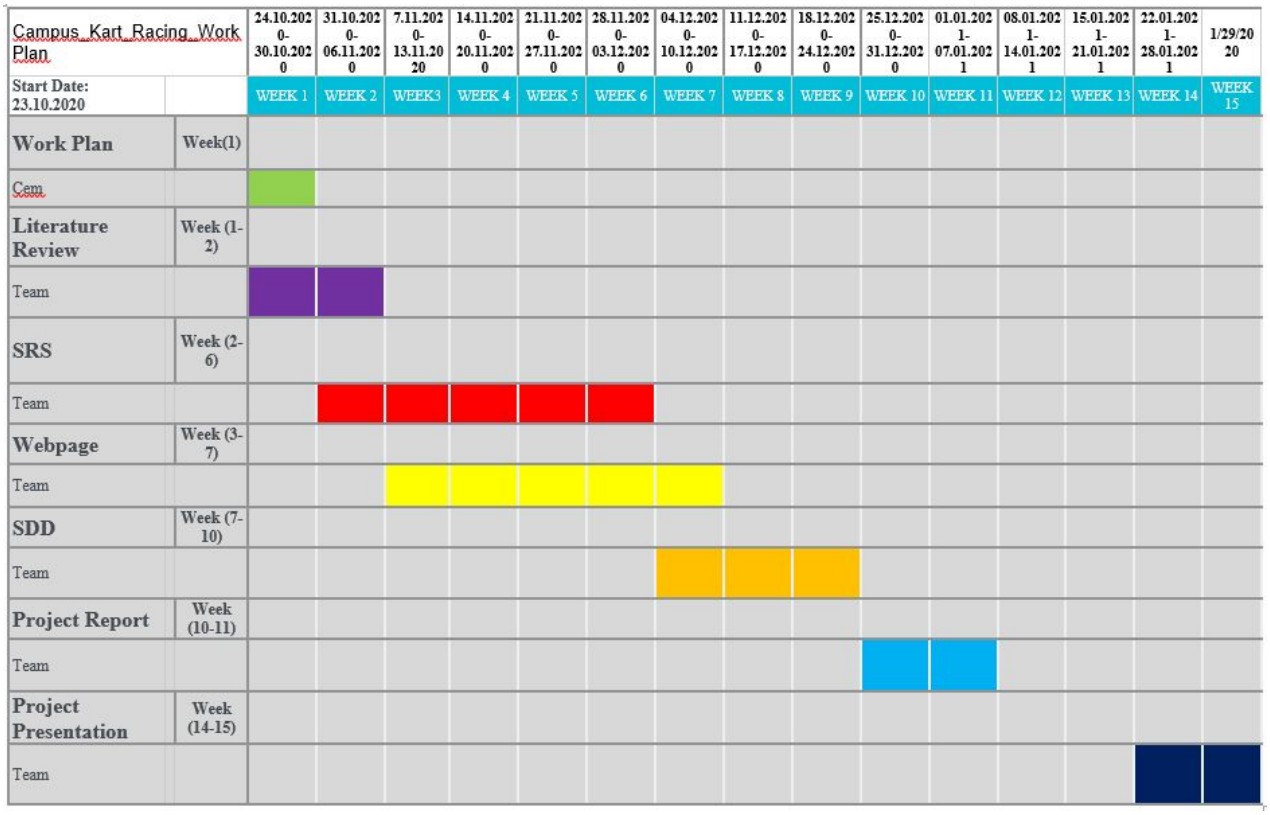

Figure 11: Project Work Plan.

2.5.3 Process Model

We chose to work with the agile method which is one of the project management methodologies for our CKRG game project. The necessary elements for our CKRG project (regular adaptation to changing conditions and even late changes in requirements are welcomed), the most convenient method is the agile method.

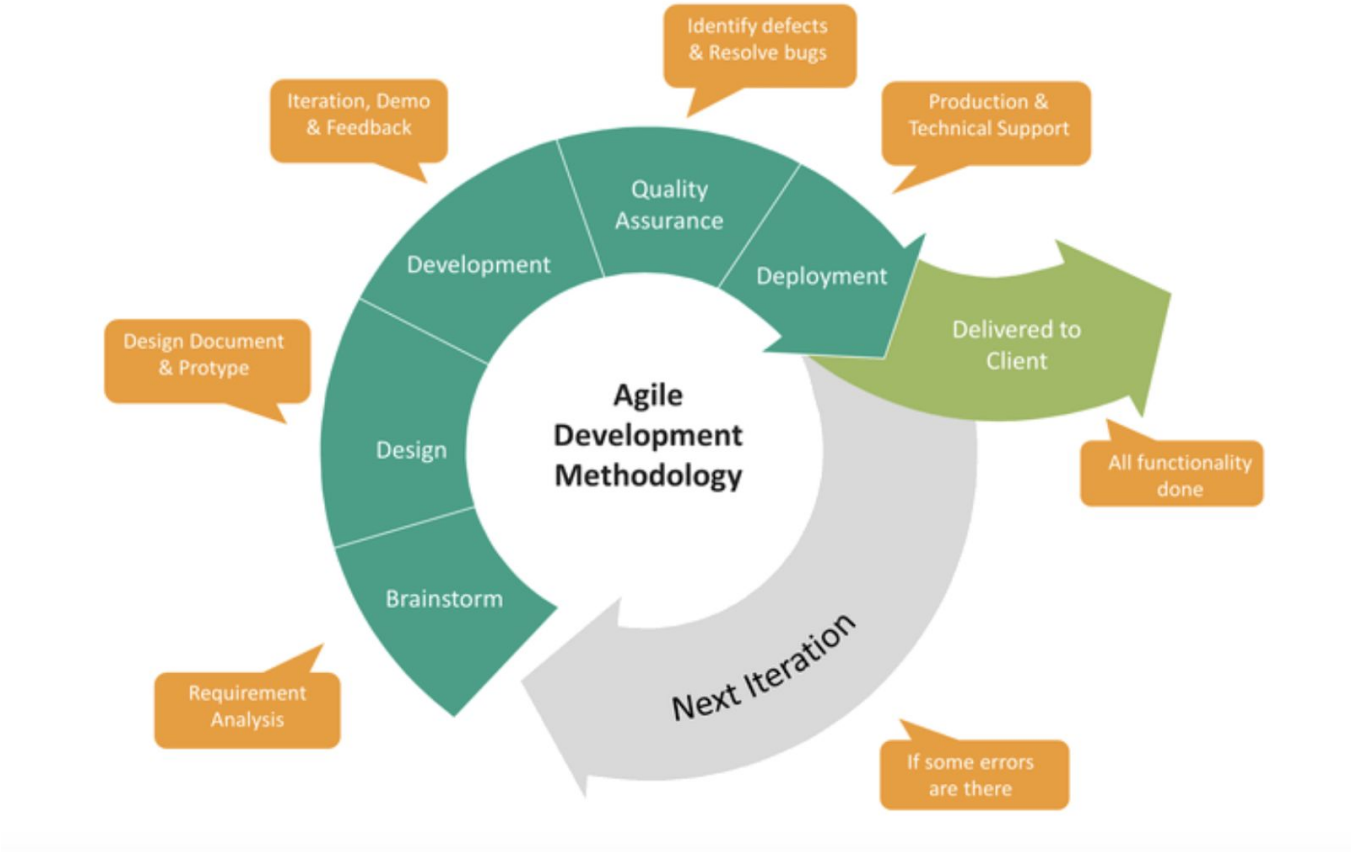

Figure 12: Agile Development Methodology.

Agile method is the general name for the iterative software development method which anticipates continuous development and testing activities throughout the software life cycle. These iterative steps are plan, design, develop, test and release.

2.6. CONCLUSION

In this software requirement specification document, all the required software requirement analysis, used systems and details of the project named CKRG are explained. Detailed functional and non-functional requirements, system, user, software and hardware interfaces, artificial intelligence are explained in detail. This SRS document will be a guide for the SDD document to be developed.

3. SOFTWARE DESIGN DESCRIPTION

3.1. INTRODUCTION

3.1.1 Purpose

The purpose of this Software Design Document is to provide detailed technical information about the project named Campus Kart Racing. In order to provide a better comprehension, this SDD includes various diagrams such as UML diagram of the project, activity diagram and block diagram.

3.1.2 Scope

This document contains details of CKRG.

Unity Game Engine is used to combine the game's physics, sound, artificial intelligence and game scenes. Unity is widely used because it offers many options such as low system requirements and physics calculations.[10] Unity uses the C# infrastructure as its coding language. It also offers cross-platform support. For these reasons, we chose Unity as the game engine.

We will be using 3Ds Max and Blender as 3D modeling tools. Blender is an open source 3D modeling tool that requires low system requirements and allows you to model in any category. 3Ds Max, on the other hand, is a powerful 3D modeling tool developed by Autodesk, one of the leading companies in the industry.

C # programming language will be used to create the script. C # is an object oriented programming language that offers a C-like experience developed by Microsoft.[11]

We will use the Photon Network infrastructure that is compatible with Unity, as the game offers in multiplayer mode other than single player mode. Photon Network is used by over 500,000 developers because it is easy to implement, self-documented and partially free.

3.1.3 Glossary

Expression	**Meaning**
CKRG	Campus Kart Racing Game
AI	Artificial Intelligence
CPG	Cross Platform Game

NPC	Non Playable Character (Controlled by AI)
GUI	Graphical User Interface
SDD	Software Design Document
UML	Unified Modelling Language (Used in Software Engineering)
PUN	Photon Unity Networking
MP	Multiplayer
OS	Operating System

3.1.4 Motivation

We are a group of seniors interested in gaming in the computer engineering department. As a group, our goal is to provide players with a playable, fun and competitive racing game experience. For this purpose, we chose Unity as the game engine we are familiar with as a group.We did research and purchased courses to make up for our shortcomings in racing game mechanics. We learn to use 3D modeling tools for designs. We are considering getting professional help in the points we are missing.

3.2 ARCHITECTURAL APPROACH

3.2.1 Simulation Design Approach

We used the Agile Management system in this project. In this form of project management, there is a continuous form of a gradual solution from the start date of the project instead of presenting the product after completion. In Agile system, projects are divided into smaller parts. It is then ranked according to importance. It is then delivered and checked in smaller weekly cycles called "Sprints." The goals are set by the group before the cycle begins. In some special cases the team must contact the customer directly and answer questions. Project workers and leaders can predict roughly how long the project will take. Feedback is received and recorded from the customer after each presentation and these notes can be used usefully in later stages. This system aims to continuously develop and improve the product.

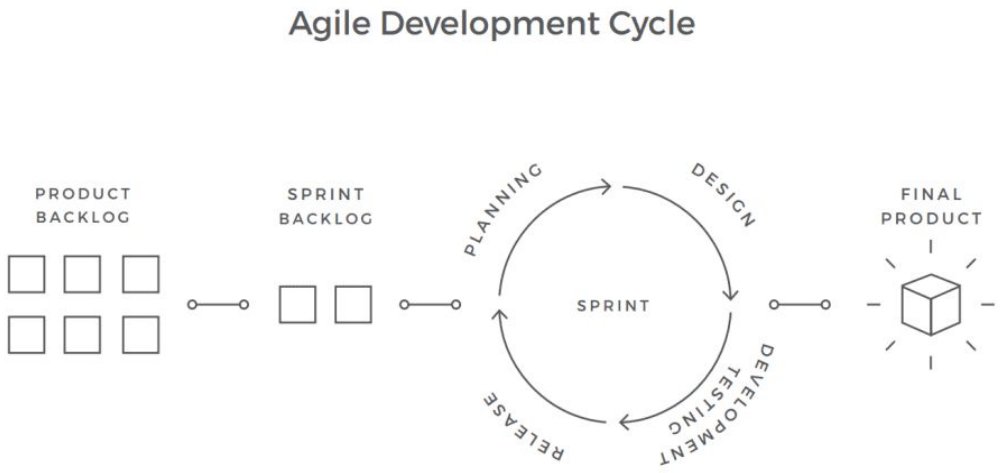

Figure1: Agile Development Cycle

3.2.1.1 Benefits of Using Agile Method

During the project, user participation is encouraged, making it easier to identify problems and make the experience more systematic for the team. There is feedback throughout the whole process, so progress is made.Thanks to this methodology, development is cared for from the very beginning of the project and major problems are avoided.

Teamwork: Team members work together. Everyone knows everyone's duty. The probability of problems is minimized.

Improvements: Various improvements are continuously made and tested in cycles to reveal the final product.

Client Side: A client is closely involved in development and can change requirements or accept the team's suggestions.

Distribution of Work: The project consists of small cycles (Sprints).

Flexibility: The scope of the project can be changed and transferred according to the client's request or variables. Flexibility can be provided according to feedback or other situations.

Gantt charts are used to plan projects of all sizes and are a useful way to show what work is planned on a given day.

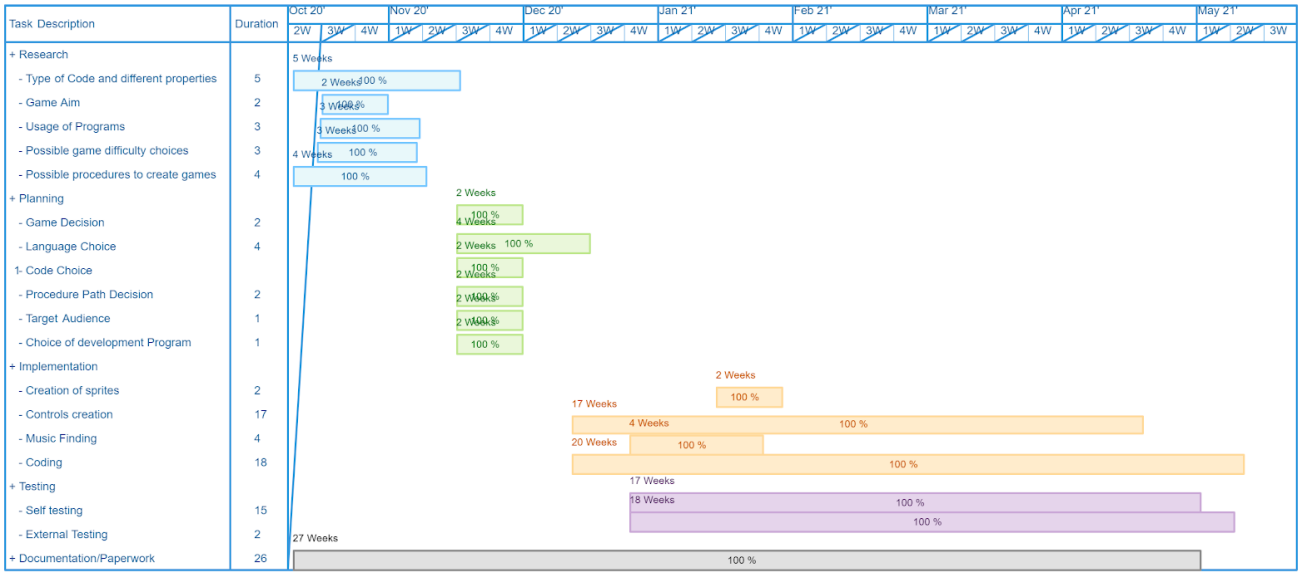

Figure 2: Gantt Chart for Our Project

3.2.1.2 Class Diagram

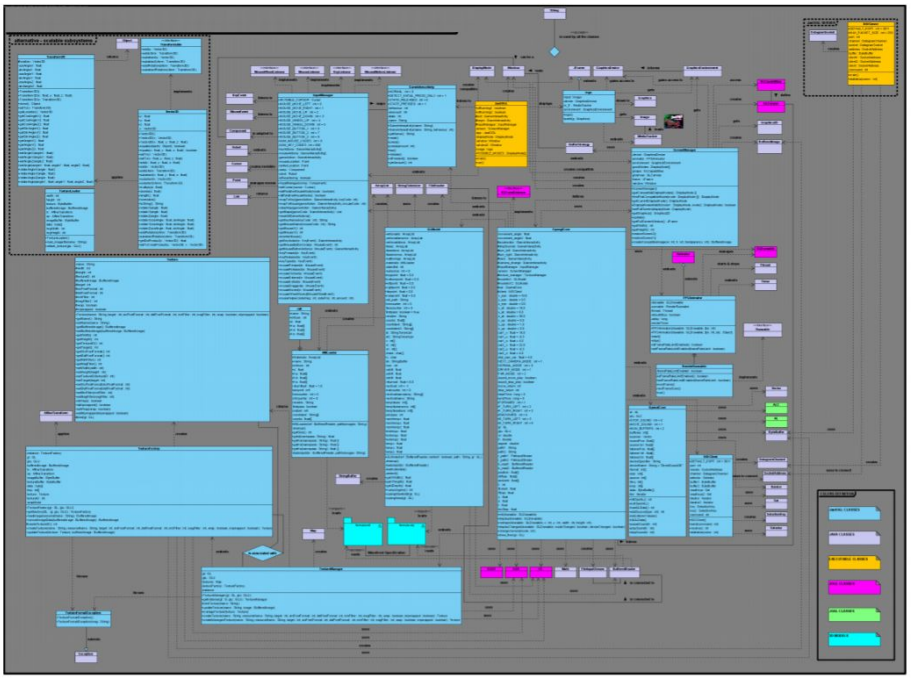

Figure 3:Class Diagram

In the figure above, the connections between classes are shown clearly. But this shape does not meet every need. This diagram, which is currently prepared only to show the in-game racing dynamics, will change in future sprint meetings and will continue to be developed. The CGame class retains the main functions, while the map and coordinates classes control the location of the tool and work with the pathfinding algorithm. Highscore class keeps the high scores and rankings as can be predicted.

3.2.2 Architectural Estimated Work Patterns

3.2.2.1 Physics and Movement

The physics of the car is calculated using three-dimensional vectors (Vector3 [12]) in the CarController script. The physical attributes of the car are:

Mass (float mass)

Direction of movement and speed (Vector3 velocity)

Position (Vector3 position)

Forward (Vector3 forward)

Upward (Vector3 upward)

A car also has some motion attributes. An AnimationCurve [13] is a curve that can be edited directly in the graphic. The following are the car's motion attributes:

Acceleration / backward acceleration (float acceleration / backAcceleration)

Inertia of acceleration (AnimationCurve accelerationDrag)

Acceleration of rotation (AnimationCurve torque)

Inertia of rotation (float rotateDrage)

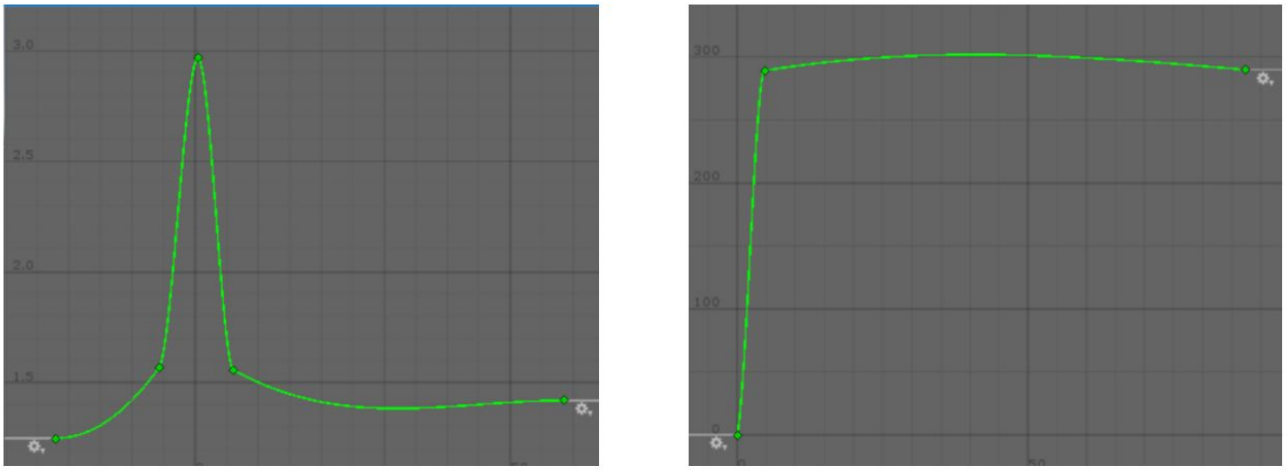

Figure 4: Acceleration Drag and torque

The calculation of the movement:

velocity + = InputY () * forward * acceleration * deltaTime (deltaTime is the time between each frame, InputX () / InputY () is the horizontal / vertical input of the player between -1 and

AddTorque (InputX () * upward * torque.Evaluate (velocity.magnitude) * deltaTime)

(AddTorque () is a function that adds acceleration to the rotation of the car. Evaluate () takes the value of the x-axis and returns the value of the y-axis of the curve. Magnitude is the length of a vector)

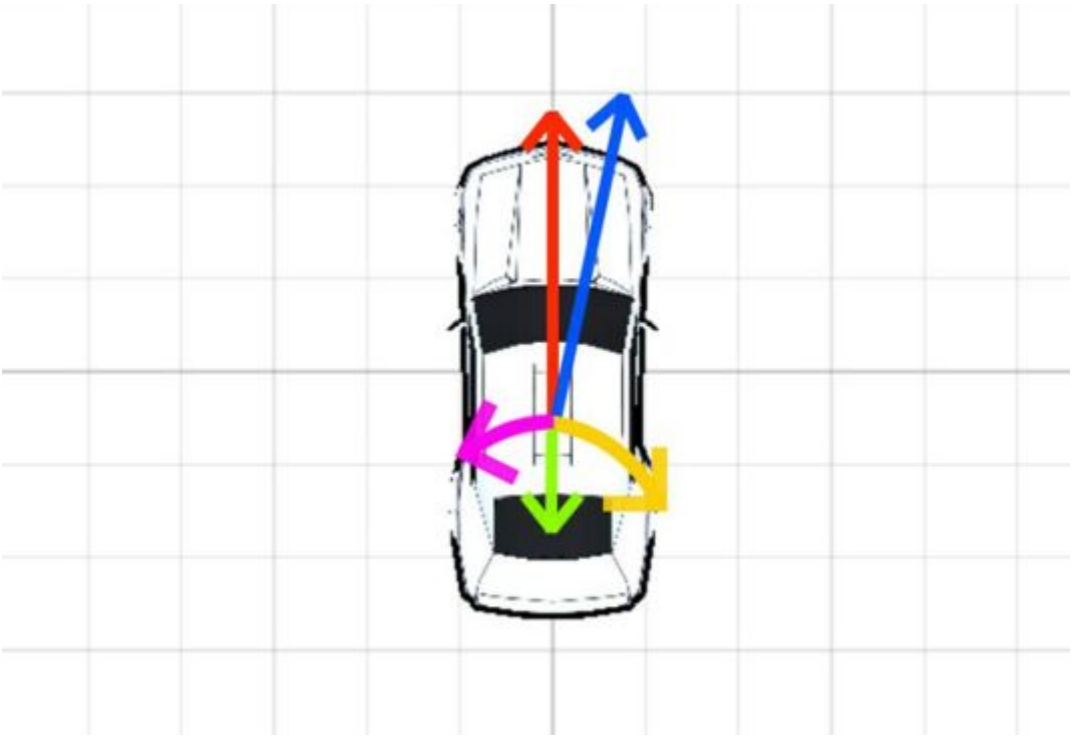

Figure 5: Car Vectors

Red: initially velocity, green: accelerationDrag, yellow: torque, purple: torqueDrag, blue: new velocity

After each frame the friction is calculated:

Velocity * = (1-accelerationDrag.Evaluate (velocity.z)) * deltaTime

angularVelocity * = (1-rotateDrag) * deltaTime

3.2.2.2 Network and Synchronization

PUN is a Unity package for multiplayer games. Offering flexible matching, custom properties, fast and reliable connection, PUN is used by many developers.

Figure 6: How PUN Works

Players connect to the server and start exchanging data. This data can be data such as the speed, location, color of the player.

The master server is a server set up for each game on the PUN system. Players can connect to this main server and can open, list and join rooms as desired. In our game, each room will represent a race.

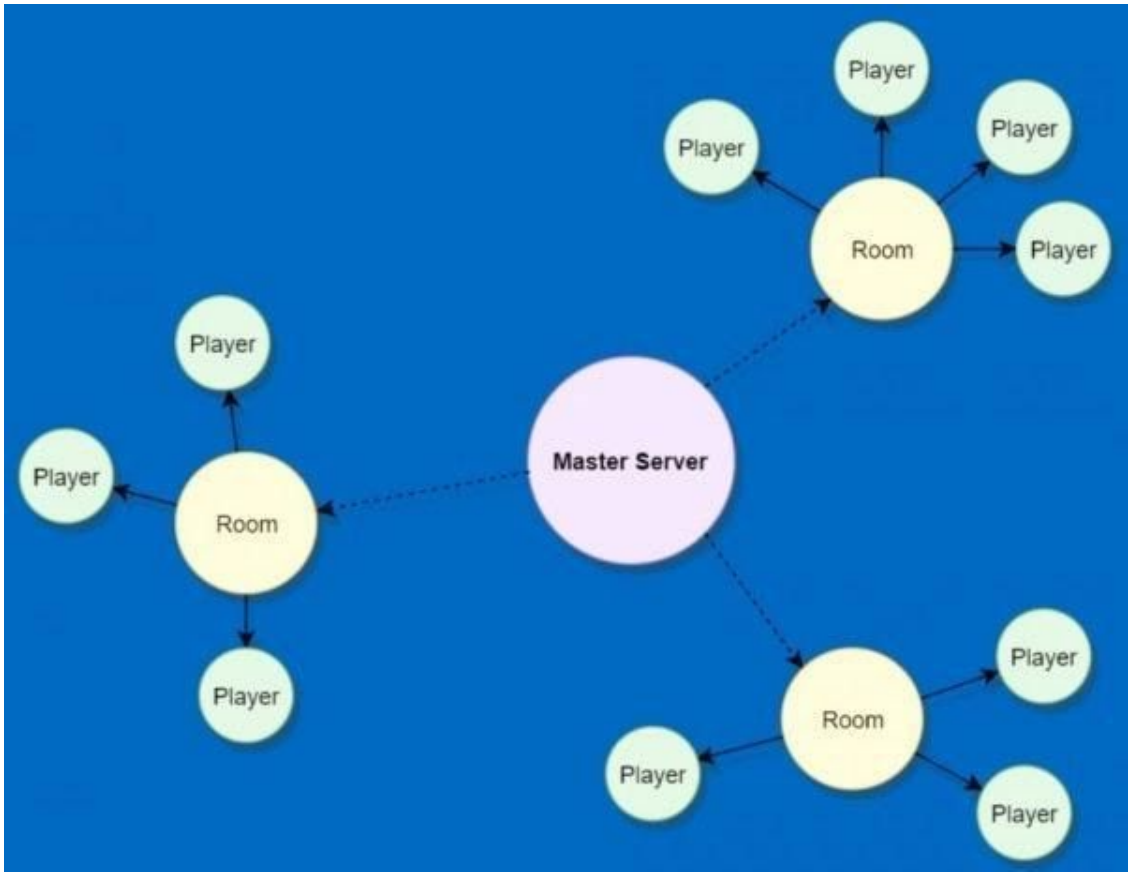

Figure 7: Master Server

3.3 USE CASE REALIZATIONS

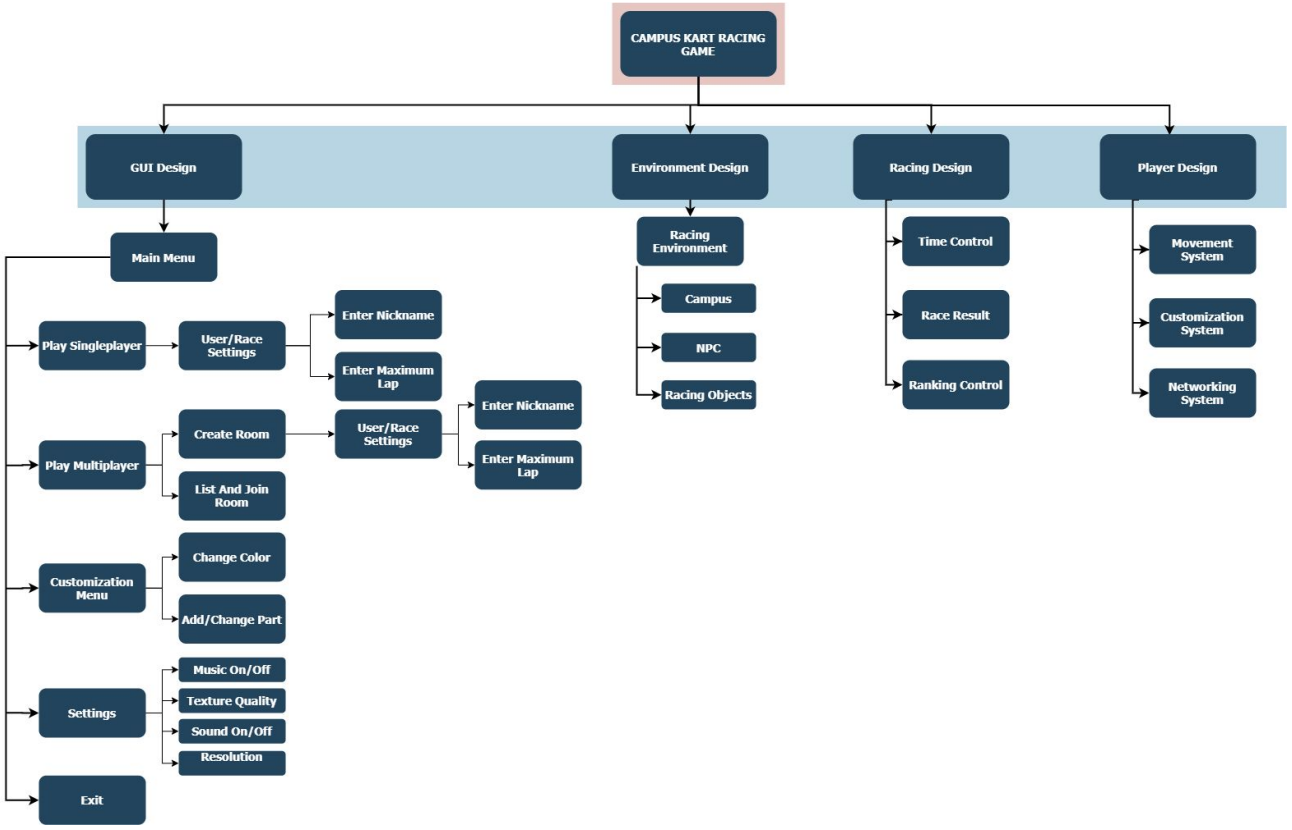

Figure 8: Project Components of Campus Kart Racing Game

3.3.1 Brief Description of Figure 6

The game consists of four main components. These are indicated in Figure 6.

3.3.1.1 GUI Design

GUI design is the main layer that provides the connection between the player and the game. The player uses the other components of the game with gui interaction.All players can access other systems from the main menu. The main menu is divided into five sub-menus. These are single player play, multiplayer play, customization, settings and exit buttons. The player can customize her vehicle in the customization system, and set her game to her own system on the settings screen.

3.3.1.2 Environment Design

Environment Design is the racing area where the race will take place. It includes the campus (Çankaya University) area, which is the main element of our game. It also includes many racing objects such as start / finish line, direction signs. Finally, if the player chooses to play in single player mode, it will also include NPC vehicles driven by AI.

3.3.1.3 Racing Design

Racing design is a system that includes the players' ranking, race times and race results.

3.3.1.4 Player Design

It is the section that contains the main mechanics of the player. Vehicle motion system, network system and customization systems are included in this field.

3.4 ENVIRONMENT

3.4.1 Modelling Environment

In this project, tools such as Blender and 3Ds Max were used to model the racing area. The polygon numbers of the models were tried to be kept as low as possible. Because the fewer polygons on the screen, the shorter it will take for the game engine to render the models. This means an increase in performance and FPS for the game. As mobile devices are not as powerful as computers today, it is very important to keep in-game performance as high as possible. We will use Blender and 3Ds Max modeling software to model a realistic environment such as cars, homes, barriers, map, level design, and other stuff.

In addition, another factor that makes a game look realistic is lighting. Skybox will be used for a realistic sky view in the game. Skybox method is to combine six different photos in skybox system to obtain a cuboid shape.

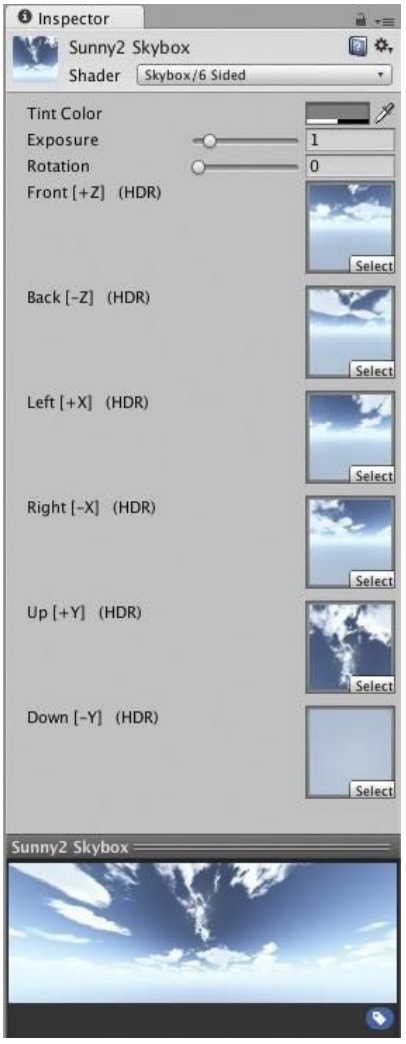

Figure 9: Example Skybox

Baked lighting system, which is better in terms of project performance, was used. Baked Lighting system, a light setting is made according to the positions of static (non-moving) objects in the scene and recorded according to these settings. In this way, the game engine memorizes the lighting and does not consume additional power for the lighting. This improves game performance.

The following objects are the design object samples from our game:

Figure10: Example Scene (Not in Unity Engine)

Figure 11: 3D Tree Objects

3.5 REFERENCES

[1]"2020 Video Game Industry Statistics, Trends & Data", *WePC*, 2020. [Online]. Available: https://www.wepc.com/news/video-game-statistics/. [Accessed: 06- Nov- 2020].

[2]"List of video games that support cross-platform play", *En.wikipedia.org*, 2020. [Online]. Available: https://en.wikipedia.org/wiki/List_of_video_games_that_support_cross-platform_play. [Accessed: 06- Nov- 2020].

[3]V. Yanev, "Video Game Demographics - 25 Powerful Stats for 2020", *TechJury*, 2020. [Online]. Available: https://techjury.net/blog/video-game-demographics/#gref. [Accessed: 06- Nov- 2020].

[4]S. D'Ambra, "What is Cross Platform Development?", *ClearTech Interactive*, 2020. [Online]. Available: https://www.cleart.com/what-is-cross-platform-development.html. [Accessed: 06- Nov- 2020].

[5]"Game engine", *En.wikipedia.org*, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Game_engine. [Accessed: 06- Nov- 2020].

[6]N. Bonfiglio, "DeepMind partners with gaming company for AI research", *The Daily Dot*, 2020. [Online]. Available: https://www.dailydot.com/debug/unity-deempind-ai/. [Accessed: 06- Nov- 2020].

[7]"Game Mechanics in Racing Game: Research", *UKDiss.com*, 2020. [Online]. Available: https://ukdiss.com/examples/research-on-game-mechanics-in-racing-games.php. [Accessed: 06- Nov- 2020].

[8]"Burnout (video game)", *En.wikipedia.org*, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Burnout_(video_game). [Accessed: 06- Nov- 2020].

[9] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

[10] U. Technologies, “Unity User Manual (2019.4 LTS),” Unity. [Online]. Available: https://docs.unity3d.com/Manual/. [Accessed: 25-Dec-2020].

[11] “C Sharp (programming language),” *Wikipedia*, 04-Dec-2020. [Online]. Available: https://en.wikipedia.org/wiki/C_Sharp_(programming_language). [Accessed: 25-Dec-2020].

[12] U. Technologies, “Vector3,” *Unity*. [Online]. Available: https://docs.unity3d.com/ScriptReference/Vector3.html. [Accessed: 25-Dec-2020].

[13] U. Technologies, “AnimationCurve,” *Unity*. [Online]. Available: https://docs.unity3d.com/ScriptReference/AnimationCurve.html. [Accessed: 25-Dec-2020].