Driving Lessons Simulator

Department of Software Engineering, Braude College of Engineering

Capstone Project Phase A

24-1-D-25

by Ilya Averuk
Itshak Efraimov

Advisor Dr. Moshe Sulamy

GitHub

Abstract

Drivers navigate complex environments fueled by theoretical knowledge of laws and signs. However, translating theory into actual driving skills often proves challenging, leading to anxieties and potential safety concerns. This project tackles this crucial gap, presenting a groundbreaking driving lesson system that revolutionizes how we learn the road.

The system leverages dynamic track construction by instructors, personalized scenarios, detailed scoring, and feedback mechanisms provide data-driven insights for skill refinement. Collaboratively, users can upload custom tracks to a shared repository, fostering a diverse learning environment. Finally, integration with game controllers enhances realism for an immersive learning experience. This innovative system empowers drivers of all levels to navigate the road with confidence and skill.

Table of Contents

1. Introduction

Navigating the complex world of traffic requires not only theoretical knowledge of traffic laws and signs but also the ability to translate that knowledge into safe and effective driving practices (Beanland, Goode, Salmon, & Lenné, 2013)[1]. However, the transition from textbook learning to real-world driving can be daunting and anxiety-provoking, often leaving a significant gap between theoretical understanding and practical skill. This gap can lead to hesitation, uncertainty, and potentially unsafe driving behaviors.

This capstone project addresses this challenge by proposing the development of an innovative driving lesson simulator built using the Unity Engine. This simulator aims to provide a realistic and engaging learning environment where users can apply their knowledge in simulated scenarios (Backlund et al., 2007)[2]. The immediate feedback loop fosters deeper understanding and builds faster reflexes, enhancing overall driving competency (Cheng, Lyu, Liu, & Wong, 2019)[3].

Furthermore, the system empowers instructors to dynamically construct personalized tracks based on individual user performance and insights. These custom scenarios address specific areas of weakness and expose users to diverse driving situations, potentially including those difficult or impossible to encounter in traditional driving lessons. By sharing these tracks in a cloud-based repository, the system fosters a collaborative learning environment where users can learn from each other and benefit from a wider range of learning experiences.

Ultimately, this project aims to revolutionize driving education by offering a personalized, engaging, and data-driven learning platform that bridges the gap between theoretical knowledge and practical skills. By equipping users with the confidence and competence to navigate real-world driving situations effectively, the proposed system has the potential to significantly improve road safety and contribute to a more skilled and prepared generation of drivers.

2. Related Work

Driving simulation has been an area of active research and development, with a wide range of applications spanning from driver training to the evaluation of road infrastructure. In this section, we review existing literature and projects related to driving simulation, and track customization features.

Driving simulators have evolved significantly over the years, offering increasingly realistic experiences for both training and entertainment purposes. These simulators range from simple desktop applications to immersive setups with motion platforms and virtual reality headsets. One notable example is the "City Car Driving" simulator developed by Forward Development, which provides realistic driving scenarios in urban environments. Another is the "Euro Truck Simulator" series by SCS Software, focusing on the simulation of truck driving across European roads.

Research in driving simulation has focused on various aspects such as improving realism, enhancing user engagement, and assessing driving performance. For instance, studies by Mourant and Rockwell (1972)[4] explored the effectiveness of simulation-based training for improving driving skills and reducing accidents among novice drivers.

Customizable tracks and user feedback mechanisms are essential features in driving simulators to cater to diverse user preferences and learning objectives. Projects like "rFactor" and "Assetto Corsa" have gained popularity among enthusiasts for their extensive modding support, allowing users to create and share custom tracks, cars, and other assets. These communities contribute to the longevity and richness of the simulation experience by constantly expanding the available content.

Research in user-centered design and human-computer interaction has emphasized the importance of soliciting user feedback to improve system usability and satisfaction. Techniques such as crowdsourcing and user ratings have been widely adopted to gather feedback and prioritize feature development in software applications (Bernstein et al., 2010)[5]. By incorporating similar mechanisms into driving simulators, developers can effectively gauge user preferences and iteratively refine the simulation experience.

In summary, driving simulation has witnessed significant advancements in recent years, driven by innovations in technology and research. By building upon existing work and leveraging state-of-the-art techniques, our project aims to contribute to the evolution of driving simulation and provide a comprehensive learning platform for novice drivers and driving instructors alike.

3. Background

This section is dedicated to the algorithms and methods that will be employed in each of these areas to achieve the project's objectives effectively.

3.1 Unity 3D Engine

Unity 3D is a powerful game development platform widely used for creating interactive experiences across various platforms, including desktop, mobile, and virtual reality. Its intuitive interface, extensive documentation, and robust feature set make it an ideal choice for developing driving simulators with rich visuals and immersive gameplay.

3.1.1 Game Physics: Unity provides built-in physics engines (such as PhysX) that simulate realistic interactions between objects in the virtual environment. We will leverage these physics engines to model vehicle dynamics, including acceleration, steering, and braking.

3.1.2 Graphics Rendering: Unity's rendering pipeline supports advanced rendering techniques such as physically based rendering (PBR) and post-processing effects, which enhance the visual fidelity of the simulator. We will utilize these features to create lifelike environments with realistic lighting, materials, and effects.

3.1.3 User Interface (UI): Unity's UI system allows for the creation of interactive and customizable user interfaces, including menus, HUD elements, and interactive buttons. We will design user-friendly interfaces for driving instructions, track customization, and feedback collection.

Figure 1 User Interface of a basic Driving Simulator

3.2 Instructor Customization

Instructor customization allows driving instructors to create custom tracks with varying levels of difficulty and complexity, tailored to the specific learning needs of their students. This feature enhances the flexibility and adaptability of the simulator, accommodating a wide range of skill levels and training objectives.

3.2.1 Track Editor: We will develop a track editor tool within the Unity environment, enabling instructors to design custom tracks by placing road segments, intersections, signage, and environmental elements (e.g., buildings, vegetation).

3.2.2 Difficulty Settings: The track editor will allow instructors to adjust parameters such as road layout, traffic density, weather conditions, and time of day to create tracks of varying difficulty levels.

3.2.3 User Feedback System: After completing a track, users will provide feedback on various aspects such as track layout, realism, difficulty, and overall experience. We will implement a feedback system to collect and analyze this data, allowing instructors to iterate on track designs based on user input.

3.2.4 Track Ranking: Based on user feedback and performance metrics (e.g., completion time, driving accuracy), tracks will be ranked and sorted within the simulator, making it easier for users to discover and select tracks that align with their skill level and preferences.

Incorporating these elements into the driving simulator project will create a comprehensive and customizable learning experience for users, while leveraging cutting-edge technologies to enhance realism and interactivity.

4. Expected Achievements

4.1 Outcomes

The overall outcome of this project is to create a cutting-edge driving simulator that provides users with a realistic and immersive learning experience while incorporating advanced features such as instructor customization tools, and seamless user feedback integration. By combining state-of-the-art technologies with intuitive design principles, the simulator aims to improve driver education and training outcomes, enhance road safety, and inspire innovation in the field of driving simulation. Ultimately, the project seeks to empower users to develop the knowledge, skills, and confidence necessary to become safe, responsible, and proficient drivers in real-world environments.

4.2 Unique Features

4.2.1 Realistic Driving Experience: With great graphics rendering and dynamic environmental effects, users will feel fully immersed in lifelike driving scenarios, including varied weather conditions and realistic road surfaces.

4.2.2 Instructor Customization Tools: Driving instructors can create custom tracks tailored to individual learners' needs, adjusting difficulty settings, traffic densities, and environmental conditions with an intuitive track editor interface.

4.2.3 User Feedback Integration: A user-friendly feedback system allows users to provide input on their experiences, guiding future updates and improvements to the simulator based on real user feedback.

4.3 Criteria for Success

4.3.1 Unity Engine Learning:

Proficiency Utilization: Demonstrated ability of the development team to effectively utilize Unity's features, such as physics engines and rendering techniques.

Efficient Development: Completion of the project milestones within predefined timelines, indicating efficient development facilitated by Unity's user-friendly interface and extensive documentation.

Community Engagement: Active participation in Unity developer forums, successful resolution of technical challenges, and incorporation of community insights into project development.

4.3.2 Firebase Integration:

Integrating Firebase into the platform will enhance its functionality by providing real-time data storage, authentication, and efficient retrieval. We’ll create a Firebase project, initialize it in our simulator, design the database structure, and handle data storage and retrieval.

5. Engineering Process

5.1 Research – Unity 3D Game Development

We chose Unity to develop our driving simulator project. It is developed by utilizing its diverse set of features and tools to bring our vision to life. Here's how we will implement key aspects of our project within the Unity environment: We script the core gameplay logic, including vehicle dynamics, user interactions, traffic behavior, and collision detection, using C#. This scripting language is integral to defining how the simulator responds to user inputs and simulates realistic driving experiences.

5.1.1 Constraints and Challenges – Unity 3D: Creating a realistic driving environment with complex graphics and physics simulation can strain system resources, particularly on lower-end hardware. Optimizing performance to maintain smooth frame rates and responsiveness while ensuring visual fidelity and realism is a significant challenge.

Secondly, integrating various components within the Unity environment, such as vehicle dynamics and instructor customization tools, requires seamless coordination and data exchange. Managing dependencies, ensuring compatibility, and maintaining modularity can be challenging, especially as project complexity increases.

5.2 Research – Customization Instructor Tools and Feedback Mechanism

Using Unity's UI system and scripting features, we will design and implement intuitive customization tools within the simulator. These tools empower instructors to build custom tracks with varying difficulties, environmental conditions, and challenges. Instructors can adjust parameters such as traffic density and weather conditions to create personalized training scenarios tailored to learners' needs. The elements allowed for the instructor to adjust are located on Cloud storage such as Firebase.

Through Unity's UI system and scripting capabilities, we will integrate user feedback mechanisms within the simulator. These interfaces enable learners and instructors to provide input regarding track design, driving experiences, and overall satisfaction. User feedback is collected, analyzed, and utilized to iteratively refine and enhance the simulator's features and functionality.

5.3 Research – Cloud for Distribution of Instructor-Created Tracks

Cloud technology represents a paradigm shift in contemporary computing, facilitating the dynamic provisioning and utilization of computing resources over the internet. This departure from traditional on-premises infrastructure empowers organizations to swiftly adapt to fluctuating demands, optimizing operational costs, and bolstering overall efficiency.

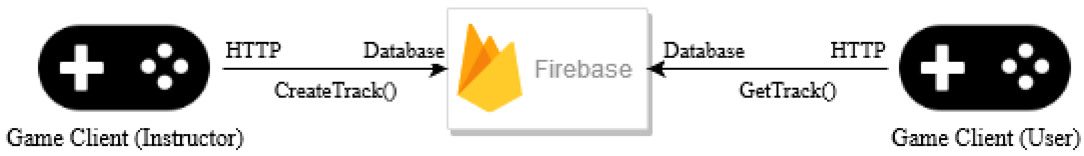


Figure 2 Overview of Firebase's role in the project

5.3.1 Google Firebase[6]: Firebase constitutes a contemporary exemplar in cloud technology, offering a comprehensive backend-as-a-service (BaaS) solution. Facilitating seamless application development, Firebase integrates features such as authentication, real-time database functionality, and hosting. Its user-friendly platform empowers developers by abstracting infrastructure complexities, allowing them to focus on building innovative and scalable applications. Firebase aligns with the principles of accessibility, cost-effectiveness, and simplified maintenance, emerging as a catalyst for streamlined app development and digital innovation across diverse domains.

5.4 Methodology and Development Process

In our project we will be using the Agile methodology. It allows the software to be released in iterations, iterative releases improve efficiency by allowing us to find and fix defects and align expectation early on. During each iteration, we will have meetings to discuss progress, bugs, and any changes that need to be made. To ensure an effective development process, we decided to divide the development to several sections:

1. Set up a basic Unity project with the necessary assets and scenes.
2. Implement basic car controls and physics for driving simulation.
3. Develop a track editor tool within the Unity environment for instructors to create custom tracks.
4. Implement functionalities such as adding road elements, adjusting difficulty, and saving tracks.
5. Set up Firebase project and integrate Firebase SDK into Unity.
6. Implement authentication, database storage, and retrieval functionalities.
7. Develop user feedback forms and implement a system for collecting and storing feedback.
8. Implement sorting algorithms to rank tracks based on user feedback.

Along the process, after every iteration we will evaluate the results we got, and make any necessary changes needed to be done before starting the next iteration. We will be focused on delivering operating software which takes into consideration feedback from users that will be testing and using our application.

6. Product

6.1 Functional Requirements

1	Accurately simulate the controls of a car, including acceleration, braking, steering, and gear shifting.
2	Environment should be real-world driving conditions, including varying weather conditions, time of day, and road surface conditions.
3	Ability to create custom driving tracks with different configurations, including road layouts, traffic density, and obstacles placement.
4	Ability to provide feedback on their driving experience, including track difficulty, realism, and suggestions for improvement.
5	Performance evaluations for users, including metrics such as adherence to traffic rules, reaction time to road signs, and overall driving proficiency.

6	User progress, track configurations, and feedback data should be securely stored and synchronized using Firebase, allowing access across different devices and sessions.
7	Track users' progress over time, including completed tracks and driving milestones achieved.

6.2 Non - Functional Requirements

1	Must run smoothly and responsively, even on lower-end hardware configurations, to provide an immersive and enjoyable user experience.
2	Ability to handle a growing number of users and tracks without sacrificing performance or stability.
3	User data, including personal information and driving performance metrics, must be securely stored and protected from unauthorized access or data breaches.
4	Interface should be intuitive and user-friendly, catering to users of varying skill levels and technical backgrounds.
5	Accessibility to users with disabilities, incorporating features such as customizable control settings.
6	The simulator should be stable and reliable, minimizing crashes, glitches, or unexpected errors during gameplay.
7	The simulator should be compatible with multiple platforms and devices, including desktop computers and laptops to maximize accessibility and reach.

6.3 Architecture Overview

The architecture of the project is comprised of two primary components:

Google Firebase: This component is utilized for the storage of user credentials, instructor-generated tracks, and feedback.

Game Client: This is a desktop application that is structured into four distinct layers:

- **Simulator Layer**: This layer is responsible for the physics of the vehicle and environment, the rendering of 3D models, and the production of sound.
- **Input Layer**: This layer handles the interpolation of various input devices such as keyboards, joysticks, steering wheels, and pedals.

- **Networking Layer**: This layer manages communication with Google Firebase for tasks such as user authentication, creating/accessing the tracks repository, and submitting feedback.
- **Analytics Layer**: This layer is tasked with the analysis and evaluation of user actions within the simulator. The purpose of this analysis is to provide detailed feedback and suggest future courses of action.

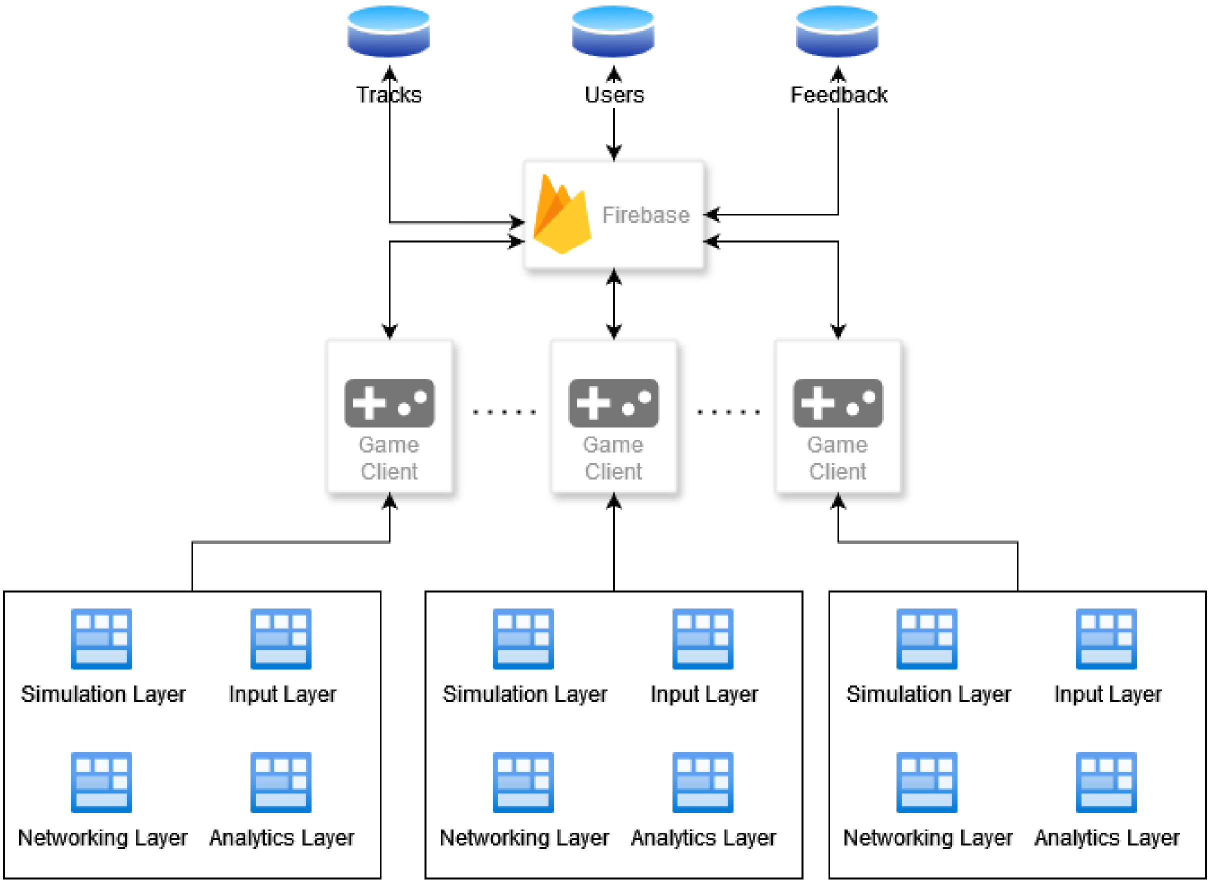


Figure 3 Diagram of the project's architecture

6.4 Scenes and Flow

Figure 4 Starting screen of the simulator

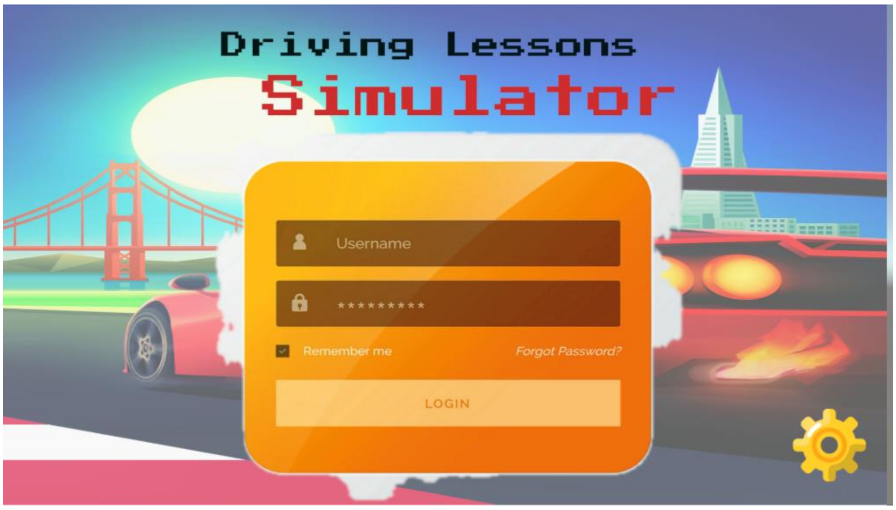


Figure 5 Login Page

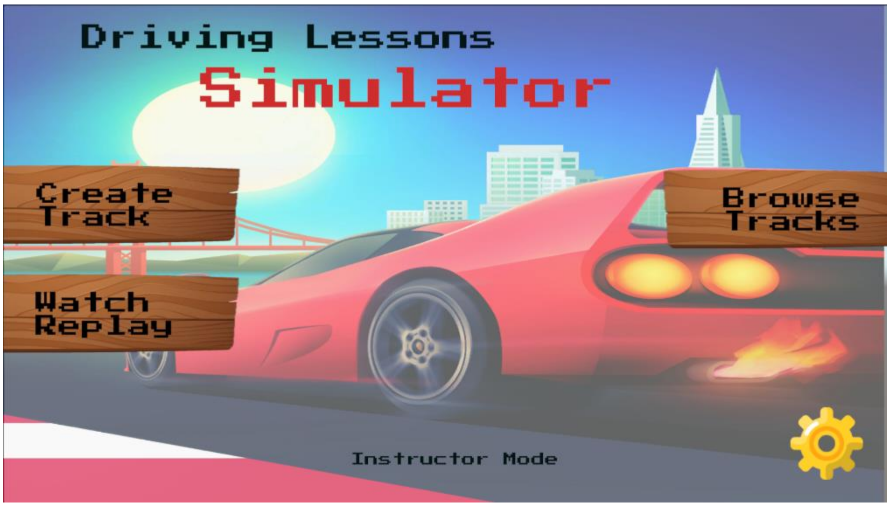


Figure 6 Home Page for Instructor User

Figure 7 Track Editor of the Instructor

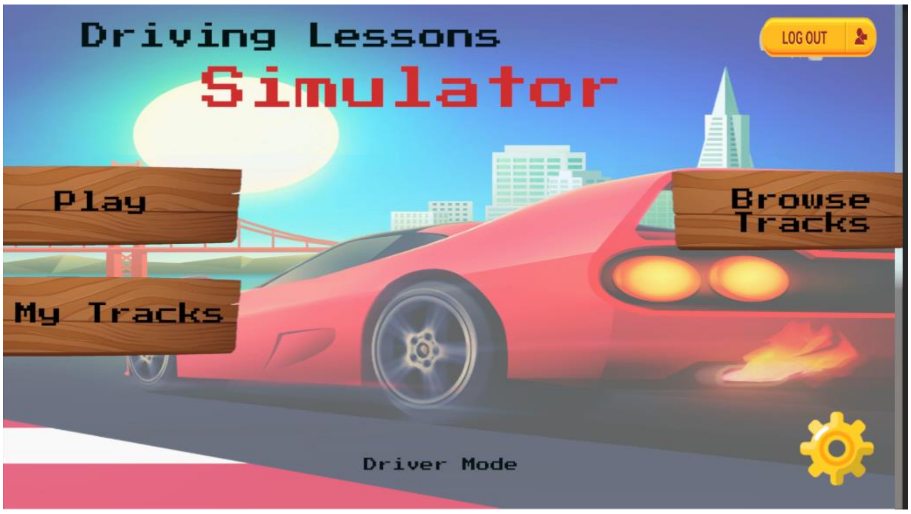
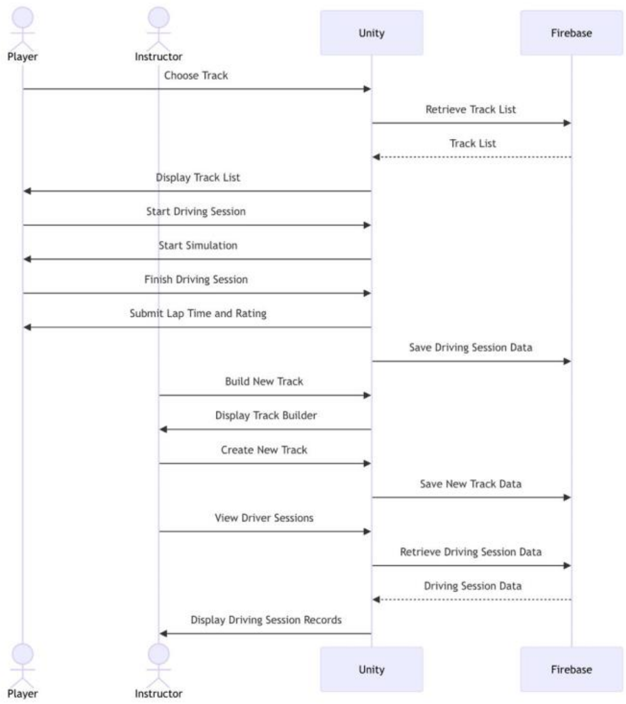


Figure 8 Home Page for Driver User

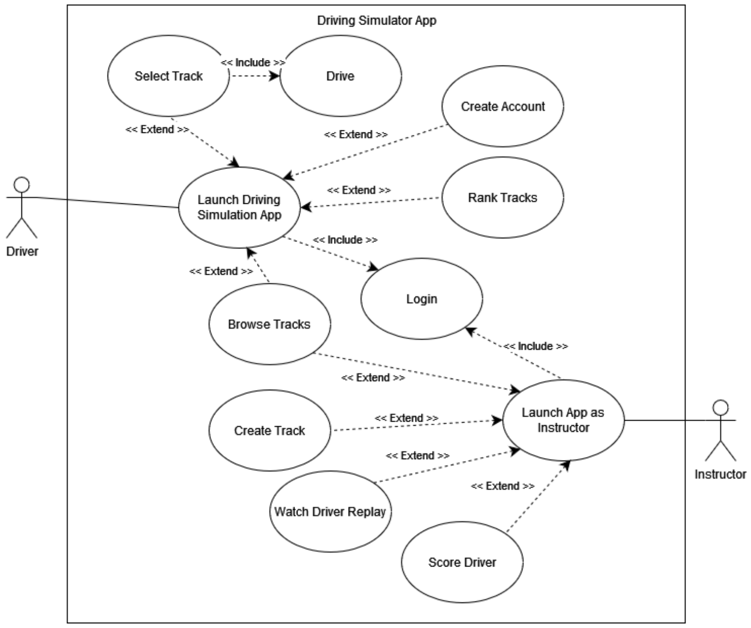
Figure 9 Driver Point of View

6.5 Diagrams

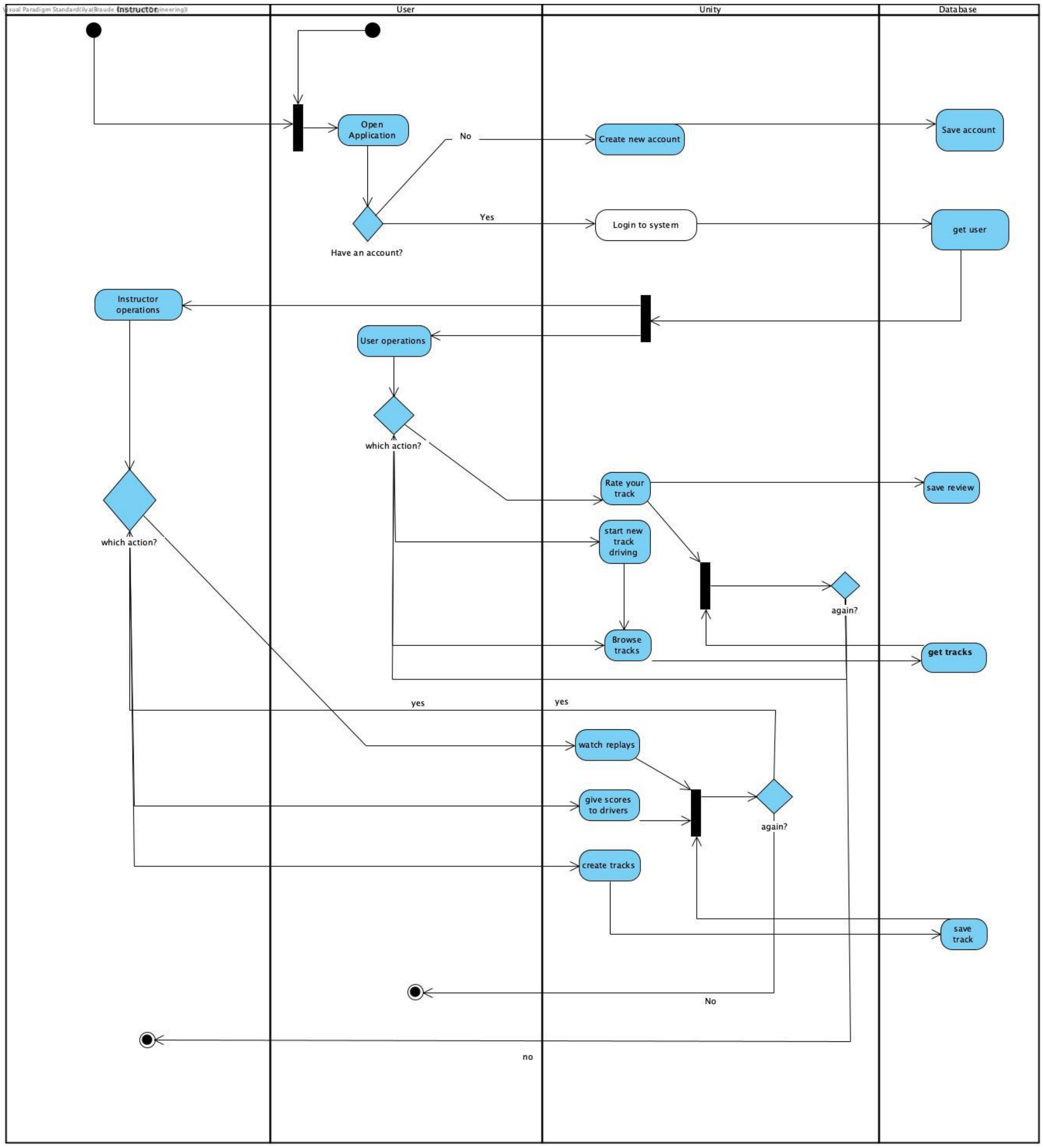
6.5.1 Sequence Diagram



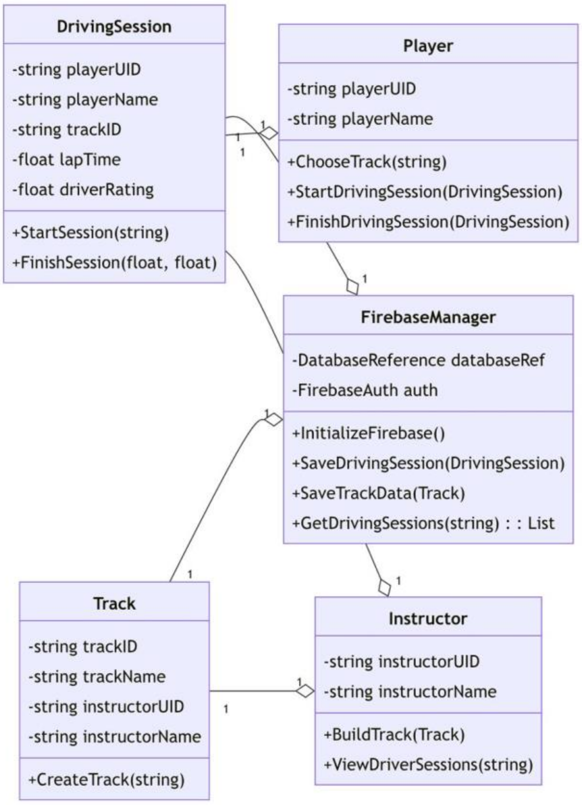
6.5.2 Use Case Diagram



6.5.3 Activity Diagram



6.5.4 Class Diagram



7. Evaluation and Verification

7.1 Evaluation

During the evaluation phase, the project undergoes meticulous scrutiny to ensure that each functional requirement is met. From the accurate replication of car controls to the faithful emulation of real-world driving conditions, every aspect is subjected to rigorous testing. Custom track creation tools are evaluated for their flexibility and ease of use, while user feedback on driving experience and performance metrics helps refine the simulator's features. The integration of Firebase for data management and synchronization is assessed to ensure seamless access and reliability across different devices and sessions. Concurrently, non-functional requirements such as performance on varying hardware, scalability, data security, intuitive user interface design, accessibility features, stability, and cross-platform compatibility are scrutinized to guarantee a smooth and inclusive user experience. This thorough evaluation process aims to deliver a robust and immersive driving simulator that meets the project's objectives while exceeding user expectations.

7.2 Verification

Unit Testing: Each individual component will be tested separately using a unit testing framework such as NUnit for Unity. For example, the car controls simulation will be tested to ensure that each control (acceleration, braking, steering, gear shifting) works as expected when activated.

Test Case ID	Description	Expected Result
TC1	Simulate acceleration in the car controls	The car's speed increases
TC2	Simulate braking in the car controls	The car's speed decreases
TC3	Simulate steering in the car controls	The car changes direction
TC4	Simulate gear shifting in the car controls	The car's gear changes
TC5	Change the environment to 'rainy'	The road surface becomes slippery
TC6	Change the environment to 'sunny'	The road surface is normal
TC7	Create a track with a specific configuration	The track is created and matches the specified configuration
TC8	Save a custom track	The track is saved successfully
TC9	Load a saved custom track	The loaded track matches the saved track
TC10	Attempt to access user data without proper authentication	The system prevents unauthorized access

Integration Testing: After unit testing, the components will be combined and tested together. For instance, the interaction between the car controls and the environment (like how the car responds to different road surface conditions) will be tested.

System Testing: The entire system will be tested as a whole. This includes testing the system's performance under maximum load, its ability to handle errors gracefully, and its behavior during and after system failures.

User Acceptance Testing: The final system will be tested by end-users. This involves creating scenarios that the end-users might encounter, and checking if the system behaves as expected.

References

[1] Beanland, V., Goode, N., Salmon, P. M., & Lenné, M. G. (2013). Is there a case for driver training? A review of the efficacy of pre-and post-license driver training. Safety Science, 51(1), 127-137

[2] Backlund, P., Engström, H., Gustavsson, M., Johannesson, M., Lebram, M., & Sjörs, E. (2007). Designing for self-efficacy in a game-based simulator: An experiment in leadership training. In Proceedings of the 2007 ACM SIGGRAPH symposium on Video games (pp. 33-38)

[3] Cheng, S. Y., Lyu, M. R., Liu, H. C., & Wong, J. M. (2019). Development and evaluation of a driving training simulator integrated with a multimedia teaching system. International Journal of Industrial Ergonomics, 70, 79-87

[4] Mourant, R. R., & Rockwell, T. H. (1972). Strategies of training and research in driver behavior. Human Factors, 14(4), 325-335.

[5] Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., & Crowell, D. (2010). Soylent: a word processor with a crowd inside. In Proceedings of the 23rd annual ACM symposium on User interface software and technology (pp. 313-322).

[6] ChatGPT. "Summarize Google Firebase"