BRAUDE
College of Engineering, Karmiel

Driving Lessons Simulator

Department of Software Engineering, Braude College of Engineering

Capstone Project Phase B

24-1-D-25

by Ilya Averuk
Itshak Efraimov

Advisor PhD. Moshe Sulami

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 personalizes 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.

The Driving Lessons Simulator is designed to serve a variety of user groups beyond typical learners, providing a safe and immersive alternative to real-world driving. Underage users who are eager to experience driving before becoming eligible for a license can practice in a controlled, risk-free environment. Disabled individuals who may face challenges with traditional driving can use the simulator to build driving skills or explore alternative controls tailored to their needs, helping them gain confidence without the pressures of real-world

driving. Additionally, elderly individuals or those with anxiety about driving can use the simulator as a tool for maintaining or improving their driving skills in a low-stress environment.

The simulator is also ideal for people preparing for driving tests or those seeking to refine specific skills, offering custom tracks designed by instructors. This includes unique driving scenarios that may be difficult or dangerous to practice in real life. The system's track creation tool, web-based dashboard, and Firebase integration ensure a personalized and data-driven experience for all users, making it a versatile solution for diverse driving education needs.

2. Development Process

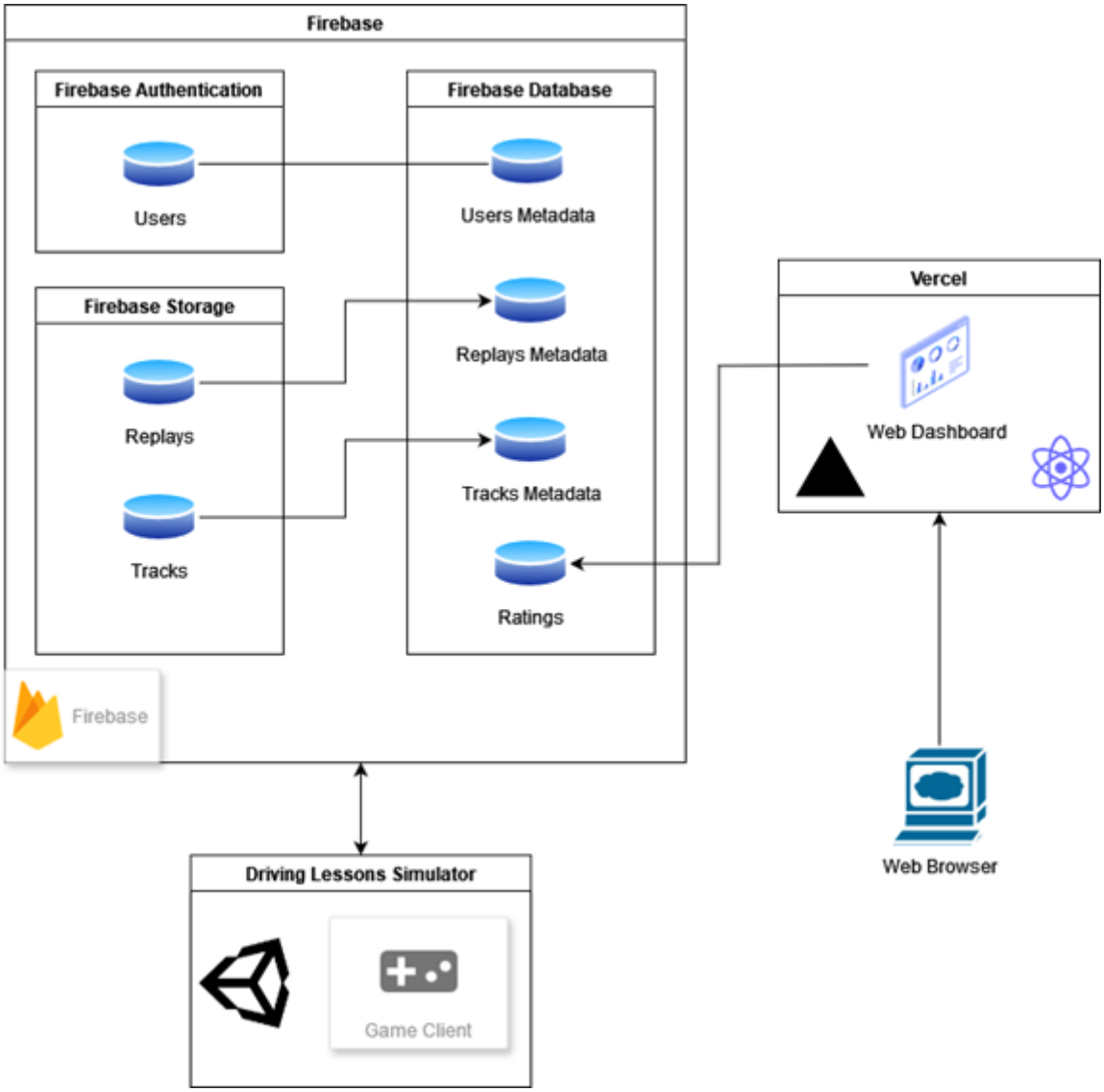
The development process for the driving lessons simulator in Unity, integrated with Firebase, began with careful planning and gathering of requirements. The core functionality was defined, including user authentication, session replays, levels, and a ranking system. Firebase was chosen for its capabilities in managing real-time data, storage and user authentication. The project setup involved creating a new Unity project, installing necessary packages, and integrating the Firebase SDK. Initial scenes for user login, main menu, level selection, and the driving simulation were developed with basic UI and placeholder assets. Firebase Authentication was integrated to manage user sessions, and a Firebase database was configured to store user credentials, session data, levels, and rankings, with security rules ensuring data privacy.

The core gameplay mechanics, including vehicle physics, controls, and environment design, were developed, with a system in place to record driving sessions for later replay. Levels were created with increasing difficulty, and user progress was tied to the Firebase database. A scoring system was implemented to evaluate driving performance, with scores stored in Firebase and a ranking system developed to display top tracks/experiences. The user interface was refined for smooth navigation, and Firebase was connected to display dynamic data like user rankings, session history and replay.

Rigorous playtesting ensured that all systems, including driving mechanics and Firebase integration, functioned correctly, and the game was optimized for performance. The process included regular updates and maintenance, incorporating new levels, gameplay improvements, and ongoing Firebase security reviews. This approach ensured a comprehensive and user-friendly driving lessons simulator with secure, real-time backend functionality provided by Firebase.

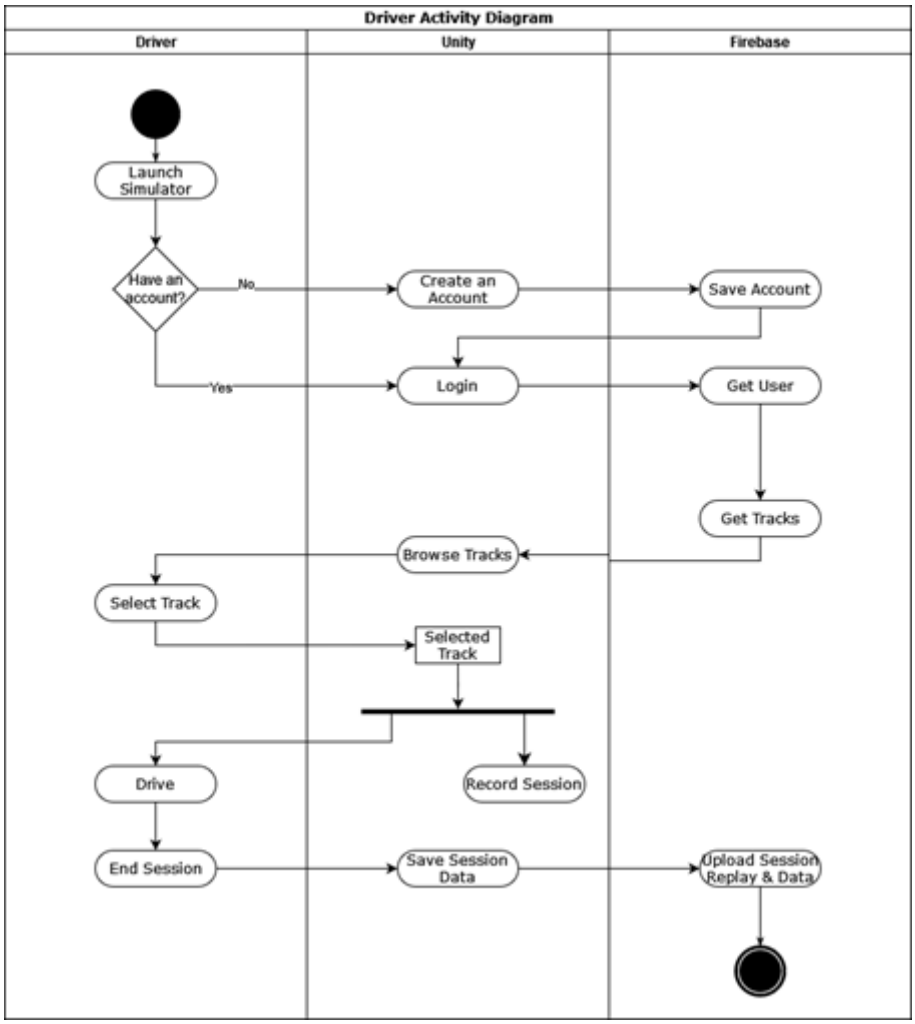
2.1 Diagrams

2.1.1 Architecture Diagram



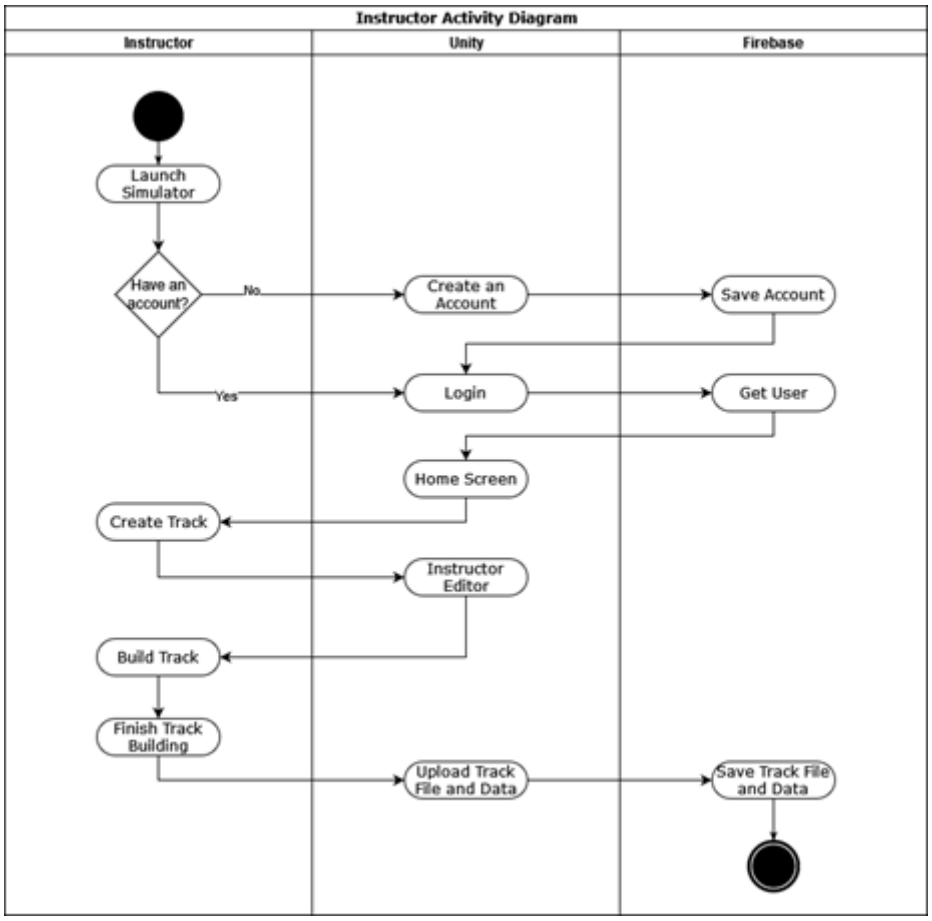
The diagram represents the architecture of the simulator, where data is hosted on Firebase services. Each Unity game client interacts with and stores data using the Firebase SDK, which is an interface provided by Google Firebase for communication with its services. Data is stored in a NoSQL database, and additional files (recordings and tracks) are stored in Firebase Storage. Firebase Authentication is used to provide an intuitive API for logging in and registering to access the simulator. The client can also access ratings data through a dashboard developed with React, utilizing Firebase's SDK, and deployed on Vercel.

2.1.2 Driver Activity Diagram



The driver activity diagram represents the main sequence of actions for a driver – driving. It begins with the user launching the simulator, creating a driver account if none exists, and then signing in. Next, the driver selects a track to drive on. Using key binds as defined in Chapter 3.4, the driver navigates around the track until clicking 'Finish session'. At that point, the simulator stops recording the session and transmits both the session recording and its metadata to Firebase, where they are stored in Firebase Storage and the Firebase Database, respectively, using the Firebase SDK.

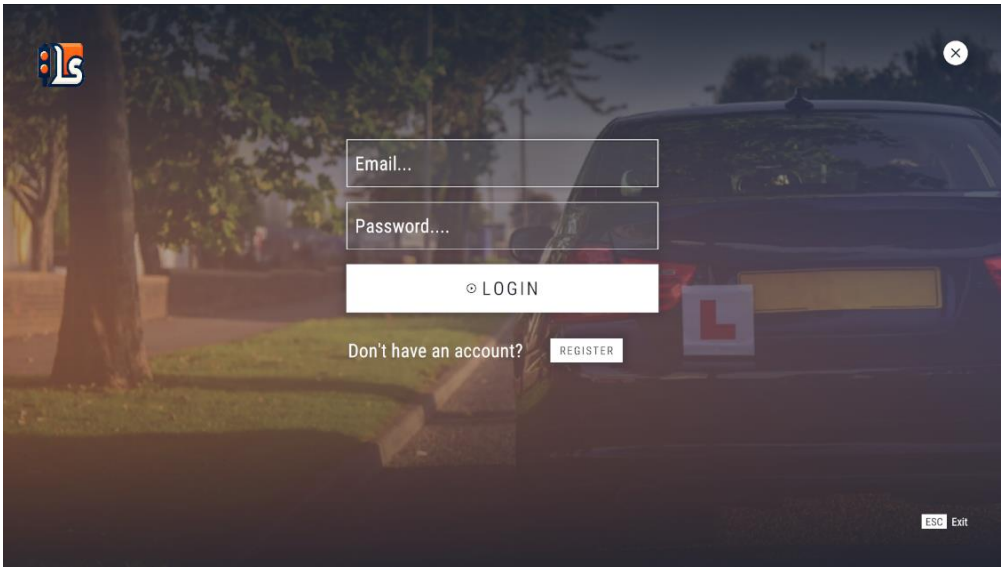
2.1.3 Instructor Activity Diagram



The instructor's activity diagram illustrates the flow of the track creation process. Like the driver's process, it begins with the user launching the simulator, creating an instructor account, and then logging in. From the home screen, selecting 'Create Track' launches the track editor. After completing the track creation using the mouse and keyboard, the instructor types the track's name and clicks 'Save'. Unity then takes the track file and its name, transmitting them to Firebase via the Firebase SDK, where they are stored in Firebase Storage and the Firebase Database.

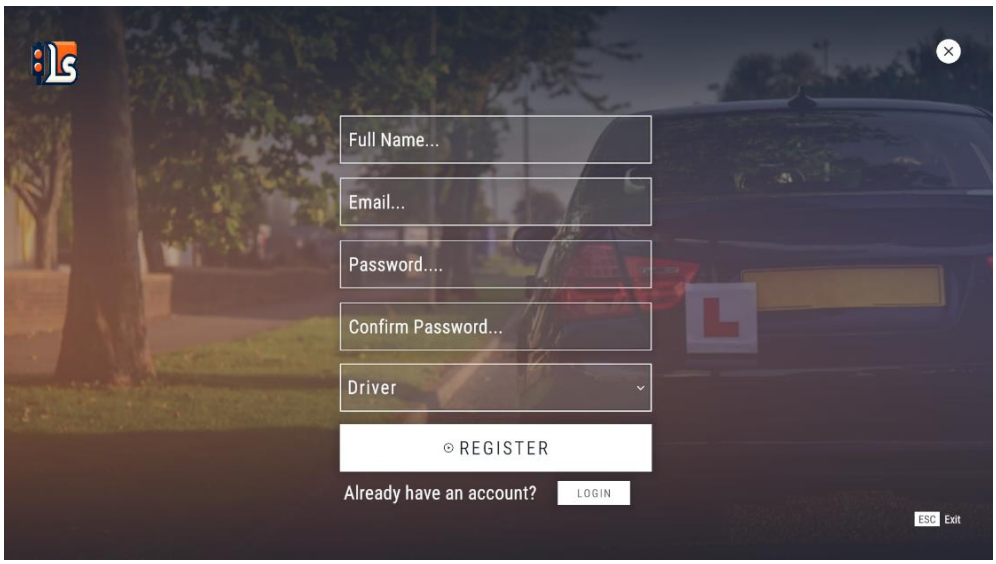
2.2 Scenes and Flow

2.2.1 Login Screen



This is the login screen of the Driving Lessons Simulator where you need to enter your credentials (email and password) if you have an account. If not, you can register by clicking the 'Register' button. After clicking on the 'Login' button, you will be directed to the simulator's home screen according to the role of your account – driver or instructor.

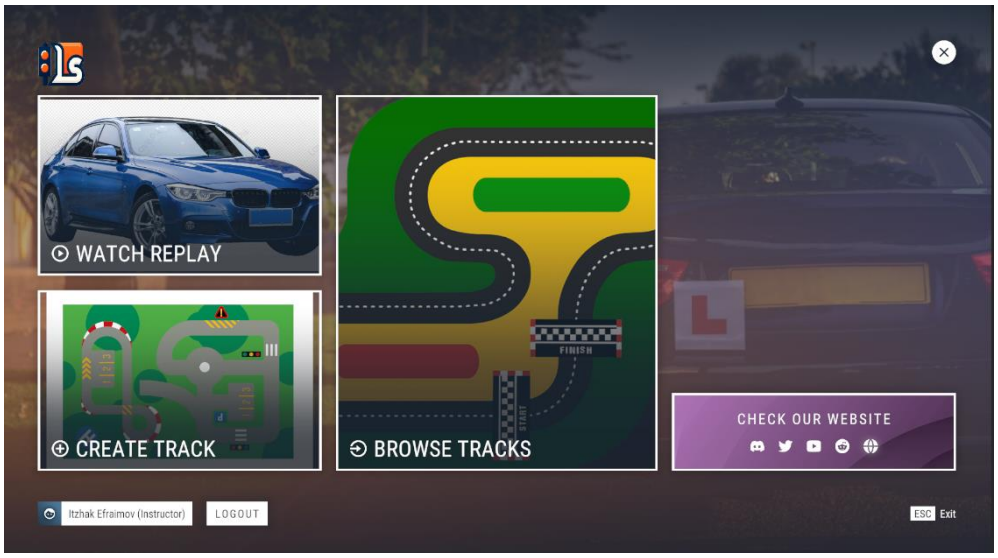
2.2.2 Registration Screen



This is the registration screen of the Driving Lessons Simulator where you can create an account. You should enter your full name, an email, password and which type of an account you would like to create – driver or instructor. It requires you to type the password and then

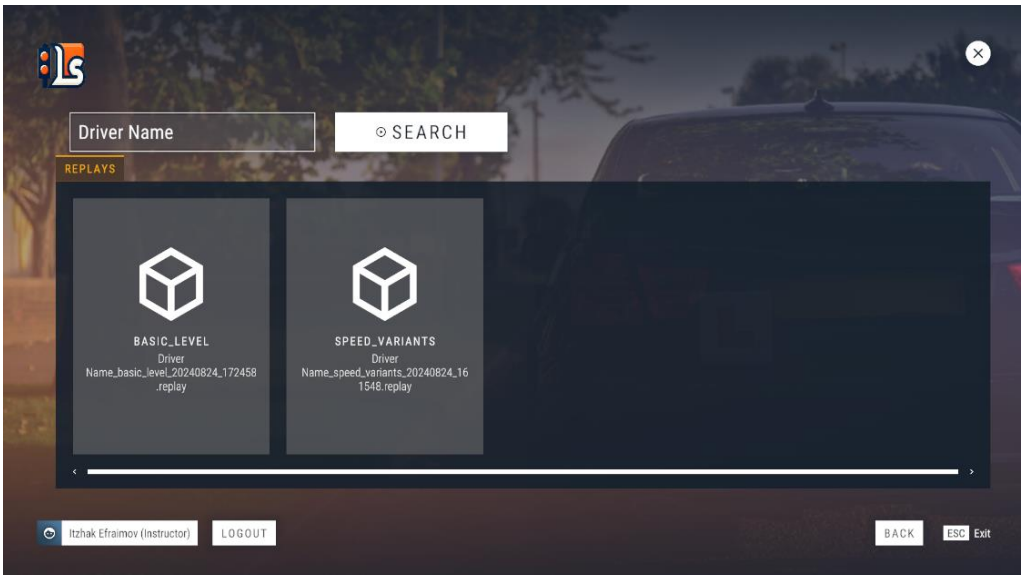
confirm it. After that, click 'Register.' You will always receive a response to show you if the action was successful or if there were any errors.

2.2.3 Instructor Home Screen



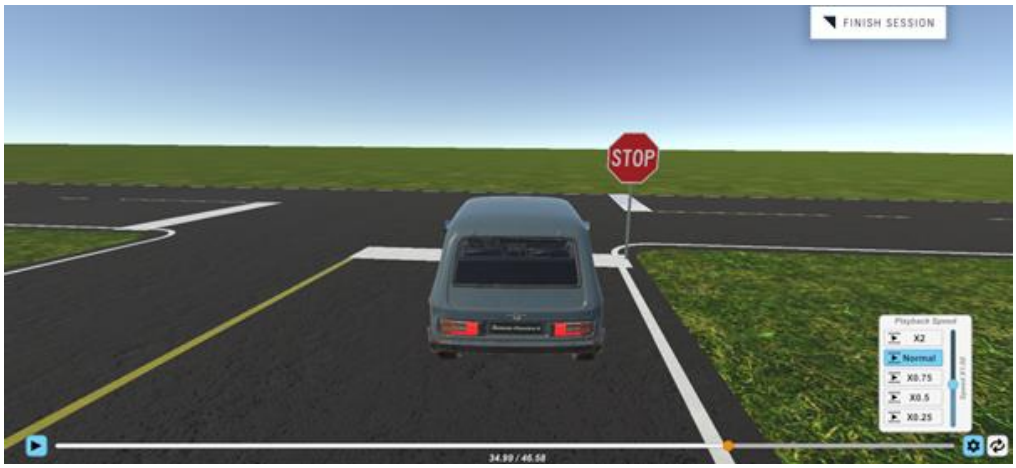
The instructor's home screen is where the instructor can select what type of action to perform in the simulator. The instructor can either watch a driver's replay of a session, create a track from scratch or update/modify a track he previously created through the browse tracks option.

2.2.4 Watch Replay Flow

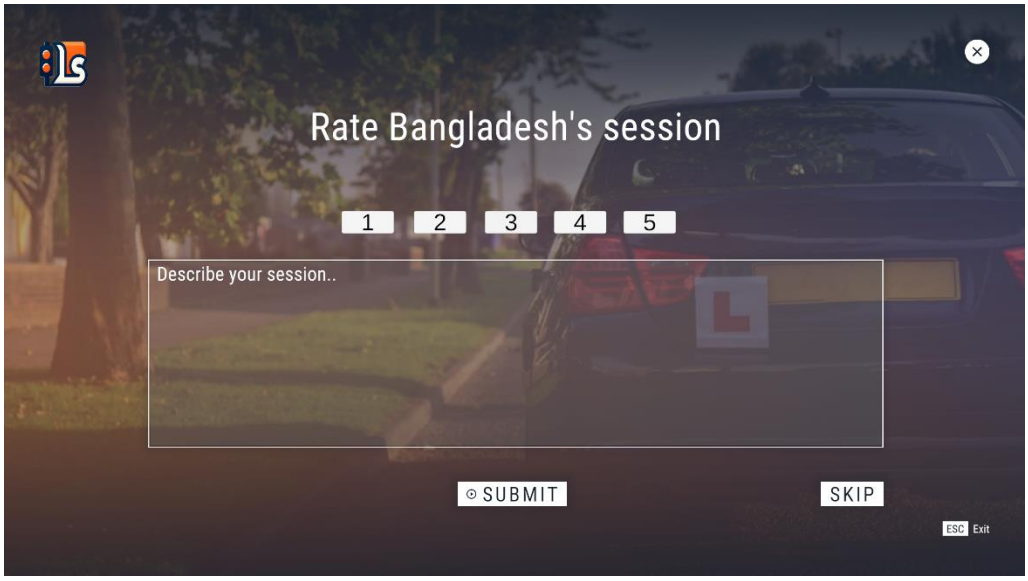


To watch a driver's replay – the instructor needs to search for all available replays under the driver's full name. The search results will be recorded sessions of the drivers, showing the track name of the session and when it was recorded. Once the instructor selects a session to

replay by clicking it, the simulator will automatically load the track and the replay of the session.

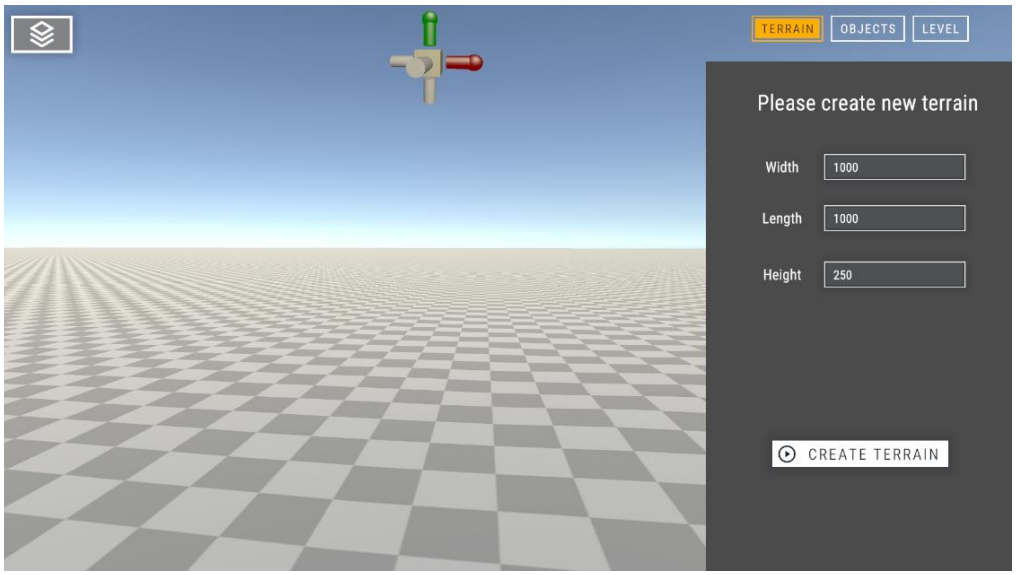


The instructor can go back and forth while watching the replay, pause it and also control the playback speed.

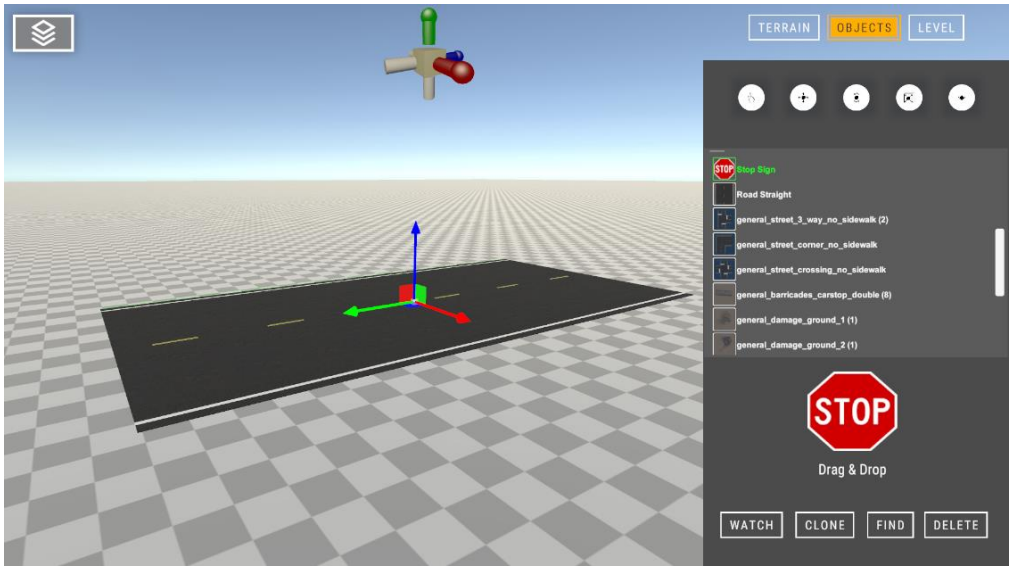


After the instructor reviews the driver's session, they will be directed to the feedback window. Here, the instructor has the option to write their feedback on the driver's performance or skip this step if they choose not to provide any comments.

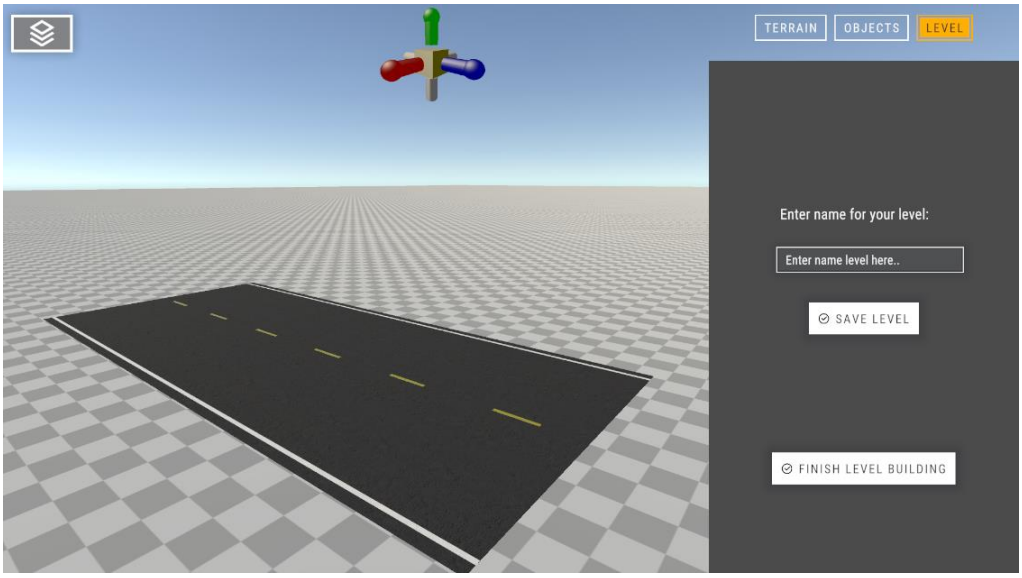
2.2.5 Create Track Flow



The instructor starts by selecting the size of the "world" he would like to create in the 'Terrain' tab and then clicks create terrain.

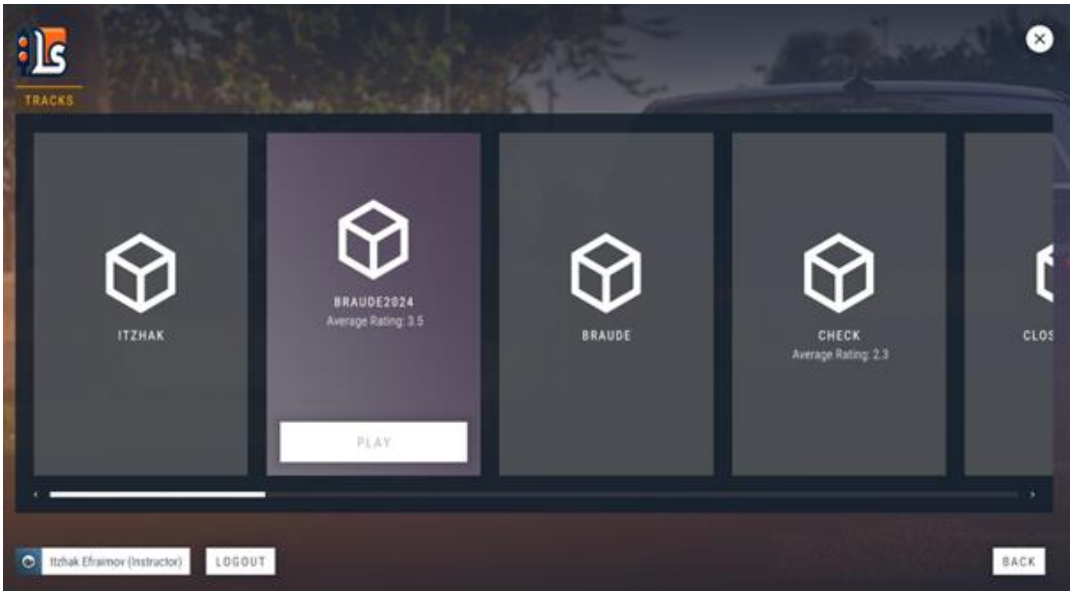


Afterward, in the 'Objects' tab, the instructor can pick any object available in the editor, such as roads of various shapes, traffic signs, and even vegetation. Objects are interacted by clicking on them. Each object can be moved in any direction and rotated by switching to a different mode in the editor's toolbar above the objects list. Objects can also be scaled in all directions. For added ease of use, the instructor can duplicate existing objects on the map by selecting them and clicking 'Clone'.



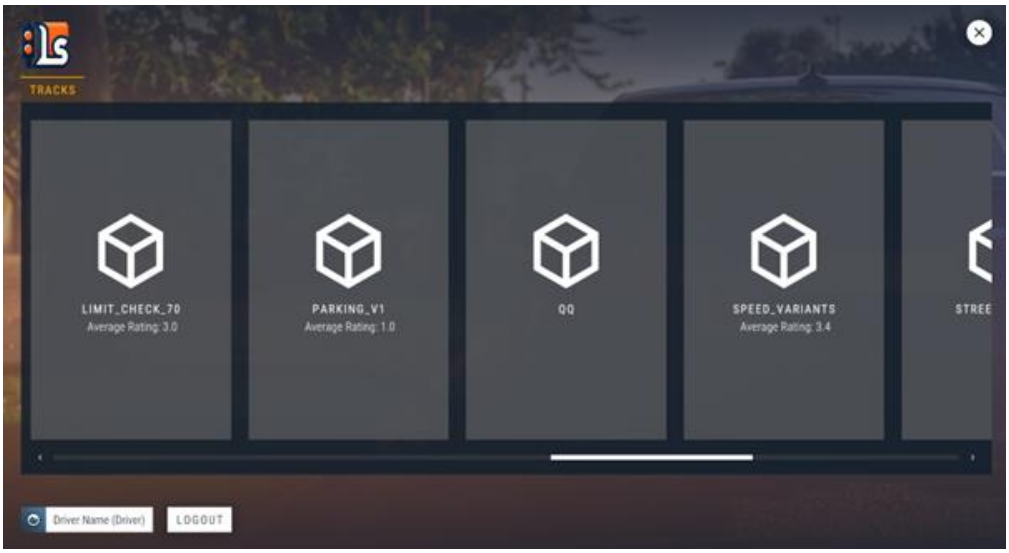
Once the instructor is satisfied with the current progress, they will click on the 'Level' tab to save and upload the creation to the cloud. The instructor will choose a unique name for the level and click 'Save Level' to complete the upload. To return to the home screen and exit the editor, the instructor will click 'Finish Level Building'.

2.2.6 Instructor Browse Tracks Flow



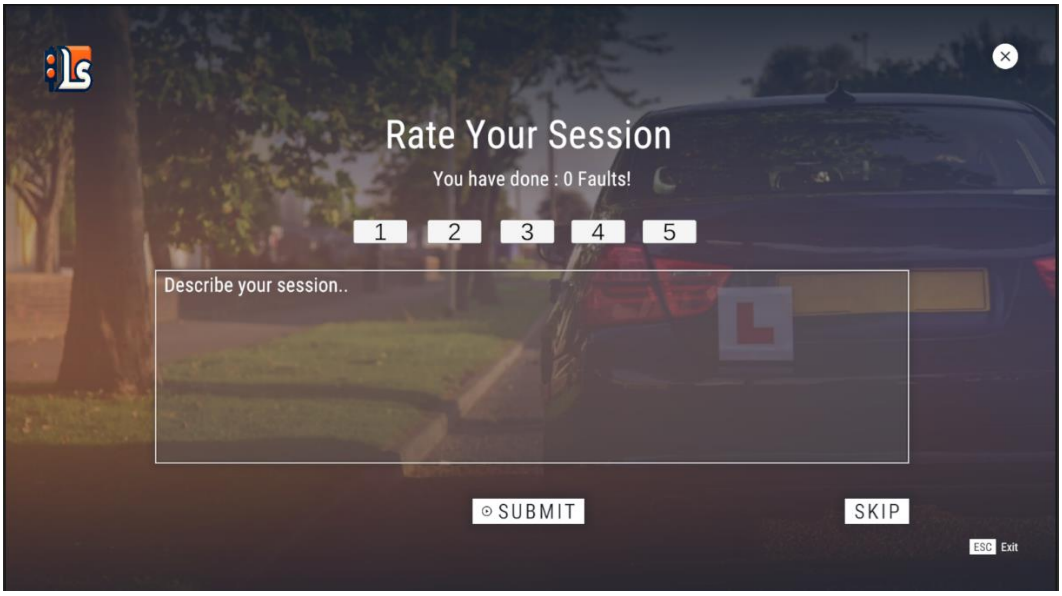
If the instructor wishes to browse his creation(s) or wants to modify/update it, he can select the track and it will be opened in editor mode, just like when he first created it.

2.2.7 Driving Session Flow



Once the driver is logged into his account, the screen he will be showing is the track browser like the instructor; however, in this case, the driver can see all tracks from all instructors and play them by clicking the box.

Now the driver is free to roam the world (driving with 'WASD' keys), changing camera angle (using the mouse and 'C'), using the car's indicators ('Z' and 'X") and even turning on the headlights ('L') or emergency lights ('J'), after he feels satisfied with the session “finish session” button will be clicked to proceed.



After the driver completes their session, they will be redirected to the feedback window. Here, the driver has the option to provide feedback regarding their satisfaction with the track. If they are not interested in offering feedback, they can choose to skip this step.

2.3 Challenges and Solutions

Challenge: Ensuring seamless communication between the Unity project and Firebase, especially when managing user data, session replays, and rankings across multiple platforms (web dashboard and Unity).

Solution: Use Firebase SDK specifically designed for Unity to simplify integration. If switching to a custom database, updating the API key and configuration files.

Challenge: Storing and retrieving session replays could increase storage costs and affect performance as the number of users and replays grows.

Solution: Optimize replay storage by compressing replay files or storing only essential data in byte array for reconstruction. Use Firebase's cloud storage efficiently with caching mechanisms to reduce load times.

Challenge: Ensuring smooth transitions between the driving session, session replays, and feedback windows for both drivers and instructors.

Solution: Implement an efficient flow between different stages of the session (driving, replay, feedback). Use state management systems in Unity and Firebase to ensure that users are smoothly redirected to the appropriate screen.

2.4 Decisions

1. We chose Firebase for the Driving Lessons Simulator due to its seamless integration with Unity, real-time data synchronization, and ease of use. Firebase's Unity SDK makes it intuitive to implement features like authentication, data storage, and real-time updates directly within Unity scripts. This was essential for our project, as we needed instant synchronization for session replays, live feedback, and track ratings. Additionally, Firebase's cloud infrastructure scales effortlessly as the project grows, ensuring the system remains responsive and secure without requiring extensive backend management.

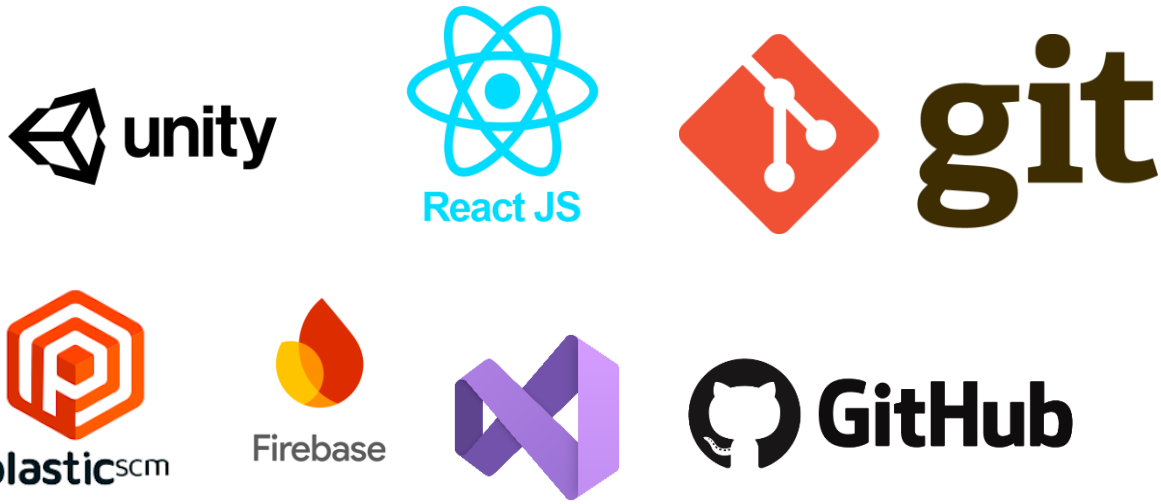
2. Finding a good UI asset for the Driving Lessons Simulator project is essential to ensure an intuitive, user-friendly experience for both drivers and instructors. A well-designed UI enhances usability by providing clear, accessible controls and navigation, reducing the learning curve for users interacting with features like track selection, session playback, and ratings. It's especially important when managing multiple roles in the project, as the interface must guide users seamlessly through different actions without overwhelming them. Additionally, a good UI asset helps create a polished, professional look, contributing to overall project quality, while saving development time by offering pre-built, customizable elements.

3. We decided to use Plastic SCM instead of Git for version control in the Driving Lessons Simulator project due to its superior compatibility with Unity and its ability to handle large binary files efficiently. Plastic SCM is specifically designed for game development environments, where large assets like 3D models, textures, and audio files are frequently used. Unlike Git, which struggles with large files and merges, Plastic SCM optimizes asset handling, allowing smoother collaboration between team members. This decision helped ensure smoother asset management and better project performance.

2.5 Tools Used

In the Driving Lessons Simulator project, we used a variety of tools to streamline development. Unity served as the primary engine for building the 3D simulation, while Visual Studio 2022 provided a robust environment for writing and debugging C# scripts. Git handled version control for code, and Plastic SCM managed large game assets efficiently. Firebase was essential for managing user authentication, real-time data, and storage, ensuring scalability. For the web dashboard, we employed React to create a dynamic and responsive

interface for instructors and drivers to view ratings and manage tracks. Together, these tools enabled a smooth workflow and efficient project management.



3. User Manual

3.1 Welcome to Driving Lessons Simulator

The **Driving Lessons Simulator** is a Unity-based application designed for driving instruction and training. It allows users to experience driving in various environments, while instructors can evaluate performance and create custom driving tracks. The simulator offers two main roles: **Driver** and **Instructor**, each with specific features and functionalities.

3.2 System Requirements

Before installing Driving Lessons Simulator, ensure your system meets the minimum requirements:

System requirements:

- Operating System: Windows 10/11
- Processor: Intel Core i5 or higher
- RAM: 8GB or more
- Graphics: NVIDIA GTX 960 or higher
- Storage: 5GB of free disk space
- Other: Internet connection for saving and retrieving session data

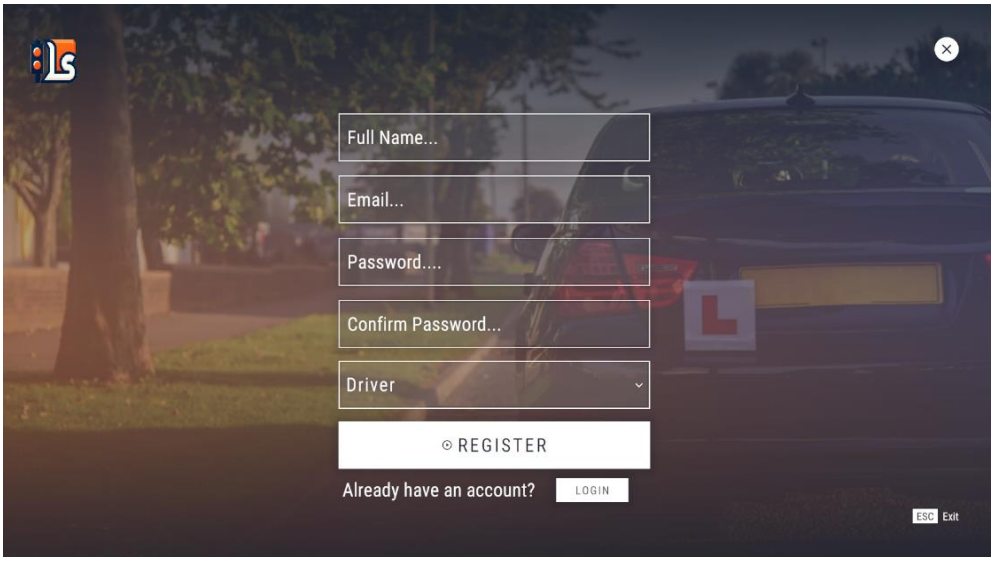
3.3 Getting Started

Installation

- Download Driving Lessons Simulator containing of production folder from: https://github.com/ilyaveruk/Driving-Lessons-Simulator.
- Click on the executable file of the game, Driving Lesson Simulator.exe.

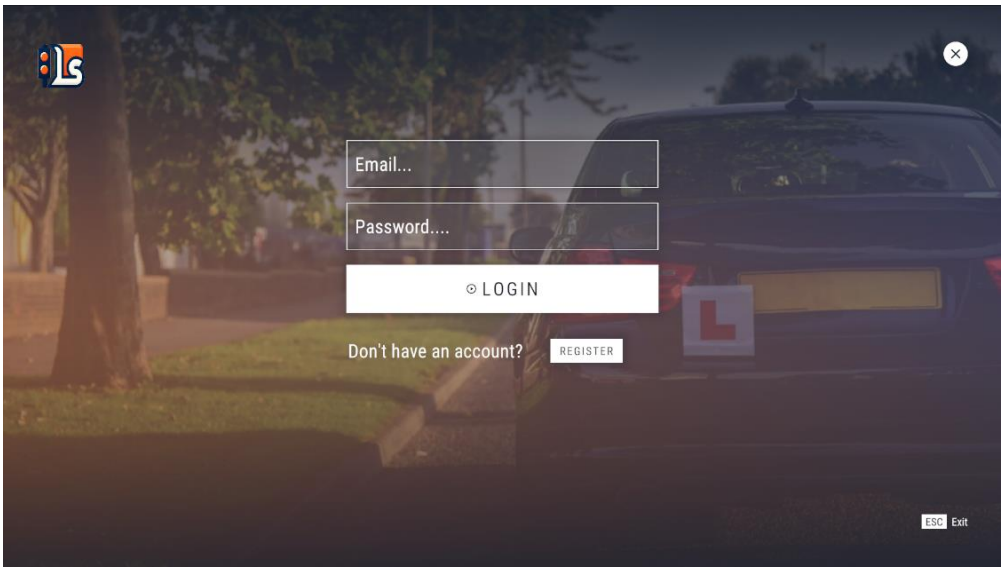
Creating an Account

- Launch Driving Lessons Simulator and click ‘Register’ button on the login screen.
- Enter a Name and E-mail.
- Create a Password and confirm it by entering it again.
- Click ‘Register’ to register your account.



Logging In

- Enter your E-mail and Password.
- Click ‘Login’.
- If the credentials are incorrect, a message will inform you what the problem is.

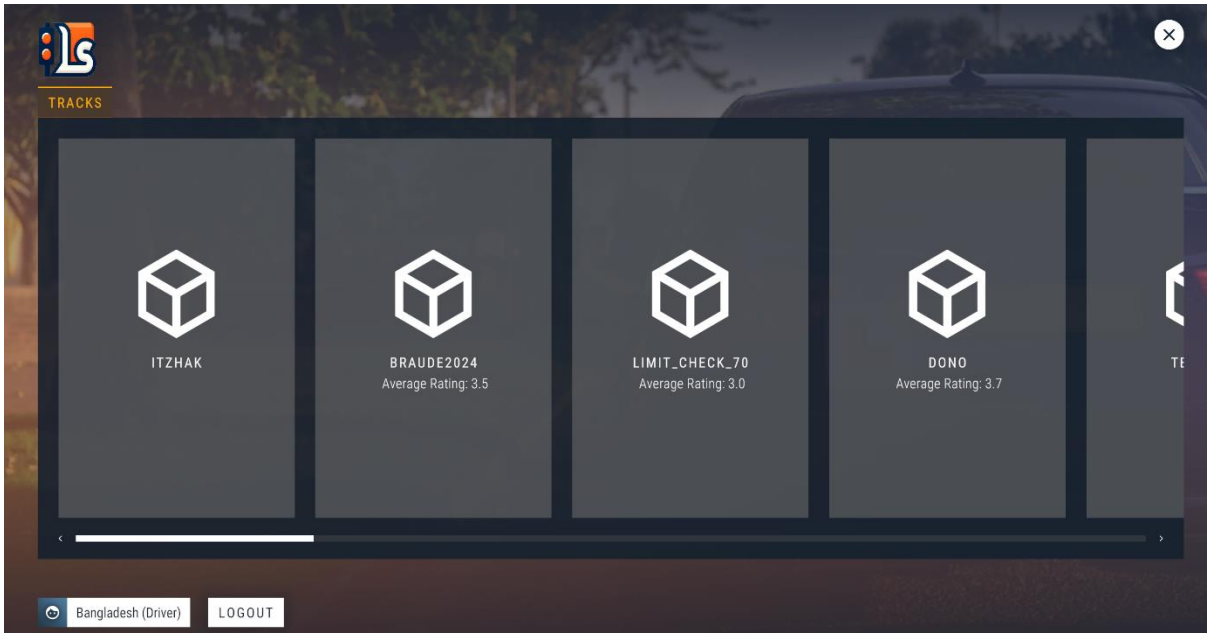


Navigating the Main Menu

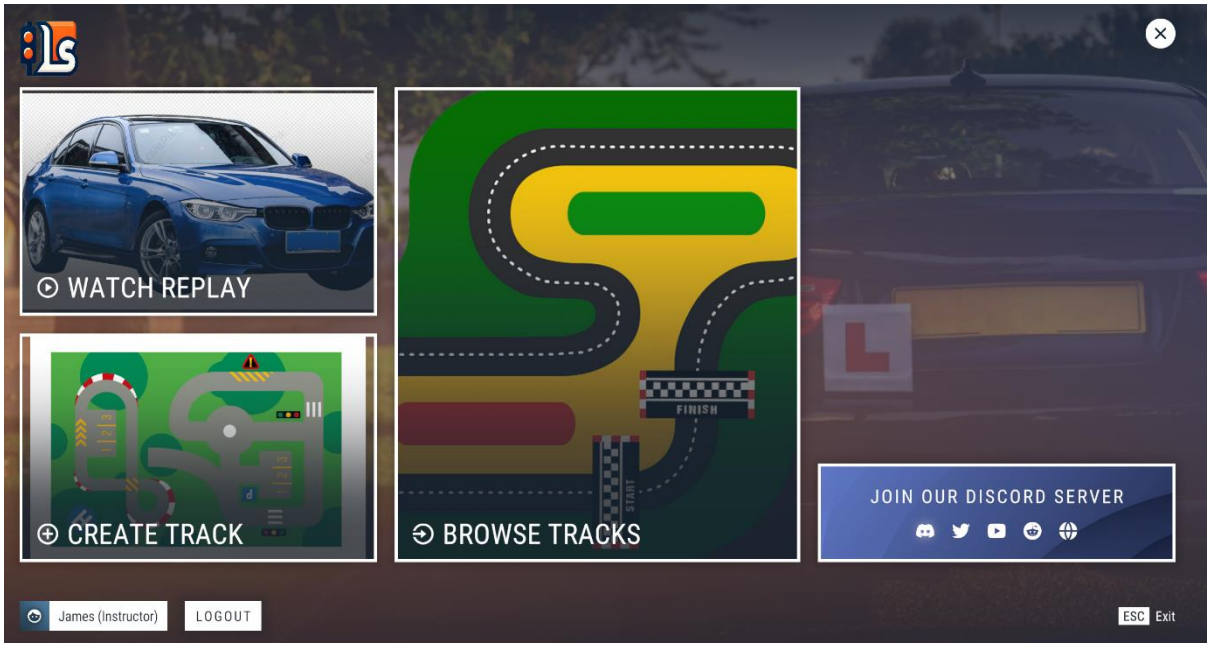
Upon successfully logging in, the system will automatically redirect you based on your user type:

- **For Drivers**: You will be taken directly to the **Track Selection** screen, where you can browse available tracks and start a new driving session.

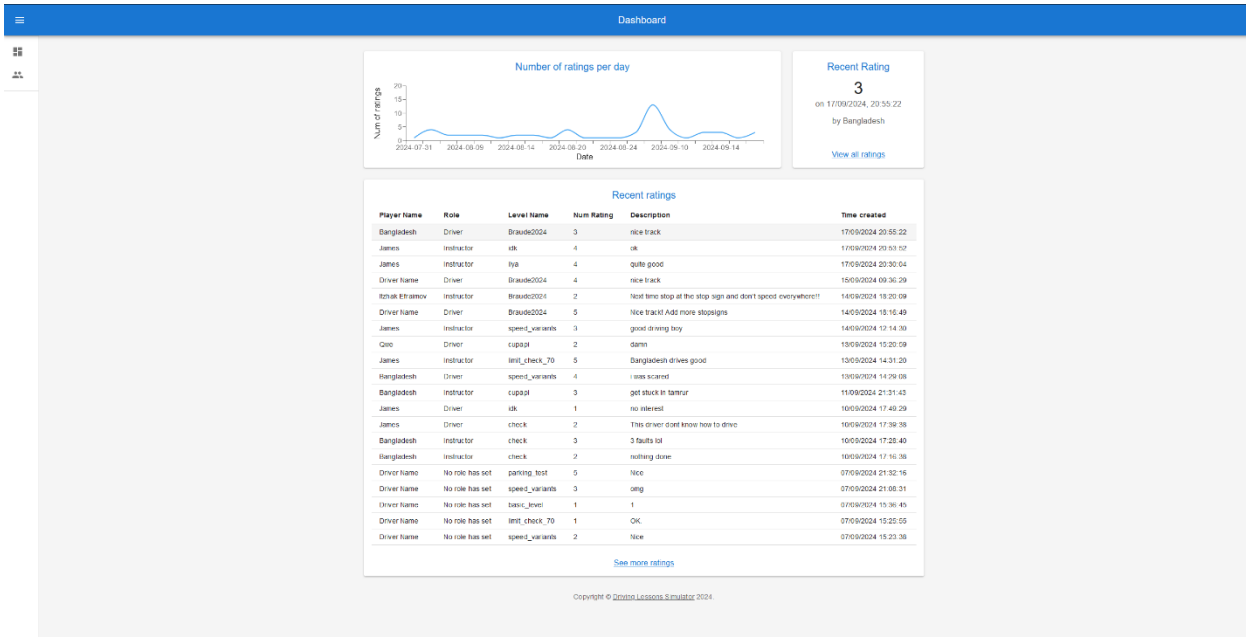
- **View Ratings**: To review your past track ratings and feedback, visit the DLS Dashboard for a summary of your session ratings and comments.



- **For Instructors**: You will be directed to the **Instructor Dashboard**, which offers several options, including:
  - **Watch Replays** of driver sessions for evaluation.
  - **Edit Existing Tracks** to refine or modify previous designs.
  - **Create New Tracks** using the track-building tools.
  - **View Ratings and Feedback**: For a comprehensive view of ratings given to drivers and track performance, visit the DLS Dashboard where you can manage and review all evaluations.



The interface is tailored to your role, ensuring quick access to the relevant tools and features for your session.



DLS Dashboard

Driver Role

A list of tracks, each with its own difficulty level, layout, and features, will be displayed. During the session, the driver's performance is tracked, controlled, and adhered to road rules. The driver can use the session as long as he wants. After the session, the driver is prompted to rate

the track based on their experience. They can provide a star rating (1-5) and, optionally, leave a comment to offer feedback on the track's design or difficulty.

Instructor Role

Upon logging in, the instructor is taken to the Instructor Dashboard, where they can manage key tasks, such as reviewing driver replays, creating new tracks, and editing existing ones. By watching replays, the instructor can analyze driver performance, provide ratings, and give detailed feedback. The Track Creation Tool allows instructors to design custom tracks or modify existing ones to challenge drivers with various obstacles and scenarios, ensuring tailored training experiences.

3.4 Game Controls and Interface

Driver Controls:

- **Steering**: Use the **A** and **D** keys to steer the vehicle.
- **Acceleration**: Press the **W** key to accelerate.
- **Braking/Reverse**: Press the **S** key to brake or reverse.
- **Handbrake**: Press the **Spacebar** to engage the handbrake.
- **Camera View**: Press **C** to toggle between different camera angles (e.g., first-person, third-person).
- **Blink Left**: Press **Z** to signal left.
- **Blink Right**: Press **X** to signal right.
- **Lights**: Press **L** to turn on the standard headlights.
- **High beams**: Press **K** to activate high beams.
- **Emergency Lights**: Press **J** to engage emergency lights.

Driver Interface:

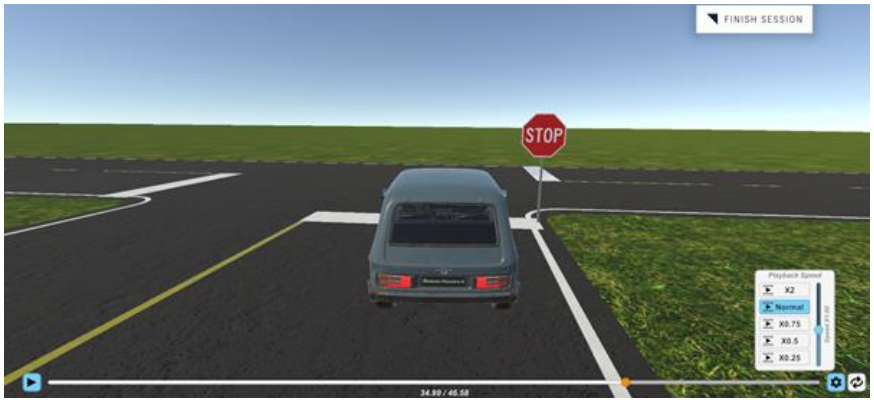
- **Track Selection**: After logging in, navigate through available tracks using the mouse. Click on a track to start a session.
- **In-Game HUD**: Displays current speed, RPM and session time.

- **Rating Screen**: After completing a session, use the mouse to select a rating (1–5 stars) and provide feedback.

Instructor Controls and Interface:

- **Replay Controls**:
  - Use the mouse to play/pause, rewind, or fast-forward replays.
- **Track Creation/Editing**:
  - Use the mouse to drag and drop track elements, place obstacles, and modify the track layout.
- **Dashboard Navigation**: Use the mouse to select options for creating new tracks, editing existing ones, or viewing replays.

The keyboard controls are specifically designed for driving actions, while the mouse is utilized for navigation and interactions within the interface, ensuring a streamlined experience for both drivers and instructors.

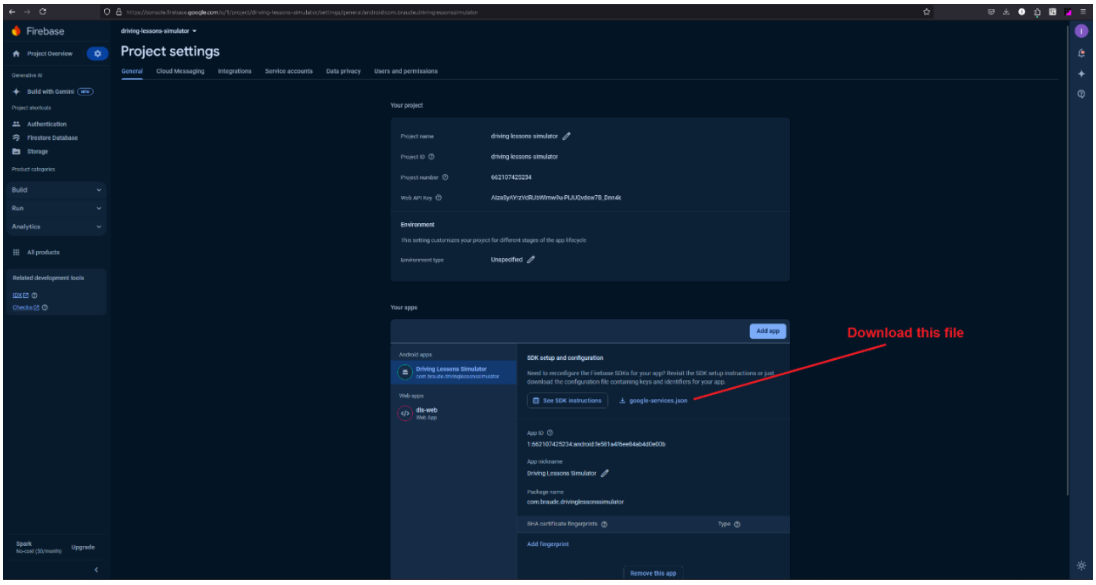


4. Operation and Maintenance

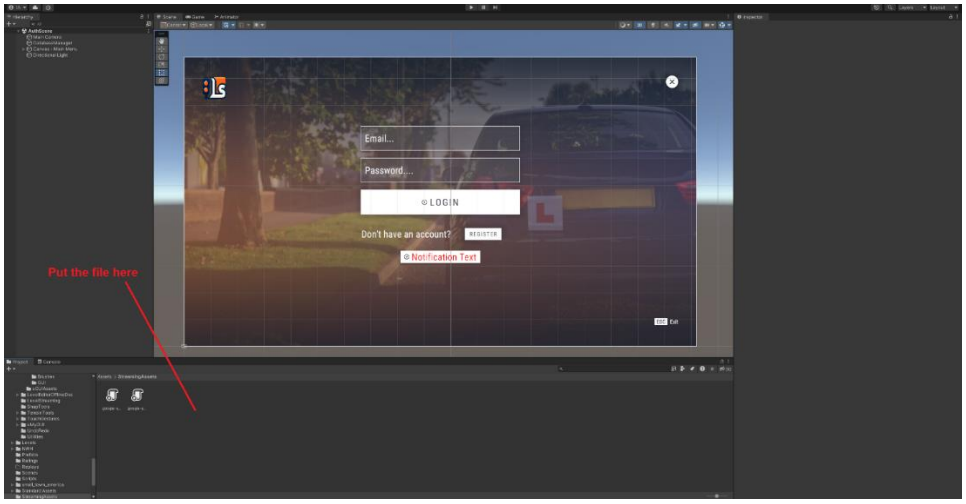
4.1 Setting up the database

In this project, we opted to use Firebase due to its seamless integration with Unity, which significantly simplifies the development process. To streamline user access, we configured the database to be public, allowing users to download and play the game without additional setup. However, if you wish to configure your own database, simply replace the default Firebase configuration by placing your custom “google-services.json” file in the following directory: Assets -> StreamingAssets.

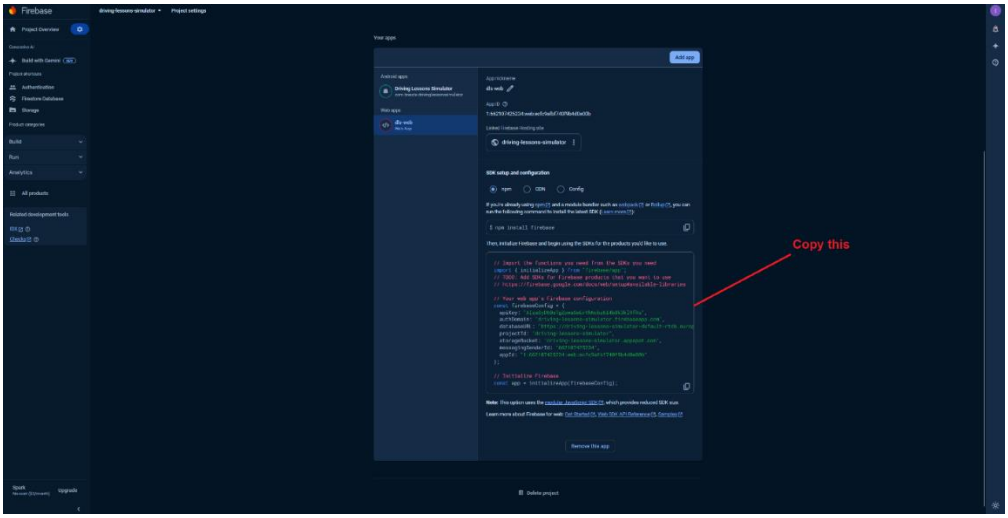
This approach ensures flexibility for developers who want to modify or personalize the backend. After setting up your database simply download the file to your computer.



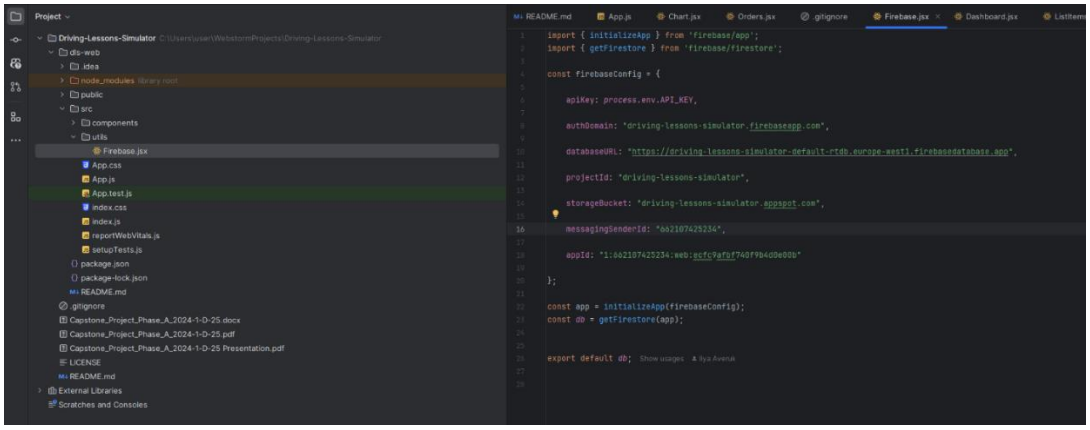
After you downloaded the google-services.json file**,** place it inside the directory: Assets->StreamingAssets



If you decide to use your own database, you will also need to include an API key for the web dashboard. After setting up your database for the web service, copy the following details: the API key, database URL, project ID, and any other required Firebase configuration parameters. These details will be used to establish a connection between the web dashboard and your Firebase database.



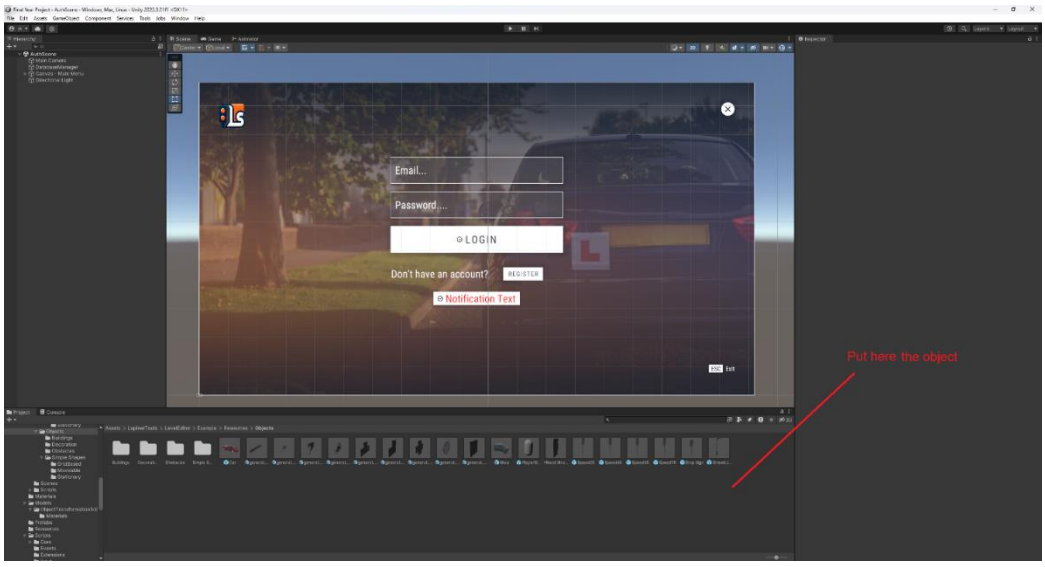
Within the dls-web project, cloned from GitHub, navigate to the Firebase.jsx component. Open this file and paste the provided code at the appropriate location. This ensures the Firebase integration is correctly set up for the web component of the project



Note: You must deploy your own project to Vercel to make it work

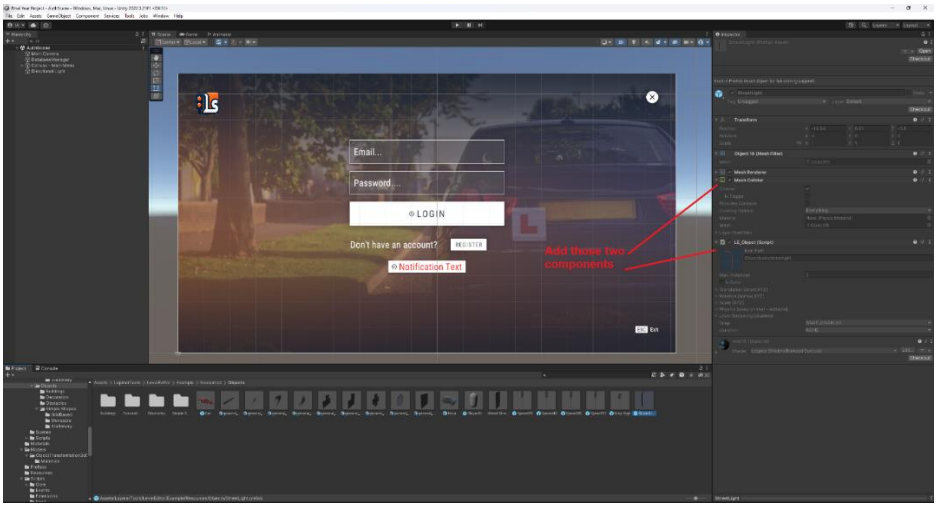
4.2 Adding objects to the Editor

To add additional objects to the project for future development, please follow these steps: First, clone the project from GitHub and open it in Unity using version 2022.3.21f1. Once the project is loaded, navigate to the following directory in the project's folder structure: Assets->LapinerTools->LevelEditor->Example->Resources->Objects, place the desired object within this folder to include it in the project.



After placing the object, you will need to add two components to ensure the system recognizes it as part of the Editor. First, add a Mesh Collider component and enable the Convex attribute. Next, add the LE_Object component, where you will need to specify the relative path for the object's icon. The icon should be placed in the following directory:

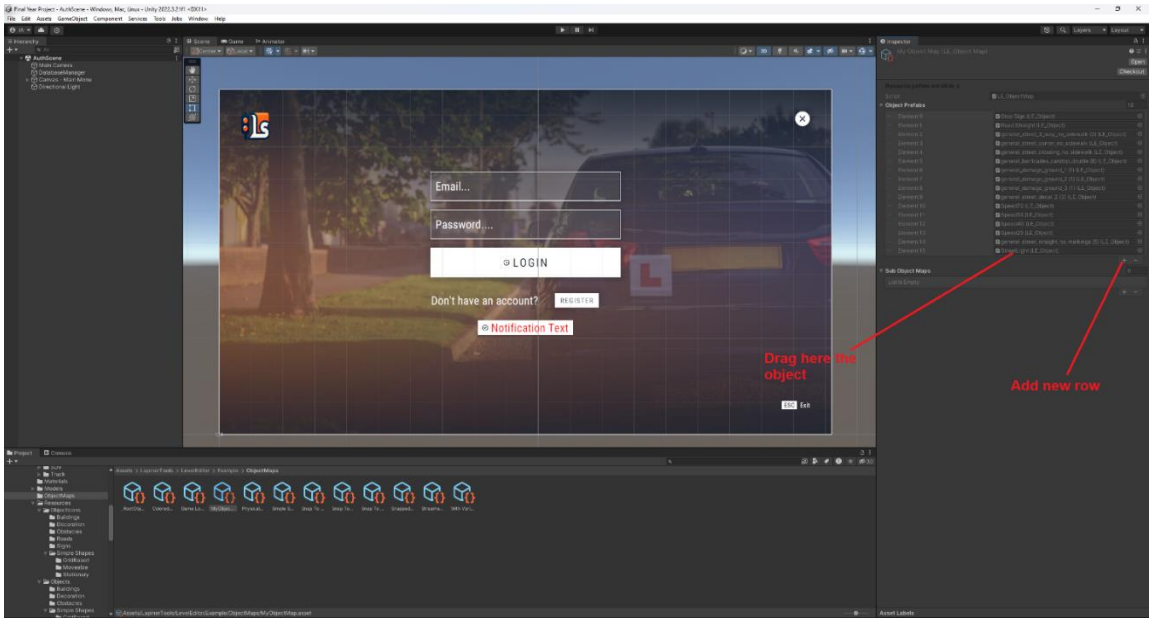
Assets->LapinerTools->LevelEditor->Example->Resources->ObjectsIcons.



Once you done with the previous steps, navigate to:

Assets->LapinerTools->LevelEditor->Example->ObjectsMaps,

Select the **MyObjectMap** object. In the Inspector window, locate the array on the right-hand side. To add your newly created object, click the + button and drag your object into the new line in the array. This ensures that your object is included in the map and can be used within the Level Editor.



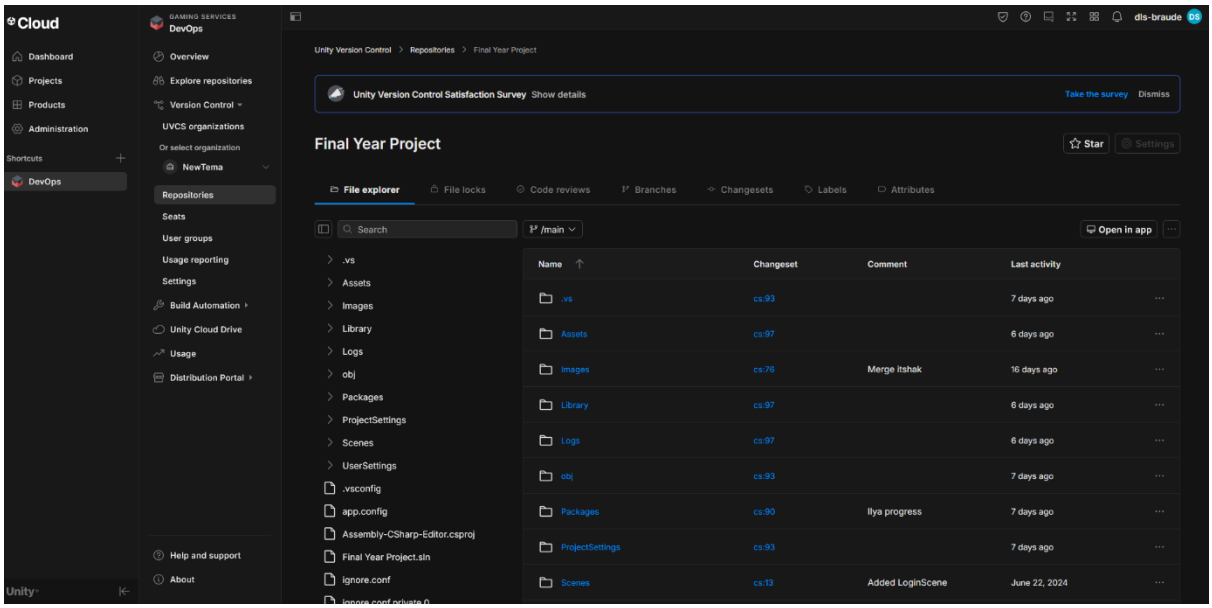
4.3 Source Code Control

To access and modify the simulator's source code, use Unity's Plastic SCM version control.

Navigate to Unity Cloud Plastic SCM and log in with the credentials:

- **Email**: drivinglessonssimulator@gmail.com
- **Password**: Braude2024

From there, you can view the commit history, clone the repository, create branches, and push changes after testing. This version control system allows effective collaboration and tracking of modifications in the simulator project.



5. Results and Conclusions

From the project, we learned the importance of creating a user-centered design that effectively separates functionality based on roles. By focusing on the unique needs of both drivers and instructors, we managed to create a system that supports each user's goals whether it's hands-on driving practice or track management and performance evaluation. We also learned how crucial real-time feedback and replays are in educational tools, as they allow for detailed assessment and improvement.

We managed to achieve this by breaking down complex tasks into intuitive workflows, implementing features like the replay system and feedback mechanisms to enhance learning. The use of Firebase for session and data management helped us maintain scalability and reliability. Throughout the development process, we maintained a focus on user experience, ensuring that controls, track creation tools, and feedback systems worked together seamlessly to create an effective and engaging training simulator.

6. Further improvements

To further enhance the **Driving Lessons Simulator**, several key improvements are proposed:

- **Dynamic weather conditions**: Adding weather effects like rain, fog, and snow would create more realistic driving scenarios.
- **Interactive elements on tracks**: Incorporating more objects and interactive elements would diversify training challenges for drivers.
- **AI assistant**: Introducing an AI to provide real-time guidance during sessions would help drivers improve their skills more effectively.
- **Instructor note-taking**: Allowing instructors to write notes while watching replays would streamline the feedback process.
- **Track validation system**: Implementing a system to automatically check the correctness of tracks (e.g., avoiding misplaced stop signs) would ensure training quality.
- **Web dashboard improvements**: Enhancements like visualizing rating trends, sentiment analysis, and comparative analytics would offer deeper insights for instructors and drivers.
- **Predictive insights and alerts**: Custom reports, predictive insights, and automated alerts could personalize the training experience.
- **Community features**: Introducing leaderboards and forums could foster engagement and encourage shared learning among users.

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. NWH Vehicles - https://nwhvehiclephysics.com/doku.php/Setup/VehicleController
5. Heat UI - https://docs.michsky.com/docs/heat-ui/ui-elements/button/
6. Ultimate Replay - https://trivialinteractive.co.uk/products/documentation/ultimatereplay_30/UserGuide.pdf
7. Lapiner Editor - http://www.freebord-game.com/index.php/multiplatform-runtime-level-editor/documentation
8. Google Firebase - https://firebase.google.com/docs/unity/setup
9. ChatGPT - https://chatgpt.com/
10. Unity Game Engine - https://docs.unity.com/
11. Unity Basics Tutorial - https://www.youtube.com/watch?v=99IecSRWA5Y&list=PLtLToKUhgzwnk4U2eQYridNnObc2gqWo-&index=2&ab_channel=Mike%27sCode