AR Mini Flying Car Final Report

**Title and Introduction: Brief explanation of your AR app**

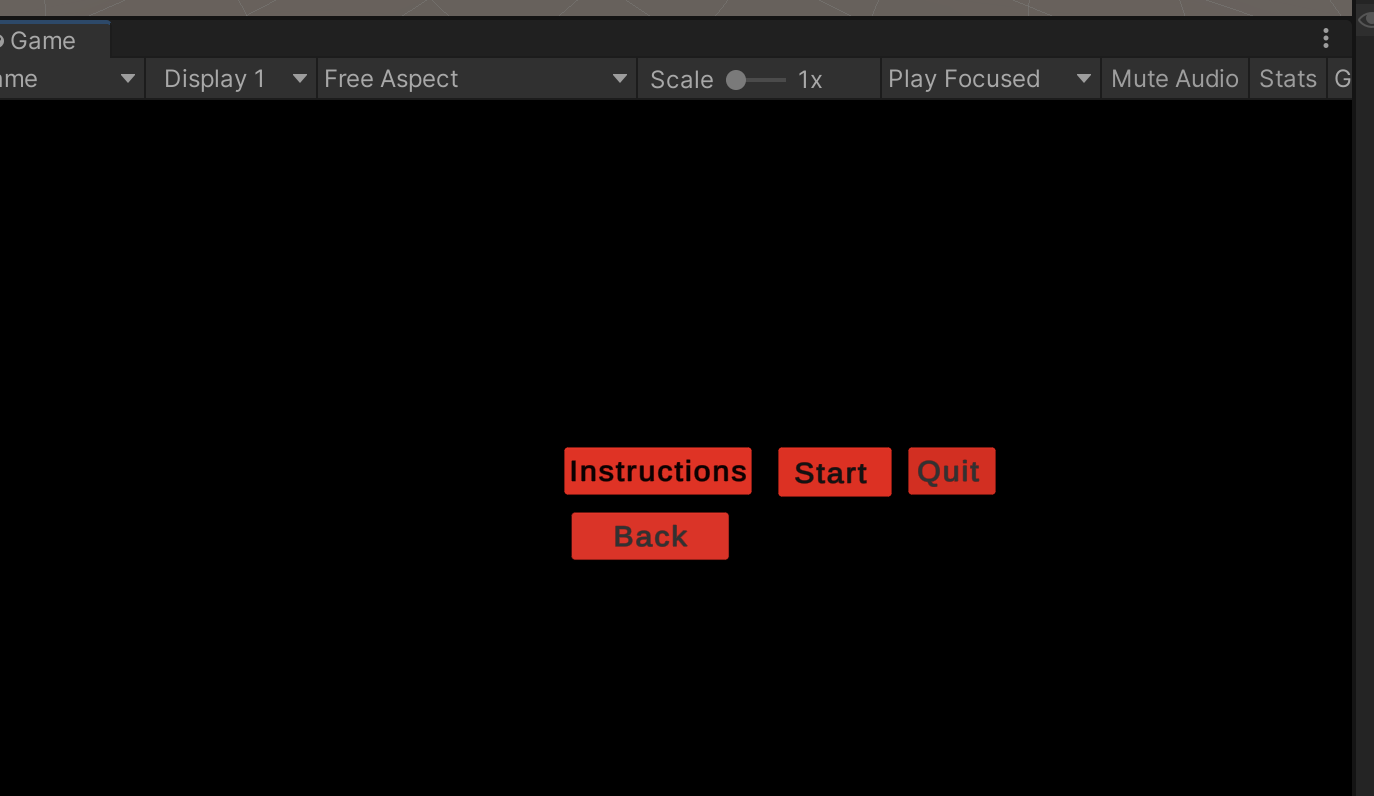
My Ar App is called Mini Flying Car. In my app, the user is first shown 3 buttons: instructions, start and quit.When the user clicks the instructions button, the instructions are shown and the user has an option to go back to the main menu scene, where they see the three buttons again-instructions, start and quit. Upon clicking start, the user enters the game scene. In this scene, the start and instructions button are no longer present.

In this scene, the user clicks assemble wheels to assemble the car. The user does not have the option to manually assemble the car. This is because for manual assembly to work, the user would have to place the wheels precisely over the wheel collider, a task that would be too complicated for the user. After automatically assembling the car, the engine starts. You can hear the engine sound come on and the car is ready to go.The user uses the horizontal joysticks to move the car forward and back and the vertical joysticks to move the car left or right or adjust the angle of the car. The user also has the opportunity to play the radio by pressing the play radio button. When this radio button is clicked ‘Maneater’ by Daryl Hall is played. Finally, the user has the opportunity to switch between the colors red and yellow.

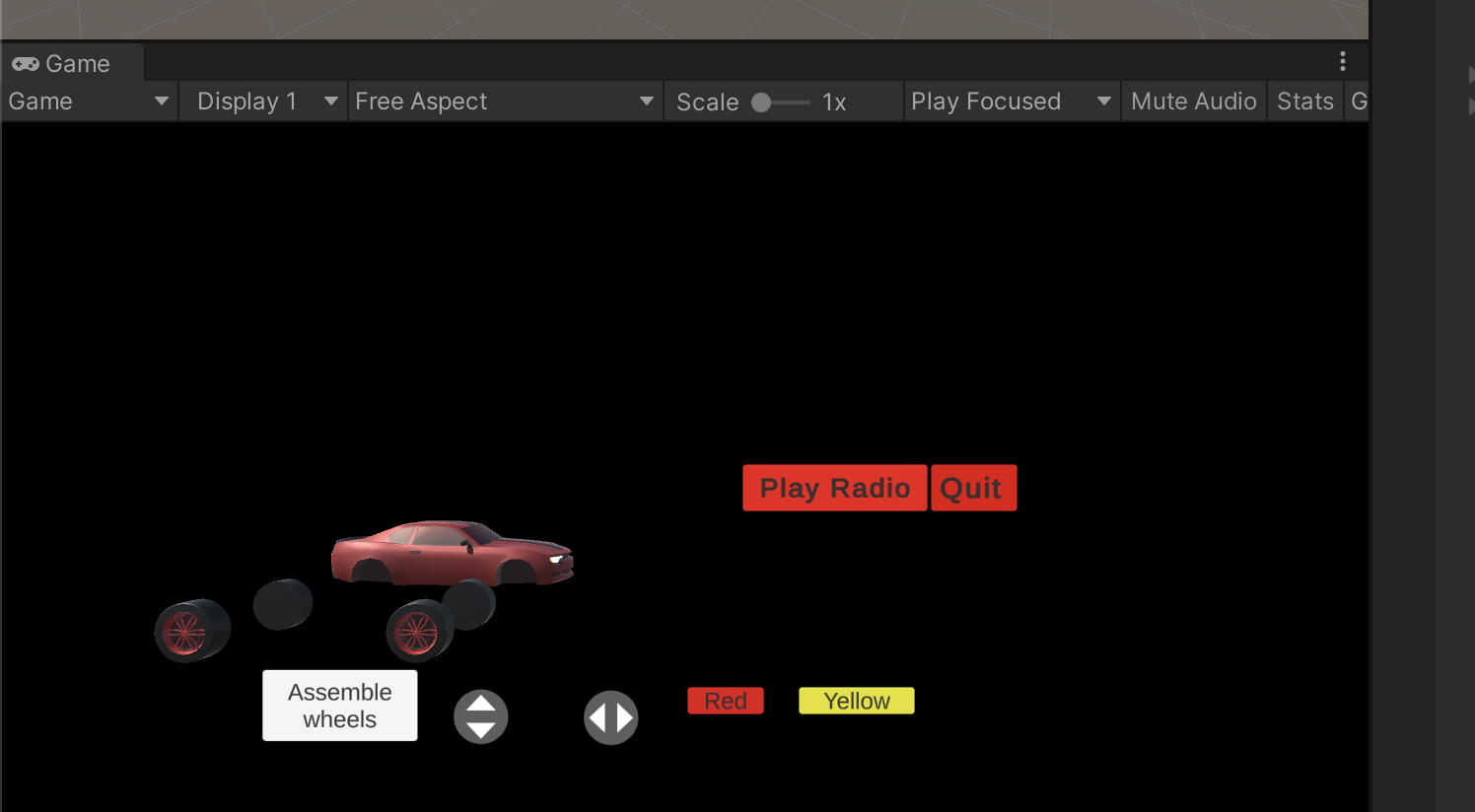
Demonstration Images:

\*Note that these images are shown in the unity scene editor. My video demo will demonstrate this in AR)\*

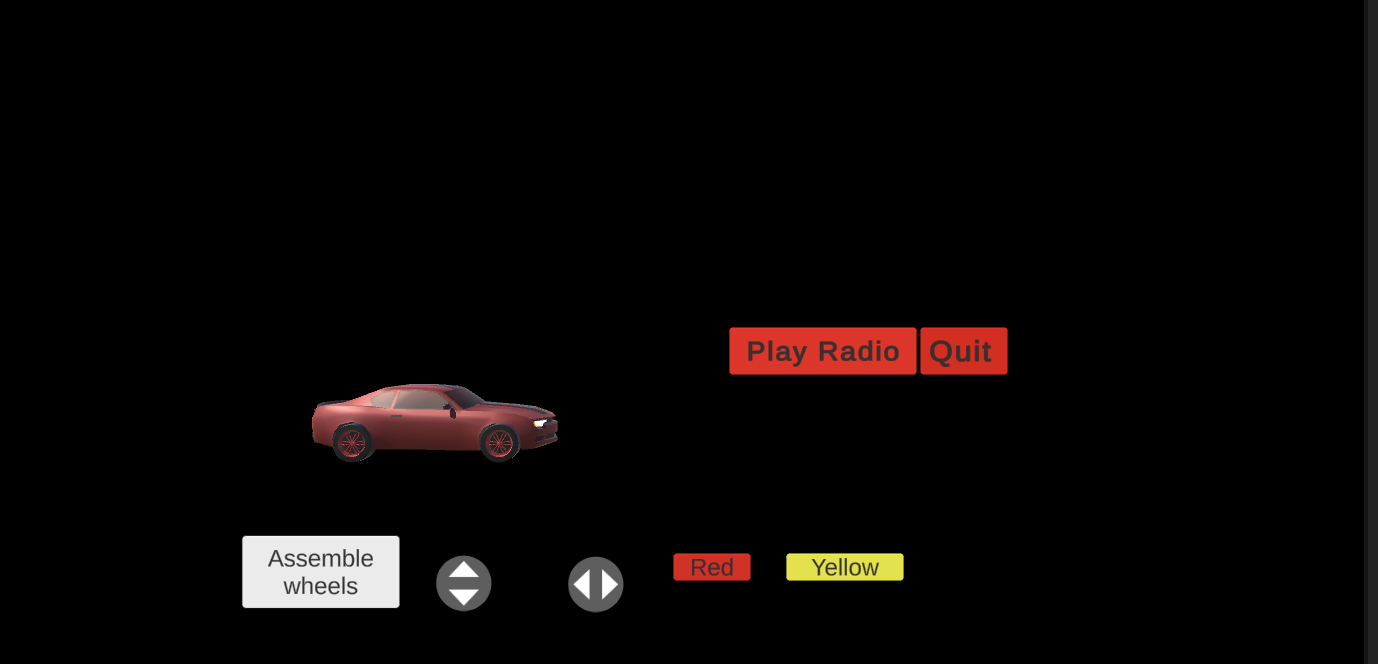
Main menu



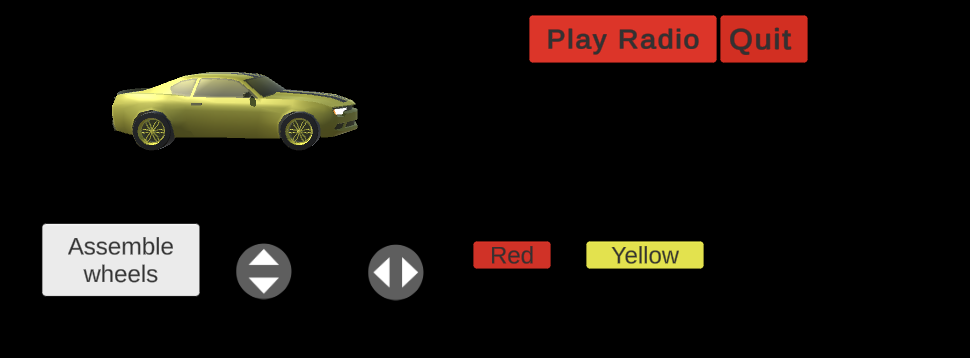
Upon clicking start



Upon clicking assemble wheels



Change car color to yellow



**AR Application Design: What are the key design considerations used to build the AR app?**

How complex the app was to build and how well it met the AR Requirements were my key considerations in the AR App development. I could have made a more creative and complex AR application if it were not for the time constraints and the soon approaching deadline. I simplified my project numerous times in order to meet the basic requirements by the deadline.

When a user plays with my AR Application, I would assume that they would want to do more.

The second key consideration was being user friendly. My AR application is easy to use. The user simply clicks buttons or uses joysticks to achieve the desired result. I make sure that the user is not doing anything that they should not be doing. My AR App is sequential. For instance, the car only pops up after the user clicks start. The user can only move the car after it is assembled and the engine comes on. The user can only play the radio and change the car when the car is present in the game scene. In total, I only have 8 buttons and two joysticks which are relatively straightforward to use.

**Integration: The build process. Any third party software tools or hardware used?**

The only third party software and hardware tools used were Unity and Xcode. Other than this, no other external tools were used in the project.

**User experience: what you have done to optimize the user experience for AR the app (referring to required AR application design features).**

To optimize the user experience for the AR App , I included the must have features for my AR app in this project. I had an assemble button which upon clicking, automatically assembled the 4 wheels to their respective positions on the car, and the wheel colliders were positioned correctly. I gave the user the ability to remotely control the car and interact with it by moving it with the horizontal or vertical joysticks. I gave the user the option to change the color from yellow to red using on screen buttons. Narration, music, or sound effects were included( as mentioned earlier). The virtual objects included the car. The digital objects included the 8 on screen buttons and the 2 joysticks.

**Conclusions: Considering initial design goals, what worked, what needs improvement, what did not work?**

Considering initial design goals, there were a few issues. The AR Plane manager in my project detects the floor higher than it usually is. I have watched almost every AR unity tutorial on plane detection on the web , but the plane manager still fails to detect the floor. It keeps on thinking that my table is the floor. This is one area for improvement in my project. To trouble shoot, I called my car a mini flying car instead. I also decided to not give the user an option to brake to simplify my car movement script. Perhaps as a future side project, I can make these changes for a better user experience.

**Suggestions for further improvements**

The only suggestions I have for future AR projects is to study the tutorials more in depth to make a more complex AR project that works exactly as is desired.

**Lessons Learned - your personal journey throughout this project.**

One of the most important lessons that I learned in this journey is to finish the entire project at least 1 week before it is due. This will give me more time to troubleshoot. I spent the last couple of days finishing the last 20% of the project. I saved it for last because I thought it was “easy”. However, I underestimated how much time I would spend troubleshooting and creating good UI. I also did not consider the time needed to work on the extra documents needed for the project, such as the paper, the demo video, the user guide and presentation. If I had not stayed overnight for the last couple of days working on this project, I am not sure that it would have been completed.

Another important lesson that I learned was that I should have taken the intermediary steps leading up to the project more seriously. If I had taken more time to throghougly go through the AR tutorials, I would not have had to troubleshoot so many bugs in my project. The AR tutorials taught me almost everything I needed to know about the project. Perhaps if I had studied the tutorials in depth and gone to the TA when I was having trouble with the tutorials, I could have finished my AR project faster and made it better.

**Feedback to the instructor, such as choice of AR dev platform, etc.**

My feedback to the professor is to provide us with more AR resources. I realized that the internet is full of resources for VR projects, but it is lacking in resources for AR projects. Since Unity changes so often, the AR tutorials are sometimes outdated. I would also suggest that he help us find more recent AR tutorials. Also, another suggestion would be to provide extra credit options for particularly creative and more advanced AR projects. This can challenge the students even more.

Finally, future students who plan to use face tracking in their AR Application should be informed that AR Face Tracking is only available on iPhones 10 and above. It is also not available on some ios phones. I spent a significant amount of time trying to troubleshoot the AR Kit Face tracking for one of the AR tutorials only to find out that it was not available on my mobile device.

**Appendix: AR App User Guideline - Simple instructions about how to install your app and enjoy the AR experience, with screen shots from the app.**

**Unity Download**

* Download [Unity for your project](https://youtu.be/5aqCVUt8KY4): version 2021.2.7f1

**Xcode Setup**

* Prepare to download xcode: make sure you have the latest ios version and the latest macOs.
* Download xcode from the apple store. Then go to xcode in the menu bar and click on preferences. Open the accounts tab and add your apple ID by clicking on the plus button.
* After apple ID is selected, click on continue. Sign in with your apple ID
* Ensure that [developer mode](https://tipsmake.com/how-to-enable-developer-mode-on-iphone) is enabled on your iphone.
  + If you have an android, look up how to enable usb debugging for you specific mobile device

**AR Foundation Kit Download and Set Uo**

* Create a regular Unity 3D project
* Watch this tutorial from [0:00-1:42](https://youtu.be/wSf0qKoDC3o) to download the AR kit. Note that unity has changed and rather than downloading each individual AR package. You download all the AR packages at once.
* You do not need to worry about setting up the AR session origin, plane manager, ray cast etc. as all this is provided in the imported project for you.
* However, if you are curious about doing this in your own future projects, check out these tutorials:
  + [Getting Started With ARFoundation in Unity (ARKit, ARCore)](https://youtu.be/Ml2UakwRxjk)
  + [Unity AR Foundation Tutorial - Plane Detection](https://youtu.be/uWWiYfPTUtU)

**Standard Assets download**

Download this asset to your project (just in case) : [Standard Assets (for Unity 2018.4) | Asset Packs](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2018-4-32351#description)

**Importing project**

* Import the unity project recovered unity export.unitypackage into unity and under assets, open “ca project” scene
* If ‘ca project’ does not work as expected, use my backup project, ‘Final AR 03:21 Project.unitypackage’
* If errors persist in both, it is recommended that you read the user guide for a second time more closely and/or view the video tutorials provided to troubleshoot or access external resources.

**After everything is set up and your phone is linked to your laptop**

* Go to file, build settings and add ‘ca project’ to the open scenes
* In build, switch platform to ios or android
* Setting up AR project settings for Android:

[Unity - AR Foundation setup (iOS & Android) [SmartAR Plugin]](https://youtu.be/c-mHinPCFSI)

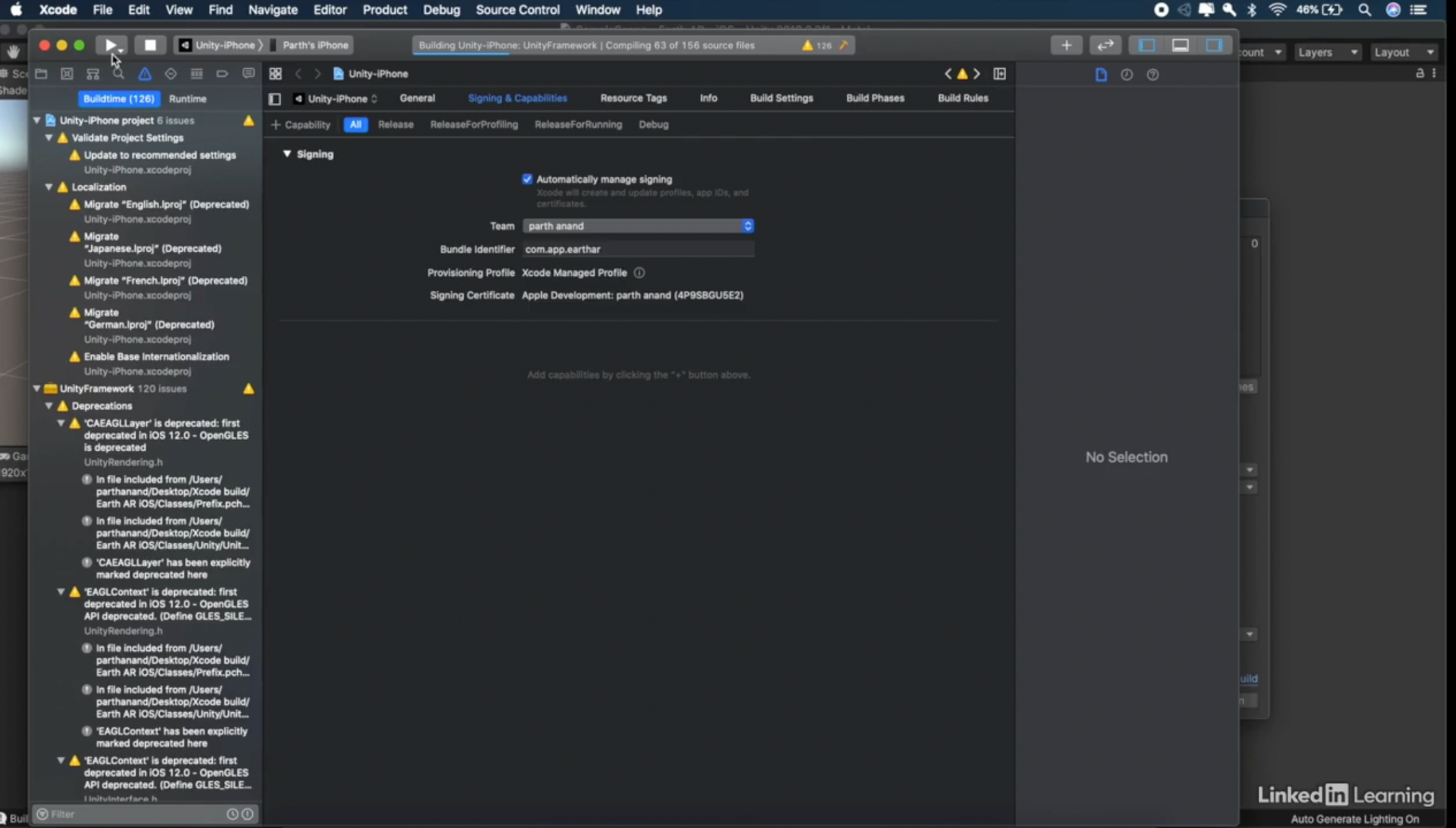
* Setting up AR project settings for Android for ios:

[Unity - AR Foundation setup (iOS & Android) [SmartAR Plugin]](https://youtu.be/c-mHinPCFSI)

* In build , click player settings. Then in project settings underneath player, make the following changes:
  + Add a bundle identifier and a camera usage description.
  + Change the target minimum ios version to 12.
  + Change architecture to arm 64 and check requires ar kit support.

**Building project**

* Go back to build settings. Then, build your project
* Choose a folder where you want to build your xcode build and it will open up in xcode.
* In xcode, click unity iphone . Then click signing and capabilities. Make sure you add a team name.
* Assuming that no xcode errors exist, select the run button as shown in the image below and make sure your iphone is unlocked.



* The AR Application is now built on your device
* See demo video for what your ar app should look like

Sources

Parth Anand(2019) ARKit and Unity: Build a Drivable Car in Augmented Reality source code

(Version NA) [Source

code].[https://www.linkedin.com/learning/arkit-and-unity-build-a-drivable-car-in-augment](https://www.linkedin.com/learning/arkit-and-unity-build-a-drivable-car-in-augmented-reality/adding-touch-controls?autoplay=true&u=74651410)

[ed-reality/adding-touch-controls?autoplay=true&u=74651410](https://www.linkedin.com/learning/arkit-and-unity-build-a-drivable-car-in-augmented-reality/adding-touch-controls?autoplay=true&u=74651410)

Unity Technologies(2018) Standard Assets (for Unity 2018.4)

(Version 1.1.6) [Source code].

[https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-201](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2018-4-32351#description)

[8-4-32351#description](https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2018-4-32351#description)

Mena(2020) ARCADE: FREE Racing Car

(Version 1.0.1) [Source code].https://assetstore.unity.com/packages/3d/vehicles/land/arcade-free-racing-car-161085#releases

Lovatto Studio(2015) UJoystick (Version NA) [Source code].

<https://assetstore.unity.com/packages/tools/input-management/ujoystick-49186#publisher>

Nox\_Sound(2021) Vehicle-Essentials (Version 1.0) [Source code].

https://assetstore.unity.com/packages/audio/sound-fx/transportation/vehicle-essentials-194951#description