Requirements Steps.

By Marwan Hany

# Assets Preparations

* 3 Vehicles with separated doors
* Image for marker
* Circle Image
* UI Elements
* Voiceovers (20 Sec maximum each)
* 3 Car engine sfx
* 3 car wheel photos (taken from the cars in unity)
* 3 car model photos (taken from the cars in unity)

# Guide Steps

1. Implement Marker Logic using 1st guide but don’t use multiple marker tracking
2. Implement Marker-less Logic using 3rd guide. Using only interact, rotate and horizontal movement logics, do not use resize.
3. Implement Project Setup
4. Implement Build Setup

# Ui Implementation

1. 1st page UI **Should** have 2 buttons that say “marker” or “markerless”
2. When a vehicle is pressed a UI will appear with buttons “color” and “car models” and “Vehicle Interactions” and “Wheel Customization” and a button with a picture of a camera this ui should be able to slide from left to right
3. When “color” button should open a ui with a color wheel
4. when car model is selected 4 photos of cars appear
5. when vehicle interactions is pressed a button toggle for “turn on engine” and “open car doors” and “rotate vehicle” with a slider underneath the rotate vehicle
6. When “wheel customization” is pressed buttons with photos of 4 different wheels appear each one different color
7. Fps counter at the top right
8. Floating panels near highlighted parts of vehicle showing different specs and part names for example “Engine V12” and a button “play audio”.
9. An x to go back to the previous ui on every ui implementation except the 1st page

# Logic Implementation

1. Use gesture implementation in the marker
2. Create a script when a vehicle is pressed the main ui appears from bottom to top
3. Add Or create animations for the doors opening and closing and add it to animation controllers
4. Add a script that when the “open car doors” is pressed the animation plays and freezes when its opened and when toggled again the animation plays closing the doors and (close door button to trigger -1 animation speed) add that it plays an sfx sound of a door openingw
5. Add the car engine sfx to the xr orgin and uncheck play on wake
6. Create a script when the turn “engine on” is toggled it plays the audio source and when untoggled it turns off the audio
7. Create an automatic rotating script when the user presses the “rotate car” button it should rotate the car and when let go of button it stops rotating
8. Add interaction manager to several parts of the car
9. Create a script so that when the user interacts with a specific part of the car a ui shows up
10. Create a script so that when the user presses “play audio” the voiceover for that specific part of the vehicle shows up
11. Create a script when the user presses the “color” button a color wheel appears and whatever the color the user selects that color is what the selected car color will be. (using rgb sliders).
12. Create a script so when the user presses “wheel customization” different wheel colors will appear and when the user presses one that is what the wheel color will appear for the selected vehicle.( using sliders again)
13. Make sure that when the vehicle prefab instantiates all vehicle prefabs instantiate with it but are all but one disabled
14. Make a script when a vehicle model is selected in the ui the vehicle already in the scene is disabled and the model that the user chose is enabled.
15. Implement all logic for a 3rd car
16. General script to switch between UIs
17. Move the scene switchers to one script instead of 2

# Final Notes

1. Make a GDD (which will basically be this document or a variation of it)
2. Make a minimum 50 second video demo
3. Everything must look good such as the quality of the vehicles and the UI look
4. Make vehicles real world size
5. Make sure all vehicles have good and clear interior
6. Make a performance report with captures of the following
   1. Capture CPU, GPU and Memory timelines.
   2. Identify top CPU-consuming methods (scripting, physics, rendering).
   3. Track GC allocations per frame and per session.
   4. Draw Call Reduction
      1. Static and dynamic batching for small meshes.
      2. GPU instancing for repeated objects (e.g. wheel rims).
   5. Mesh Optimization
   6. Texture Management
   7. Canvas Splitting
   8. Graphic Raycaster Management
   9. Efficient Update Patterns
   10. Build & Player Settings
       1. Graphics API Selection
       2. Quality Levels
       3. Script Backend (mono vs. IL2CPP)

# Submissions

* GDD
* Video demo
* APK build
* Github Repo link

# Video References

<https://www.youtube.com/watch?v=wGzIQEiu6Tg&ab_channel=heritasharma>

<https://www.youtube.com/watch?v=M05CZN5zu0o&ab_channel=StickyLock>

<https://www.youtube.com/watch?v=5RsWOS5fOqk&ab_channel=VueXR>

# Grading criteria for reference

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Description | Percentage | Marks |
| AR Functionality Implementation | Quality of marker-less and marker-based AR implementation, including placement accuracy and tracking behavior. | **22.5%** | **9** |
| UI & User Interaction | Effectiveness of interactive panels (color, wheels, engine, doors, rotation), layout usability, and user experience. | **20%** | **8** |
| 3D Assets & Visual Quality | Realism and consistency of 3D car models and materials, visual polish, and asset integration. | **15%** | **6** |
| Voiceover & Audio Features | Implementation of voiceover buttons and sound effects (engine, background), clarity and timing of audio. | **10%** | **4** |
| Performance Report (Optimization Areas) | Include a 1–2-page performance report with profiler screenshots and notes on what can be optimized. | **10%** | **4** |
| Submission Quality | Completeness and organization of deliverables: GDD, GitHub repo, APK, and demo video. | **12.5%** | **5** |
| Understanding & Discussion | |  | | --- | | Ability to clearly explain and justify design and technical decisions during discussion. |  |  | | --- | |  | | **10%** | **4** |
|  | **Total** | **100%** | **40** |