



Shared Worlds

DESIGN 6400 - Graduate Design Studio

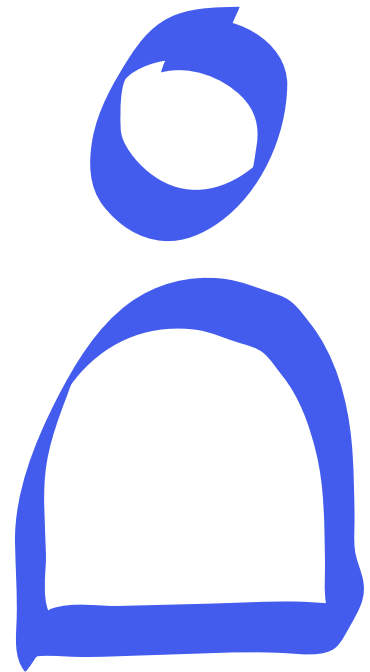
Instructor: Matthew Lewis

Sebastian King

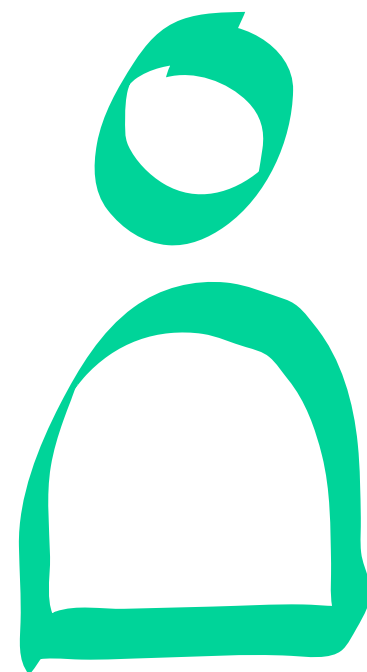
Joshua Antolovic

Alberto Vega

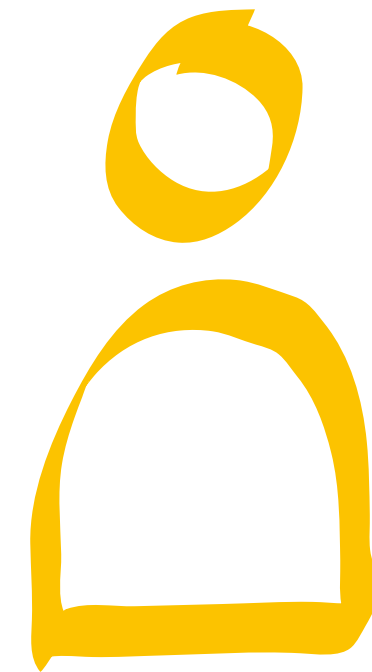
The team



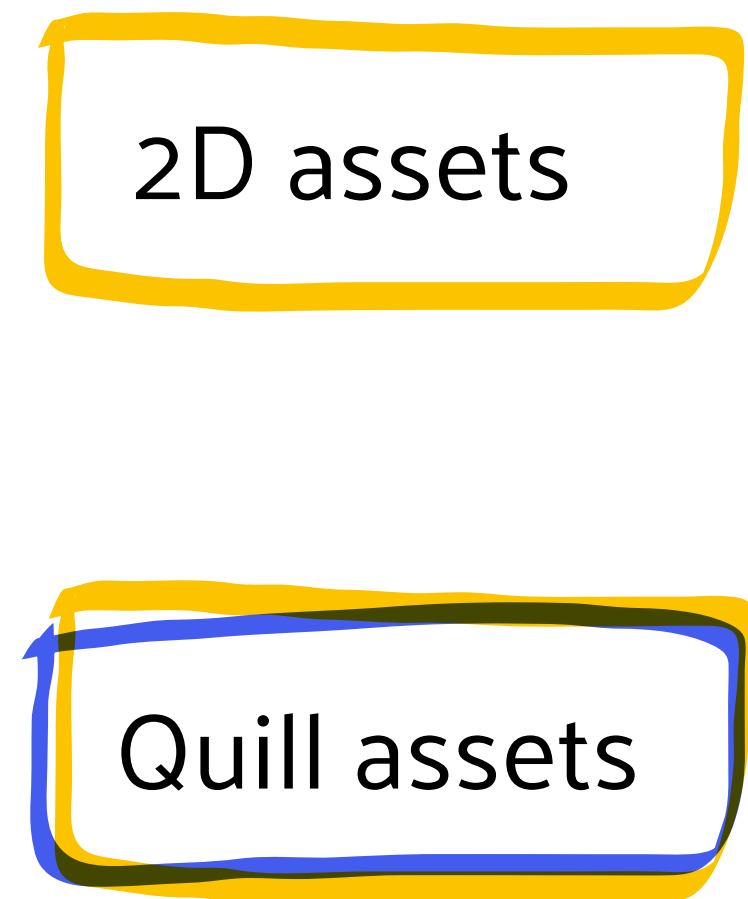
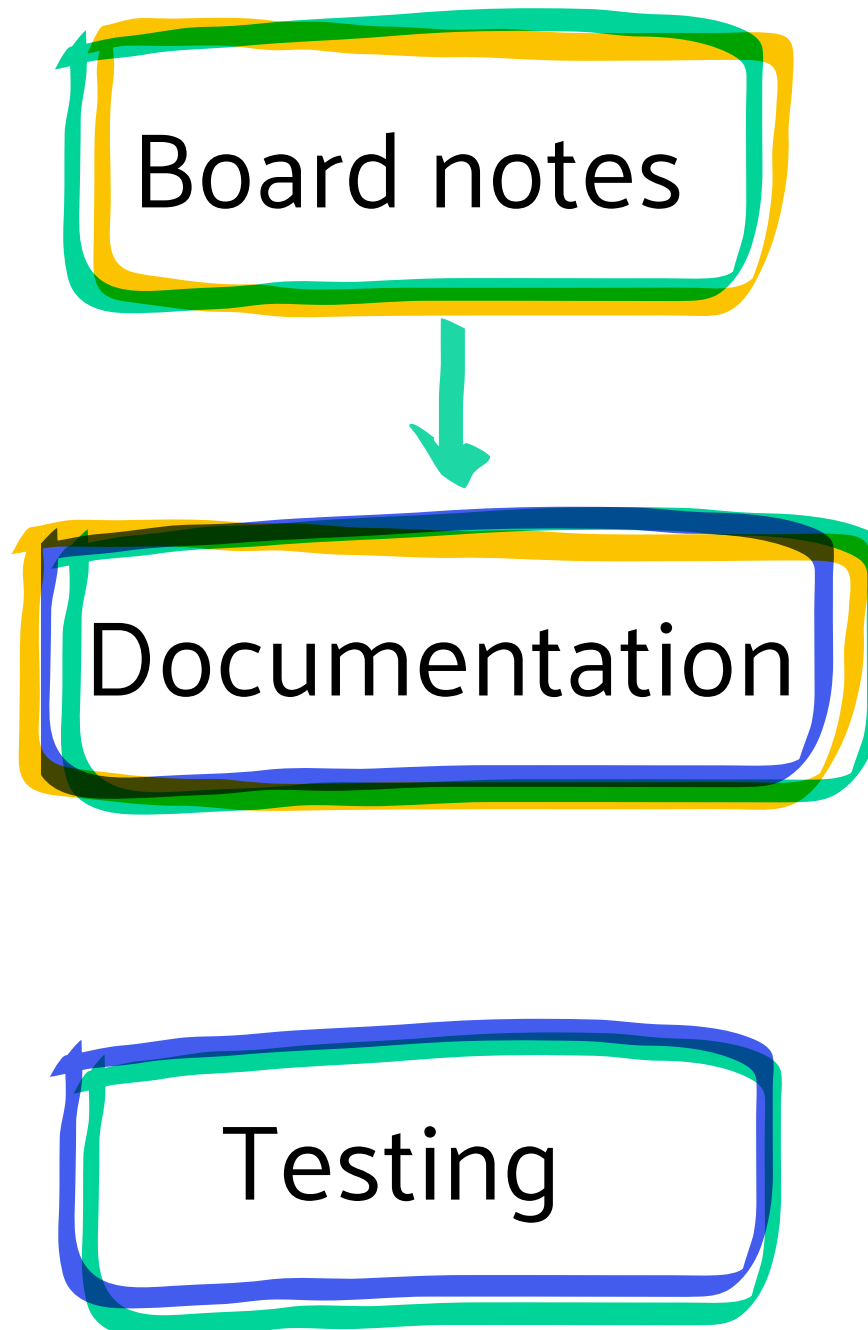
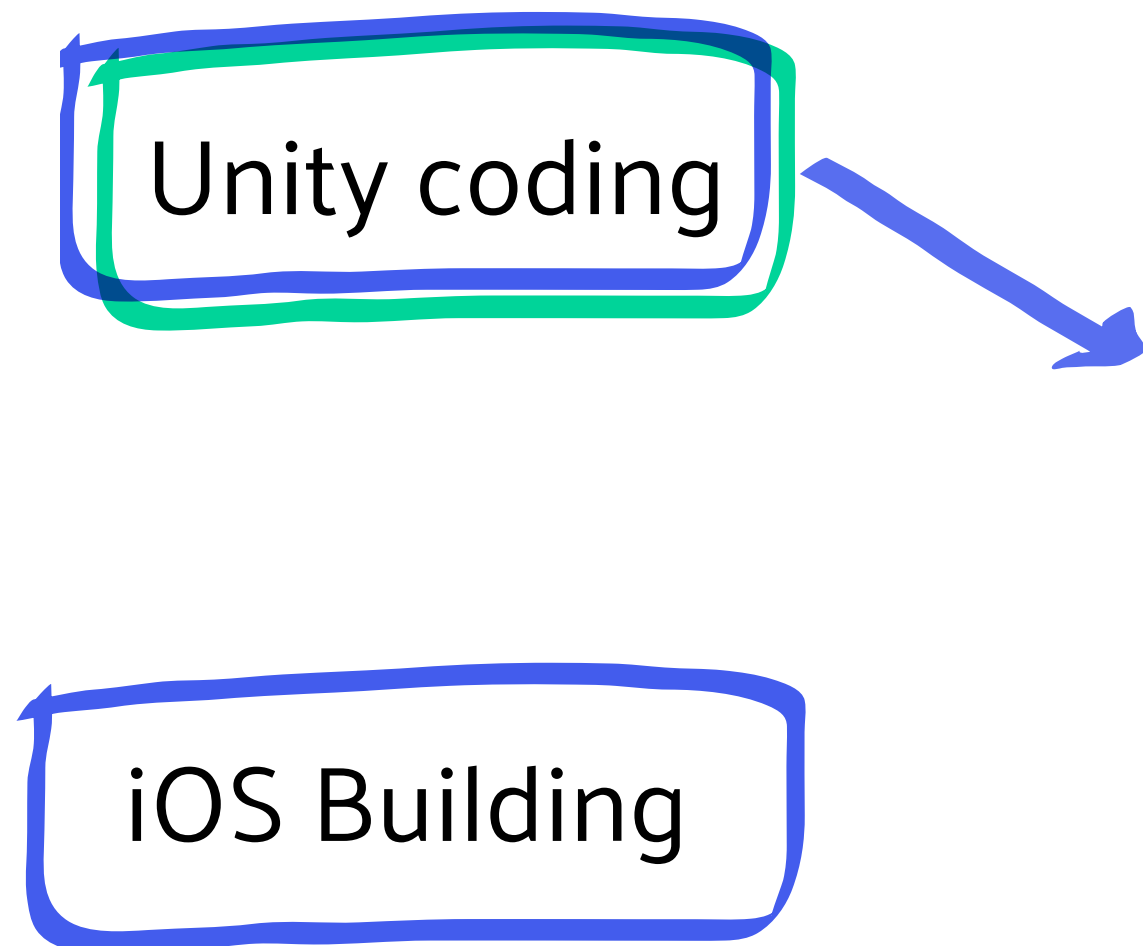
Sebastian



Josh



Beto



Common interests



Ways that scale define AR

Tracking strategies

Anchors
Projectors
Markers
Geospatial

Stationary (Static)
(Desk Buddies)
(Tilt Five)
Mobile (Dynamic)
(Alberto's 2nd project)
(Geospatial stuff)

Abstracting game design engines and frameworks

Asset creation tools

Asset tools
Maya
Blender
Houdini
Gravity Sketch
Tilt Brush
Substance Painter

(Niantic)
Lightship
Occlusion

Real time VR programs
Unity
Aero (less interactivity)
Glitch 3JS? (web)
Unreal

Unity

Unity C# — Unity

C#

Pure C#

Constructor

Unity

Godot

MonoGame

RPG Maker

Face Recognition
Multiplier
Spark (Meta)

AR

(google) maps
ARCore
Apple-ish

Unreal

C++ Blueprints

Androidish

Phone

(apple)
ARKit
Occlusion

Apple

(Unity) AR Foundation
Scale? object recognition

Health

feelings

dual screening (multi)

double stim

hyper reality

alternatives

over/under stimulation

feel?

visual? now?

Slider

Pass through

AR

MR (quest 3 & pro)

Magic Leap & HoloLens

Tilt Five

Connecting and understanding AR frameworks

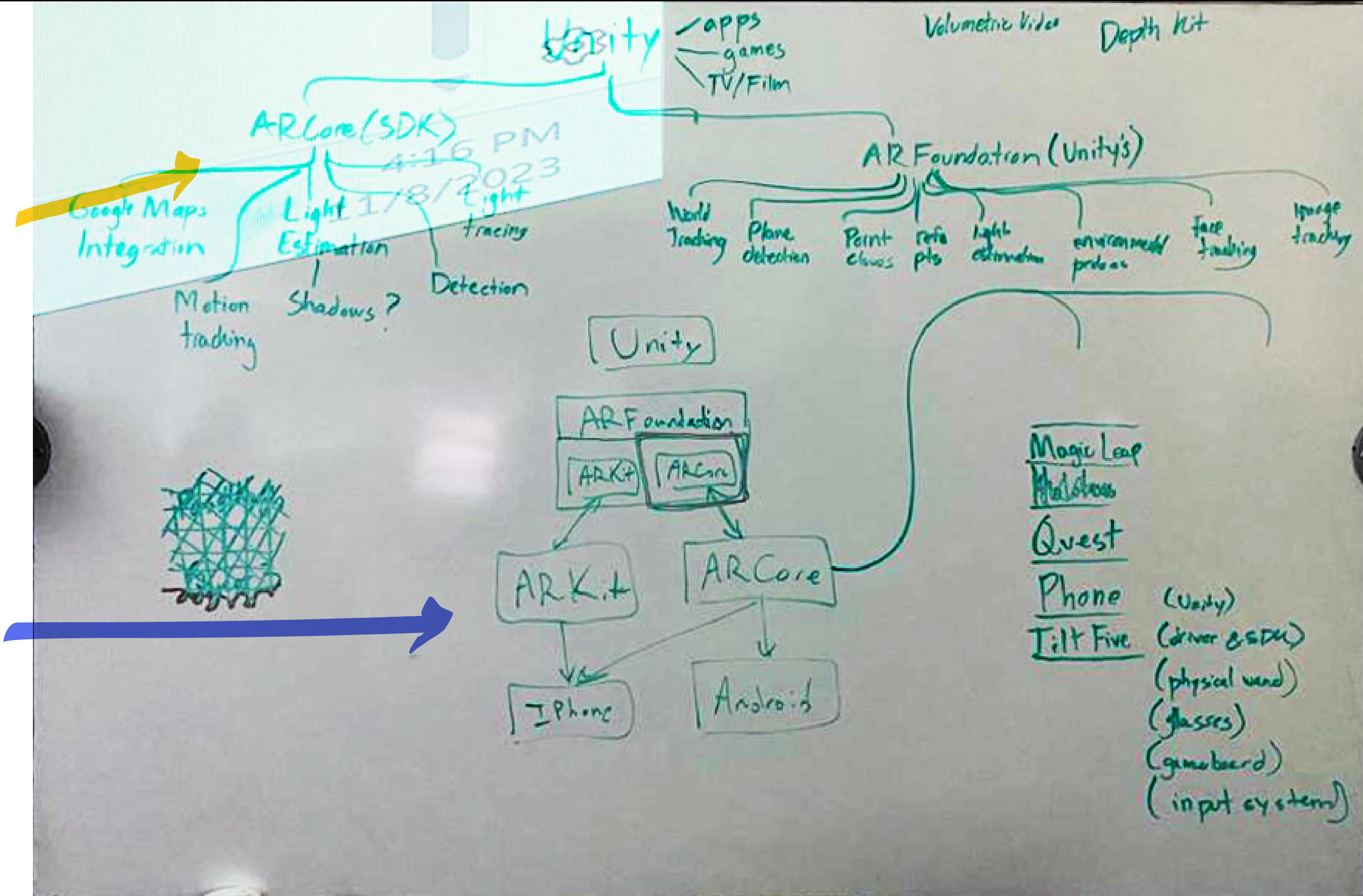
Concepts about health and stimulation as it relates to technology

Types of AR devices

We decided to focus on understanding how **Unity** relates to the different **AR frameworks** and their capabilities.

We started outlining the different features of **ARCore** and **AR Foundation**.

It wasn't immediately clear how these features were connected, so we drew a second diagram trying to understand how **Unity** connected with the **AR SDKs** we were finding



Key Takeaways

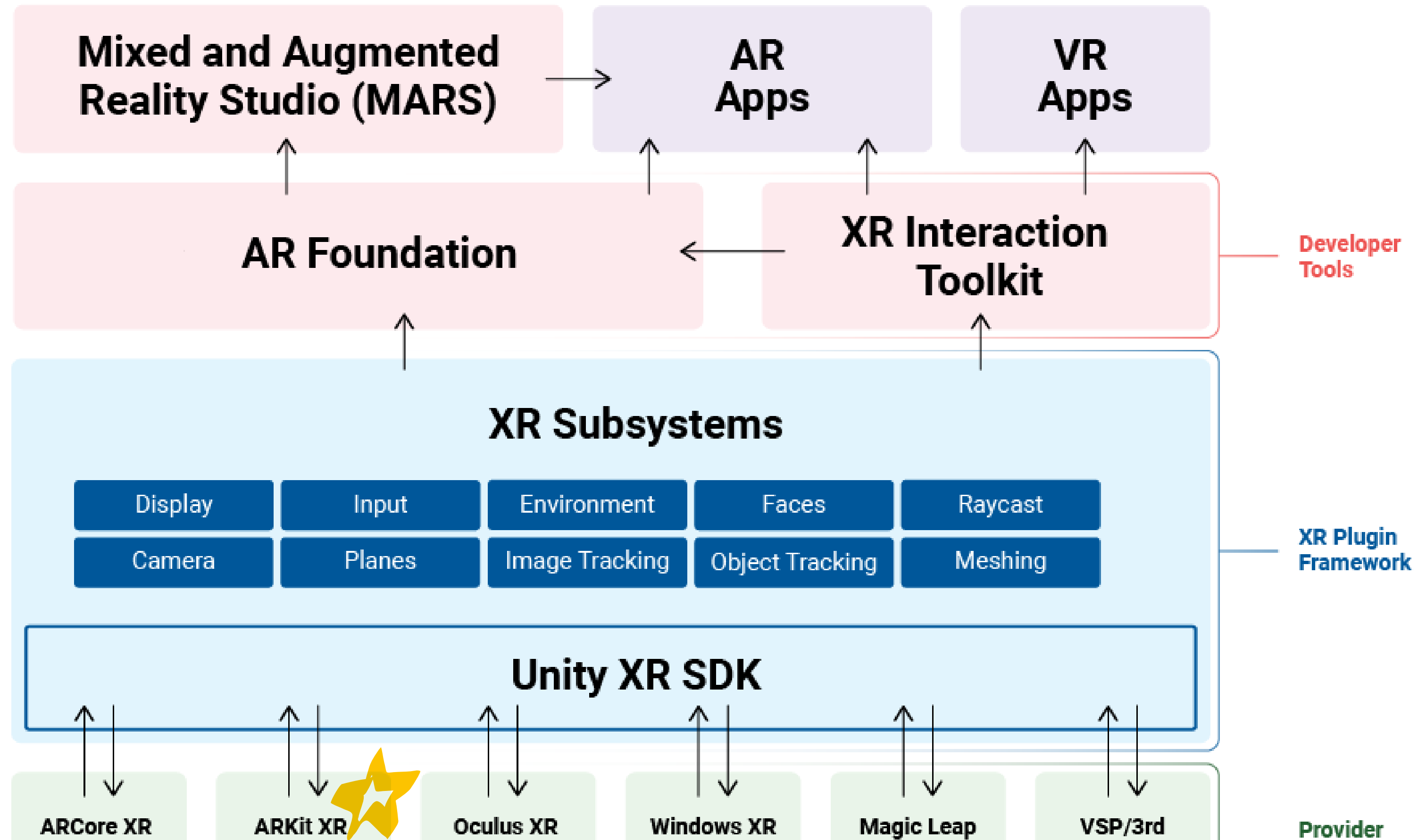
AR Foundation is a package that acts as a layer between Unity and several AR SDKs so that they can communicate

AR Foundation has support for ARKit and ARCore, as well as a list of devices

Different AR SDKs have their own intended devices and limitations

XR system diagram

Source: Unity Documentation / Manual



- Officially supported platforms
- Verified Solutions Partners
- Innovators

Schedule (?)

O44

fx

	A	B	C	D	E	F	G	
1			Start Task	In Progress	Finished			
2								
3			Week 1: 11/6 - 11/10	Week 2: 11/13 - 11/17	Week 3: 11/20 - 11/21	Week 4: 11/27 - 12/1	Week 5 - 12/4 (Presentation)	
4	Documentation		Reference sheets					
5						Presentation Slides		
6			Research*					
7								
8	Organization/Design	Figma Diagrams		Figma Diagrams				
9				User Journey (part of diagrams)				
10				other diagrams here				
11								
12								
13	Classic Design				Tile Fabrication			
14					Digital asset creation			
15								
16								
17								
18								
19								
20								
21								
22								
23								
24	Technical stuff	Scripting Tasks		Learning AR Foundation/Unity Editor				
25				Lightship				
26				Phones into dev mode				
27				Build Prototyping (loading applications onto phone stuff)				
28				Image Tracking				
29				Multiple Image tracking				
30				Game object management				
31						Linking multiple gameobjects to one image		
32						UX/UI		
33						Multiplayer?		
34								
35								
36								

What would our AR experience look like?

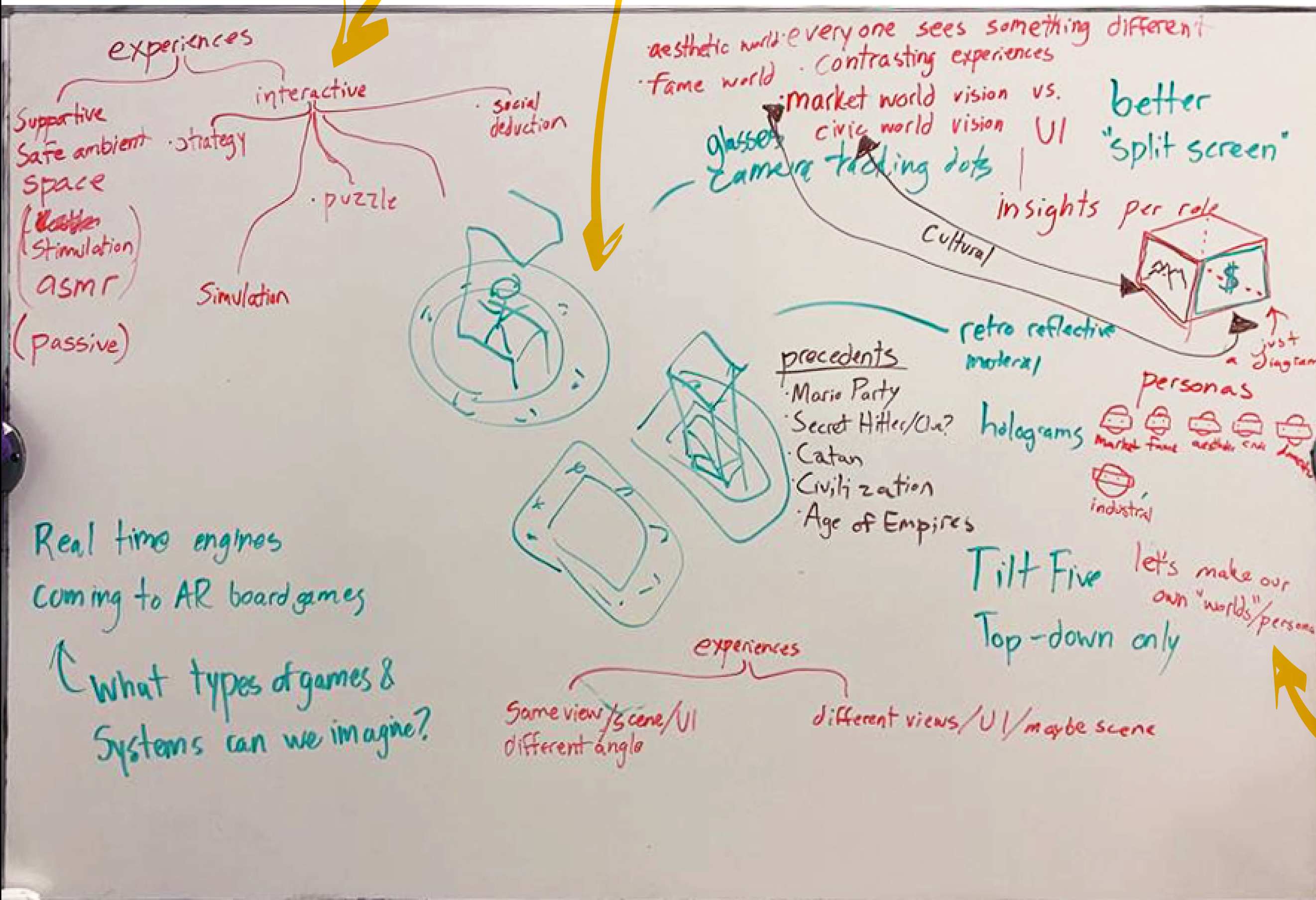
brainstorming

Themes and game strategies we wanted to explore.

Game spaces can be defined with different AR placemats (though this wasn't something we would be able to do with the Tilt Five).

We wanted to combine ideas of spatial AR representation, territory, and perspectives

- Could we use the toolkit laid out by Sébastien Proulx and assign roles to players? How would we differentiate what the **different roles** see?
- Would the space be **static** or a **simulation** like builder-type games?
- We came up with an idea about **unique perspectives** being shown through **different AR glasses**, as well as potentially being determined by position around the play space. Each player would see the 'world' differently.



Technical Medium Considerations



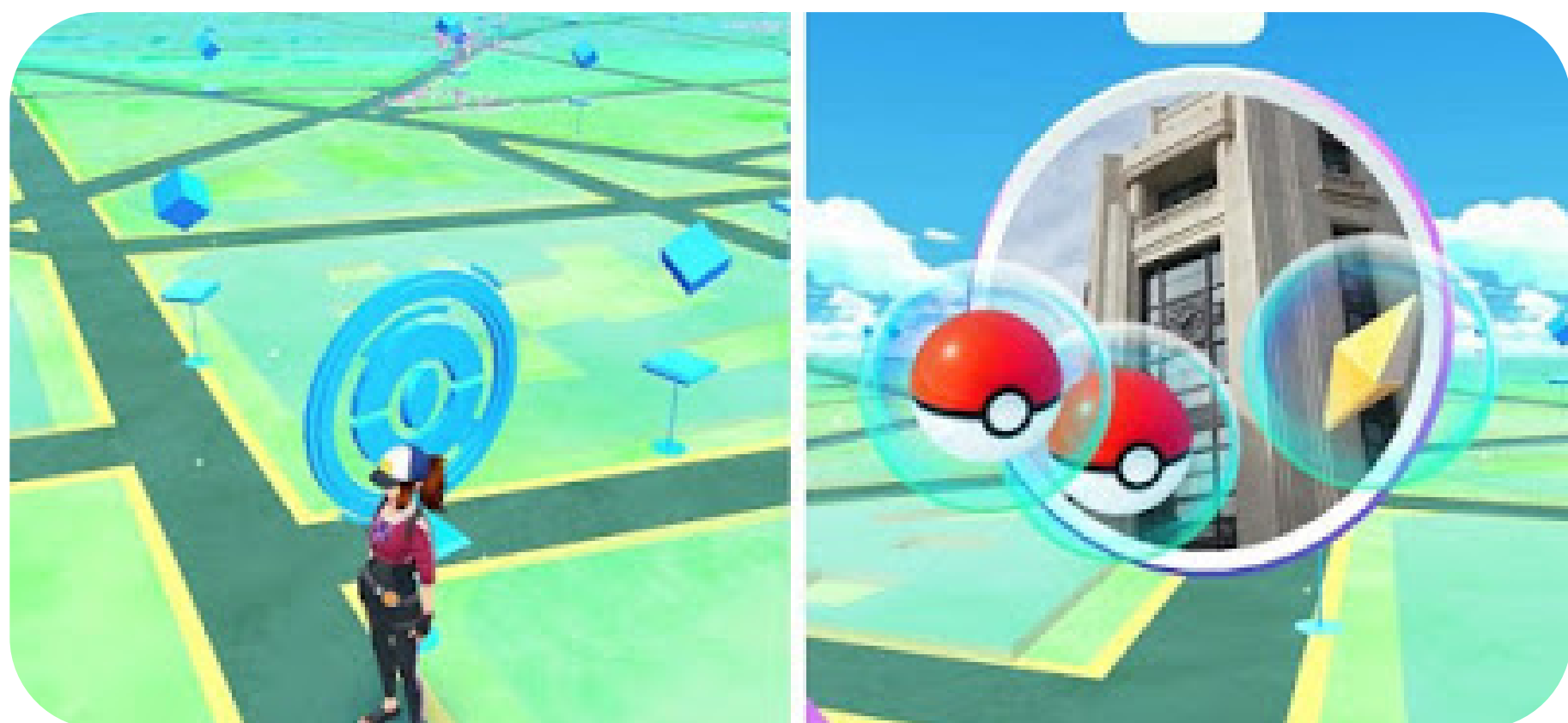
Tilt Five?

We would be restricted to a stationary space and only one pair of glasses but maybe there could be some perspective tricks or asynchronous gameplay?

ARCore or ARKit?

Instead of a defined AR space maybe we make a phone app where users can build worlds?

Maybe image tracking could be used?



Niantic Lightship?

What if there was some sort of geospatial aspect to the game? What resources are available for making multiplayer?

First ideas

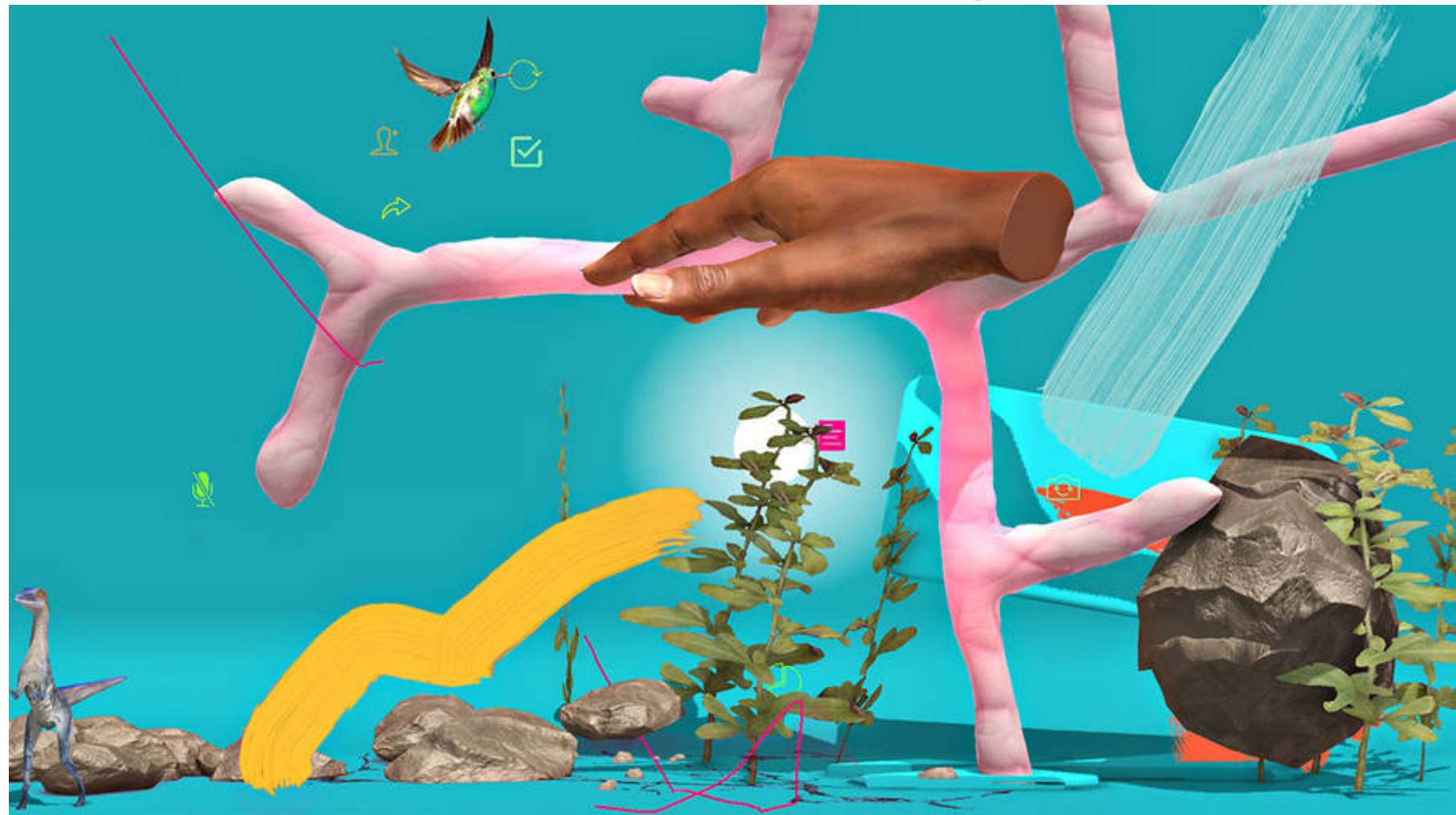
- Use of **tiles** to build out spaces
- **Individuals** get their own spaces to build, would there also be a **communal** space (and how would that work)?
- Will we use any sort of **asynchronous gameplay** (i.e. players with different roles and different gameplay loops)?
- How game-like do we want the experience to be? More ambient?
- Still deciding between Quest and Tilt 5, we would need to test both

exquisite corpse



Jake and Dinos Chapman

Kit bashing



Theo Triantafyllidis

tile could be different assets

- exquisite corpse collective world

IDEAL

- markers (tiles (squares)) (multiple) physical & digital map
- Marker recognition w/ AR Foundation table-based

Unity primitives, cylinder, cone, cube... building blocks

Backup

if multiple markers does not work, then... tilt five &/or Quest 2 to place/model digital assets. → then room-based.

① Multiple markers → Unity AR Foundation Seb.

② Five / Five m. diagrams new definition link

texture / tile packs (Kitbashing-ish)

different prop

①	C		
		S	②
		J	A

individual individual individual individual

Asyn gameplay

Separation of roles rules

Solo or teams

if you want

lawless

both okay

table scale

- ambient mode
- resources mode

like waste garbage tiles? what do we do with all this SHIT

Learning route

Balance: Ideas / time / technical requirements

What kind of prototype can we build?

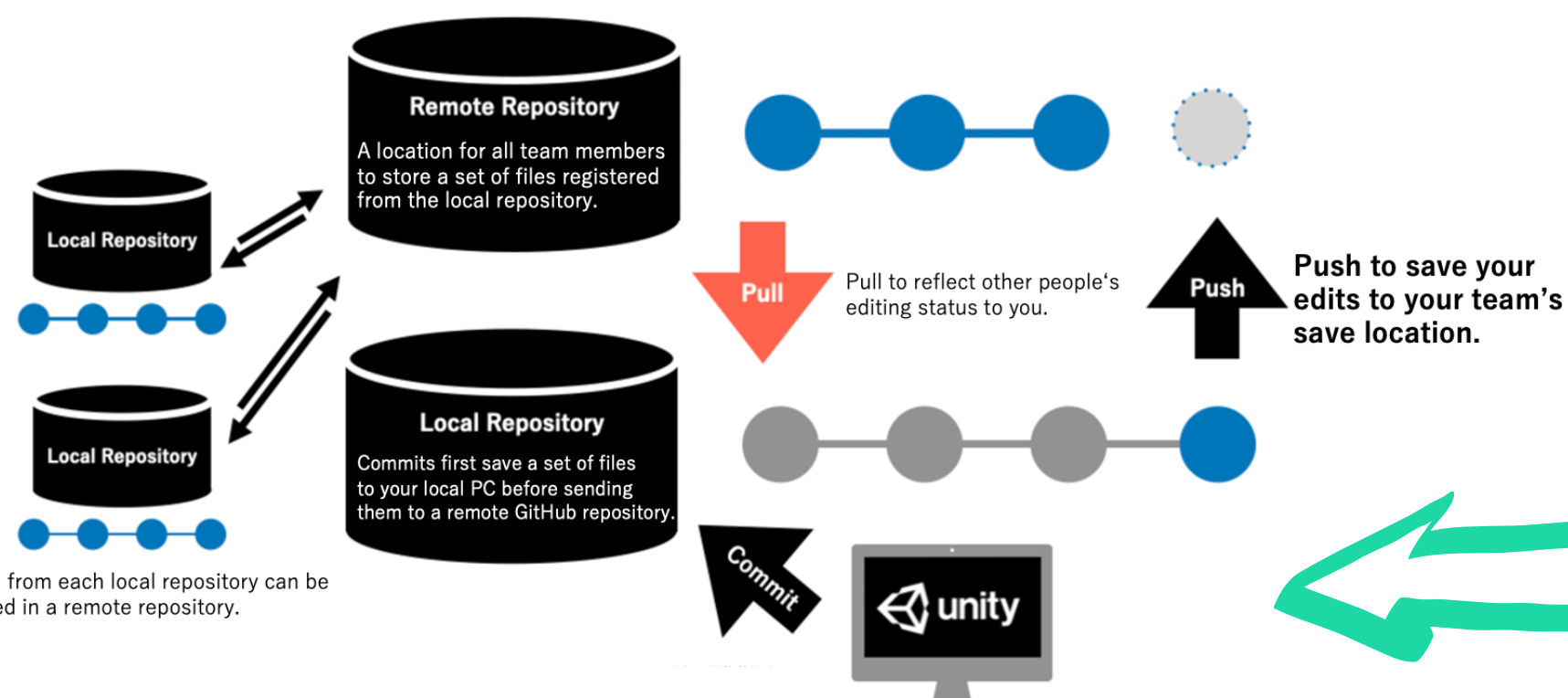
What we need to learn or consider to built it?

Very MOSCOW method: What is essential and what could we do without/fake?

Scheduling the team work

Repository workflow

Using Github to do version control/team development for our Unity project



Technical Challenges

① Quest Multiplayer?

local

↳ Prototype: Solo-version on multiple Quest 2's

② Image Tracking w/ multiple markers?

↳ Meta XR?

Limitations:

- Number of markers?

- Marker size?

③ Working w/ Passthrough ✓

↳ Passthrough API for Quest

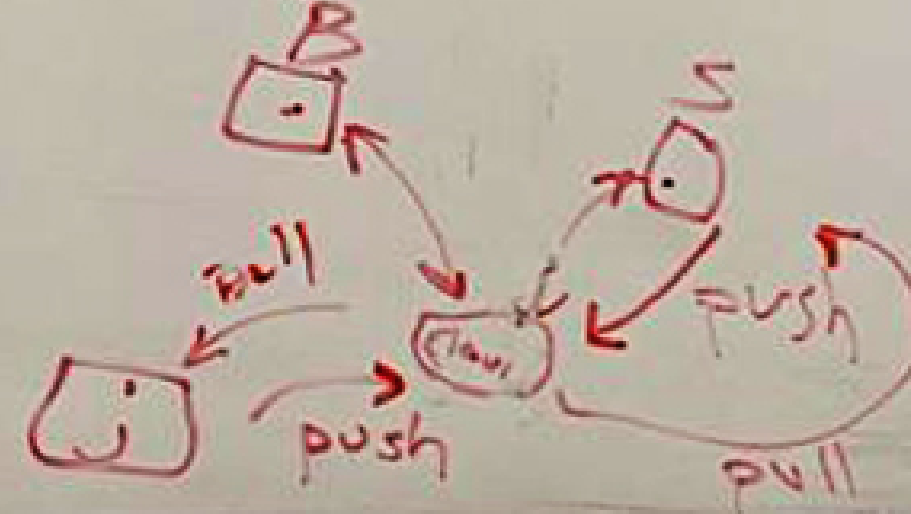
Integration of Passthrough & Image Tracking

④ Learning how Quest works w/ AR Foundation ✓

Meta XR SDK
Is OpenXR plugin easy to use?

Most important!

V15
V2
V3
V4



Testing providers

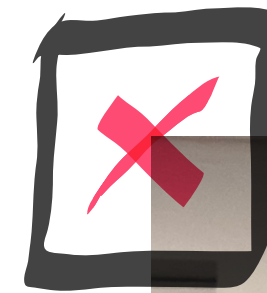
Meta Quest 2/3

- We hit a wall once we realized Meta doesn't allow camera feed for development processing. We were unable to track images via passthrough.



Tilt 5

- The technical side was fine, but we would need to install some proprietary software and we didn't think the tech was what we wanted to use/needed for this idea



Josh
Jumpscare

Building in Mobile

We decided to pivot and look into mobile development for our AR experience

We would use **AR Foundation** and make an iOS build

We also ended up making a new schedule with a set of tasks that we wanted to get done, including:

- Figma diagrams
- Hand-drawing tiles
- Hand-drawing 3D assets in Quill
- Debugging Unity.. :(

Figma

User journey diagram (B)

- ★ - Player Objectives
- Rules
- Moments / Procedure / Steps / Operations (timeline)
- ★★★ - Persona, codesign or for fun

Finished Version Diagram (Maximum Finished product)

- Multiplayer (J&S&B)
- Geo-based - ambiguous shapes & models description
- Project timeline (?) (Diagram or Slide)
 - Our process (success & Failure)

System Diagram (J&S)

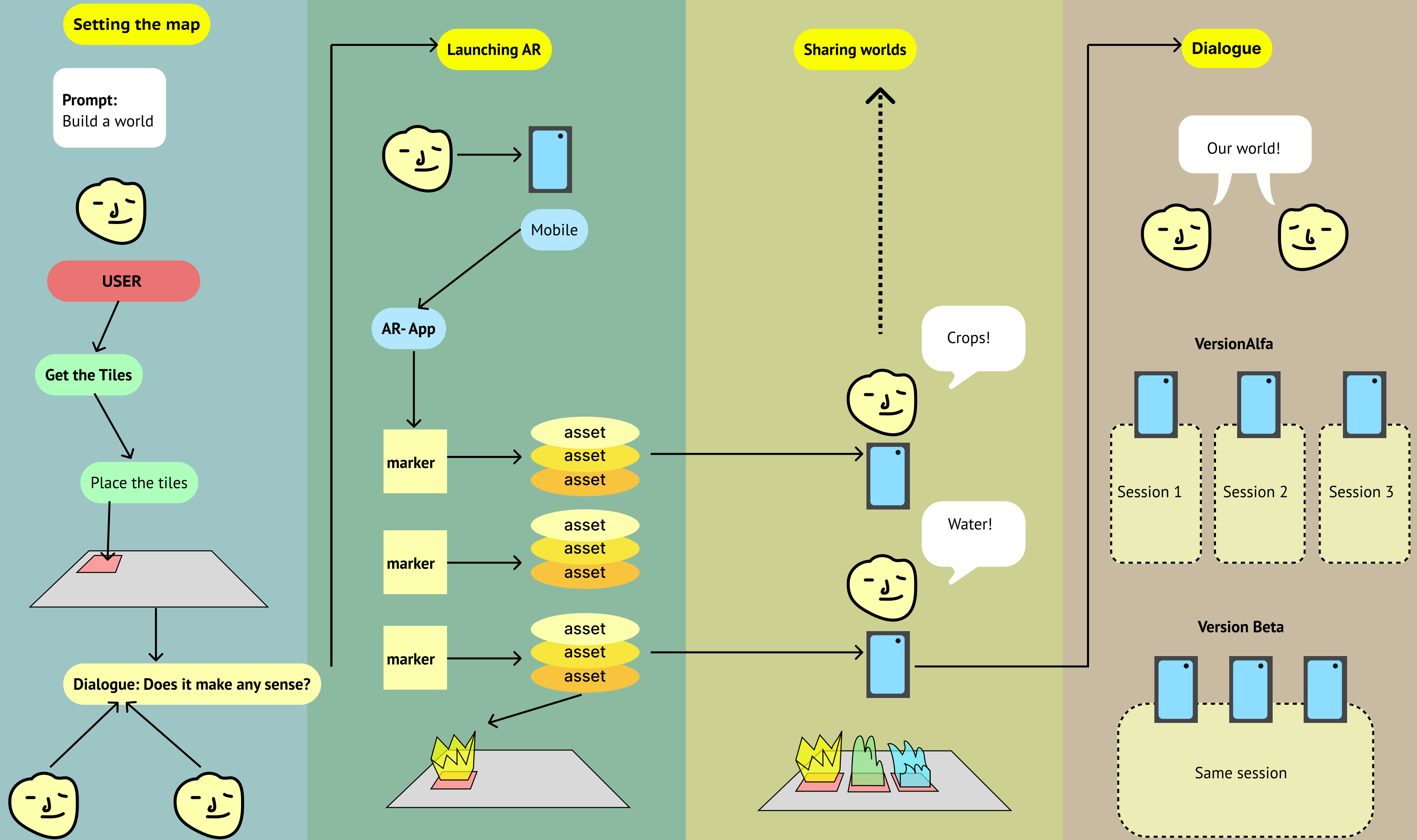
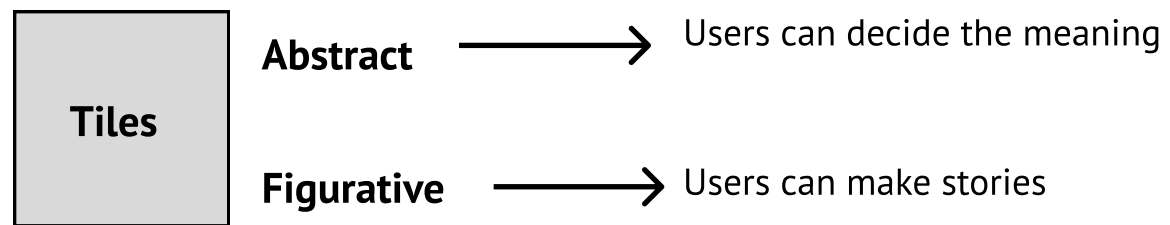
- technical



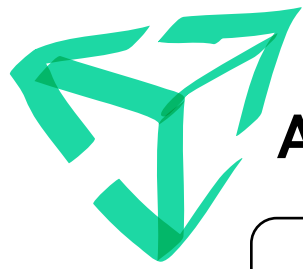
- MVP, 12/4
- no multiplayer
- 1 scene
- 3 images (tiles)
- 1 asset per image
- future versions, (more assets per image)

- 3 images → Hand drawn Photoshop Assets (Anybody)
- 9 3D models → Quill (J&S)
- Documentation videos (B)
- Unity (J&S)
- Diagrams (B)
- XCode (J&S)
- Moral support (Small Fry & Nicby)

User's journey



System diagram



Application

Open app, load core components of Unity game engine, AR Foundation

Feature check / device compatibility

AR Foundation initializes AR Session

Initialize camera

Process camera feed for detection of images in library

Retrieve 3D model and render using Unity

Calculates position, scale, orientation

Capture and process inputs

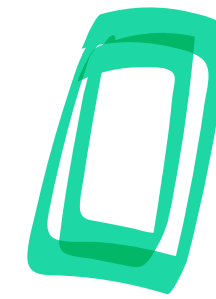
Unity manages memory and resources

Upon exit, shuts down AR Session and other processes

Unity and AR Foundation perform cleanup operations

Unity and AR Foundation perform cleanup operations

App closes and Unity ensures all processes have terminated



iPhone

app triggers a call to the OS to display permission dialogue

Configures sensors for AR tracking and positioning

Stream video data for processing

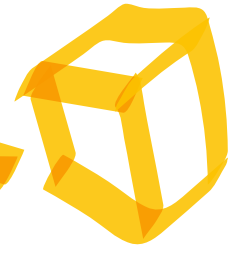
Run computer vision algorithms particular to each OS

Image recognition

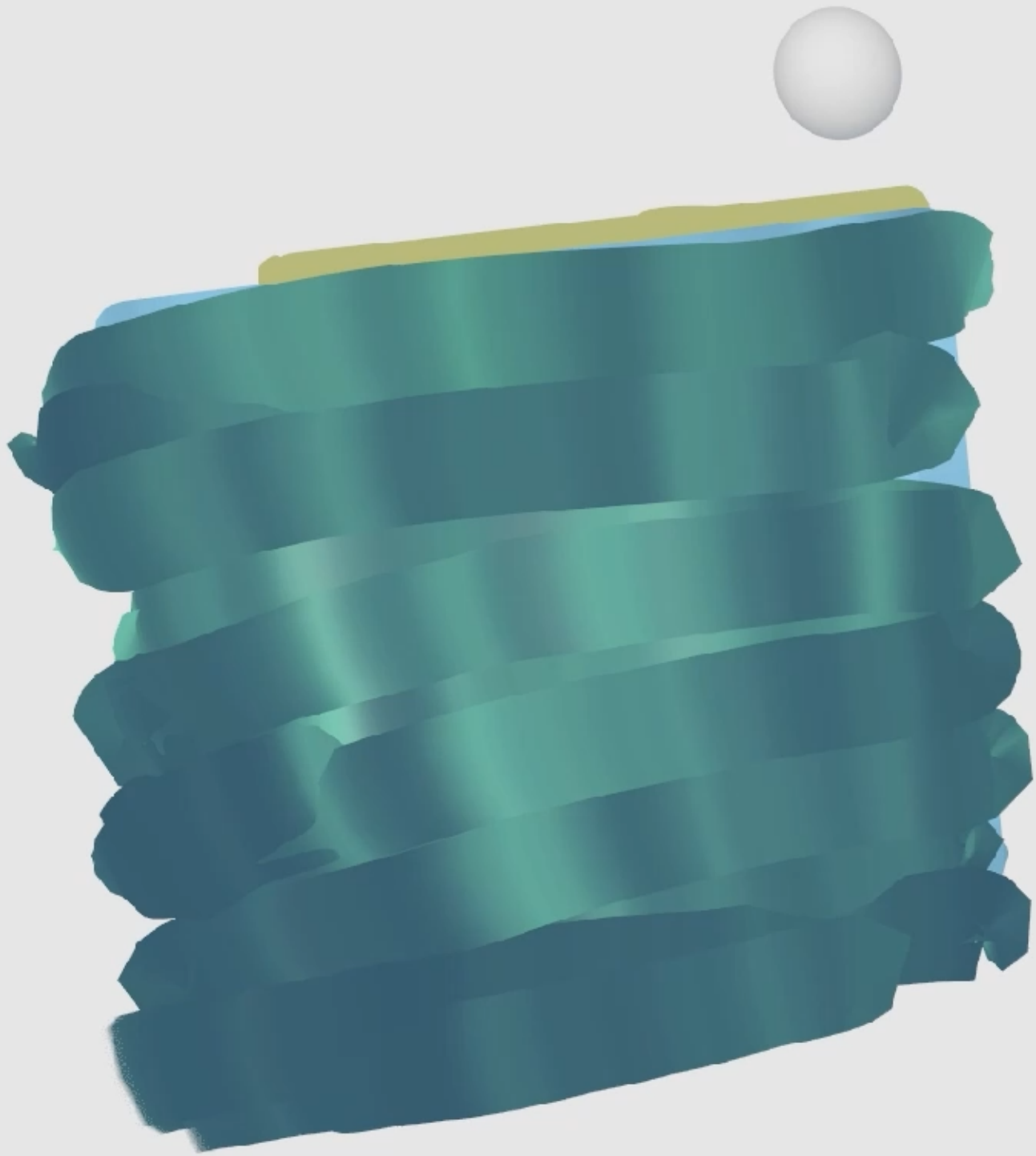
Overlays the 3D model into the camera feed

Update the overlay per inputs

Quill props



3



Testing Mobile Development



Unity-iPhone Unity-iPhone > Marllon's iPad Build Failed | Today at 14:24

Unity-iPhone

General Signing & Capabilities Resource Tags Info **Build Settings** Build Phases Build Rules

PROJECT

- Unity-iPhone

TARGETS

- Unity-iPhone
- Unity-iPhone Tests
- UnityFramework**
- GameAssembly

Linking - General

Setting

Other Linker Flags

Debug

Release

ReleaseForProfiling

ReleaseForRunning

Quote Linker Arguments

UnityFramework

<Multiple values>

-ObjC -l"libc++" -l"salite3" -l"..." -framework "CoreTele...

\$(inherited)

-weak_framework

CoreMotion

-weak-System

\$(OTHER_LDFLAGS_FRAMEWORK)

\$CONFIGURATION_BUILD_DIR/il2cpp.a

-ObjC

-ld64

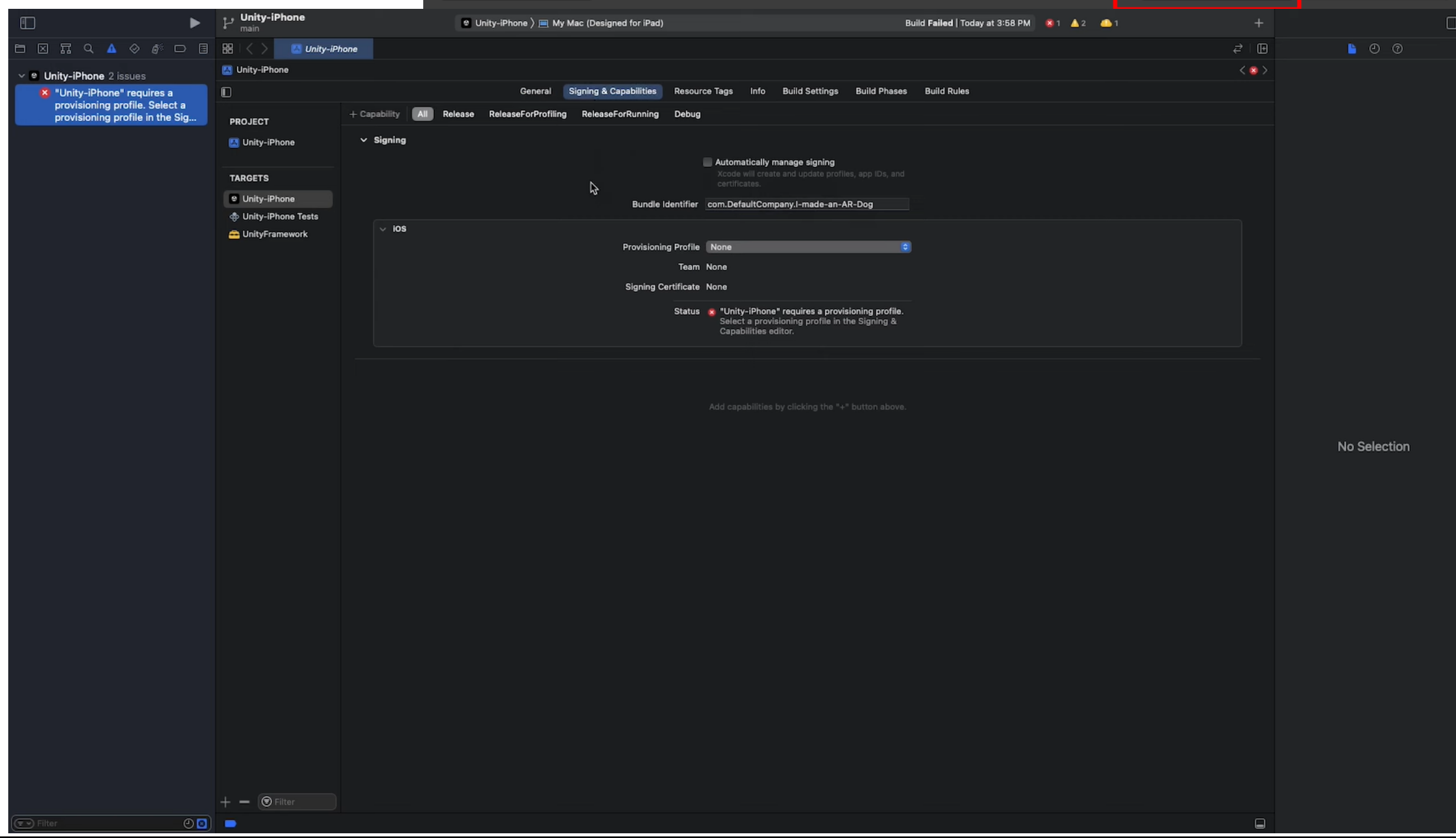
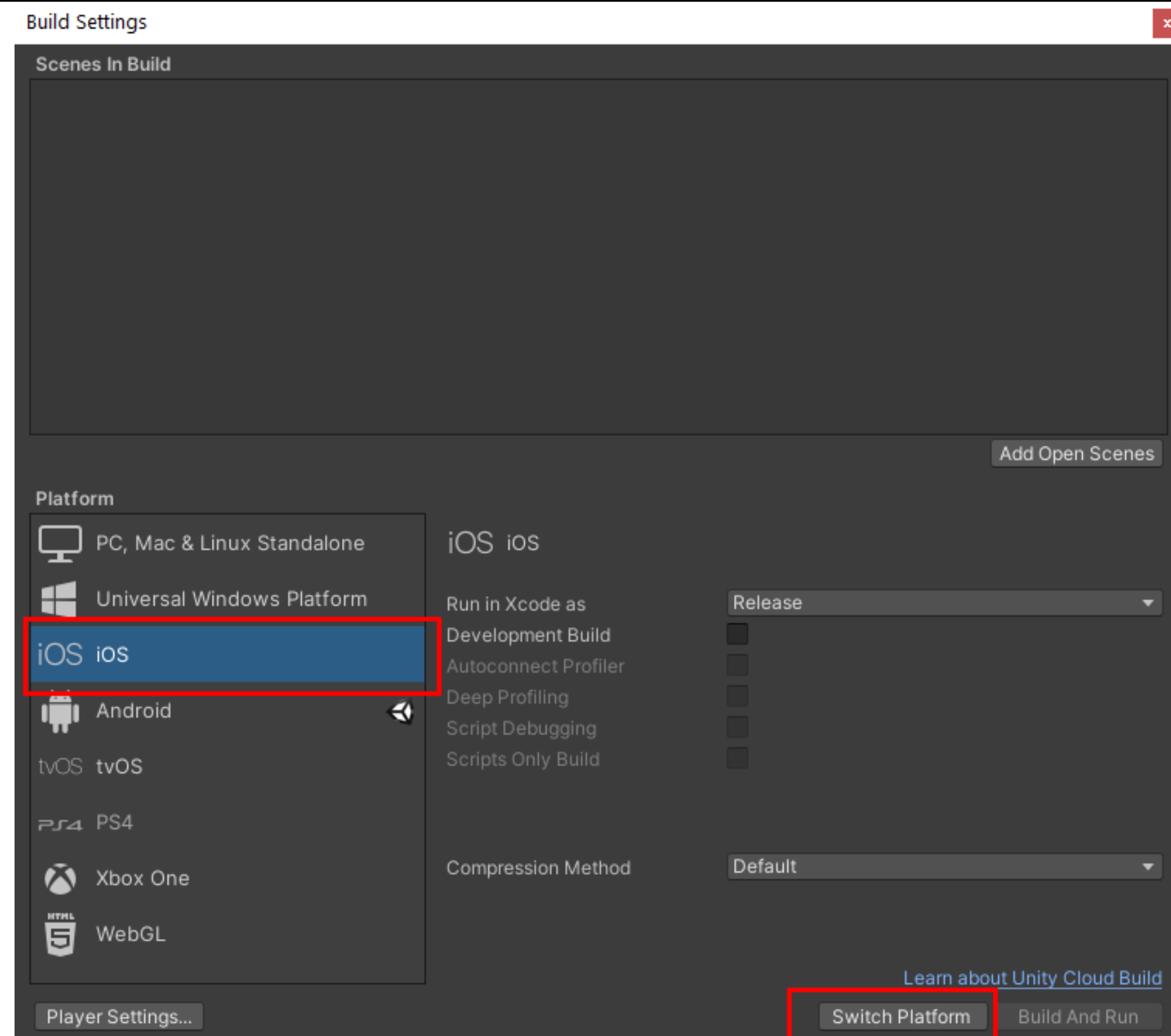
```
Showing Recent Issues
Unity.II2CPP.Bee.BuildLogic.ToolchainNotFoundException: IL2CPP C++ code builder is unable to build C++ code. In order to build C++ code for Mac, you must have Xcode installed.
Building for Apple Silicon requires Xcode 9.4 and Mac 10.12 SDK.
Xcode needs to be installed in the /Applications directory and have a name matching Xcode*.app. Or be selected using xcode-select.
It's also possible to use /Library/Developer/CommandLineTools if those match the listed requirements.
More information and installation instructions can be found here:
https://developer.apple.com/support/xcode/
Specific Xcode versions can be downloaded here:
https://developer.apple.com/download/more/
Unable to detect any compatible iPhoneOS SDK!
at Unity.II2CPP.Bee.BuildLogic.iOS.iOSBuildLogic.GetCompatibleXcodeInstallation(Architecture architecture, Version xcodeMinVersion, Version platformSdkMinVersion, Identifier platformSdkIdentifier, XcodePlatformSdk& compatiblePlatformSdk, XcodeInstallation& compatibleXcodeInstallation)
at Unity.II2CPP.Bee.BuildLogic.iOS.iOSBuildLogic.UserAvailableToolchainFor(Architecture architecture, NPath toolChainPath, NPath sysRootPath, Boolean targetIsSimulator)
at Unity.II2CPP.Bee.II2CPPExeCompileCppBuildProgram.BuildProgram.Main(String[] args, String currentDirectory)
at Unity.II2CPP.Building.InProcessBuildProgram.StartImpl(String workingDirectory, String[] arguments) in /Users/bokken/build/output/unity/il2cpp/Unity.II2CPP.Building/InProcessBuildProgram.cs:line 51
Error: Unity.II2CPP.Building.BuilderFailedException: Build failed with 0 successful nodes and 0 failed ones
Error: Internal build system error. BuildProgram exited with code 1.
Unity.II2CPP.Bee.BuildLogic.ToolchainNotFoundException: IL2CPP C++ code builder is unable to build C++ code. In order to build C++ code for Mac, you must have Xcode installed.
Building for Apple Silicon requires Xcode 9.4 and Mac 10.12 SDK.
Xcode needs to be installed in the /Applications directory and have a name matching Xcode*.app. Or be selected using xcode-select.
It's also possible to use /Library/Developer/CommandLineTools if those match the listed requirements.
More information and installation instructions can be found here:
https://developer.apple.com/support/xcode/
Specific Xcode versions can be downloaded here:
https://developer.apple.com/download/more/
Unable to detect any compatible iPhoneOS SDK!
at Unity.II2CPP.Bee.BuildLogic.iOS.iOSBuildLogic.GetCompatibleXcodeInstallation(Architecture architecture, Version xcodeMinVersion, Version platformSdkMinVersion, Identifier platformSdkIdentifier, XcodePlatformSdk& compatiblePlatformSdk, XcodeInstallation& compatibleXcodeInstallation)
at Unity.II2CPP.Bee.BuildLogic.iOS.iOSBuildLogic.UserAvailableToolchainFor(Architecture architecture, NPath toolChainPath, NPath sysRootPath, Boolean targetIsSimulator)
at Unity.II2CPP.Bee.II2CPPExeCompileCppBuildProgram.BuildProgram.Main(String[] args, String currentDirectory)
at Unity.II2CPP.Building.InProcessBuildProgram.StartImpl(String workingDirectory, String[] arguments) in /Users/bokken/build/output/unity/il2cpp/Unity.II2CPP.Building/InProcessBuildProgram.cs:line 51
at il2cpp.Program.DoRun(TinyProfiler2 tinyProfiler, String[] args, RuntimePlatform platform, Il2CppCommandLineArguments il2CppCommandLineArguments, BuildingOptions buildingOptions, Boolean throwExceptions) in /Users/bokken/build/output/unity/il2cpp/il2cpp/Program.cs:line 319
```


Building iOS- XCode



In order to build to an iPhone, we needed to:

- Switch platforms in our Unity project build settings and make a build on a Mac
- **Open** the generated project in Xcode
- Put an **iPhone** into developer mode
- Register our **apple account** as a developer account (the free version)
- Build the project to the **connected phone** via Xcode



Iterative Unity Development

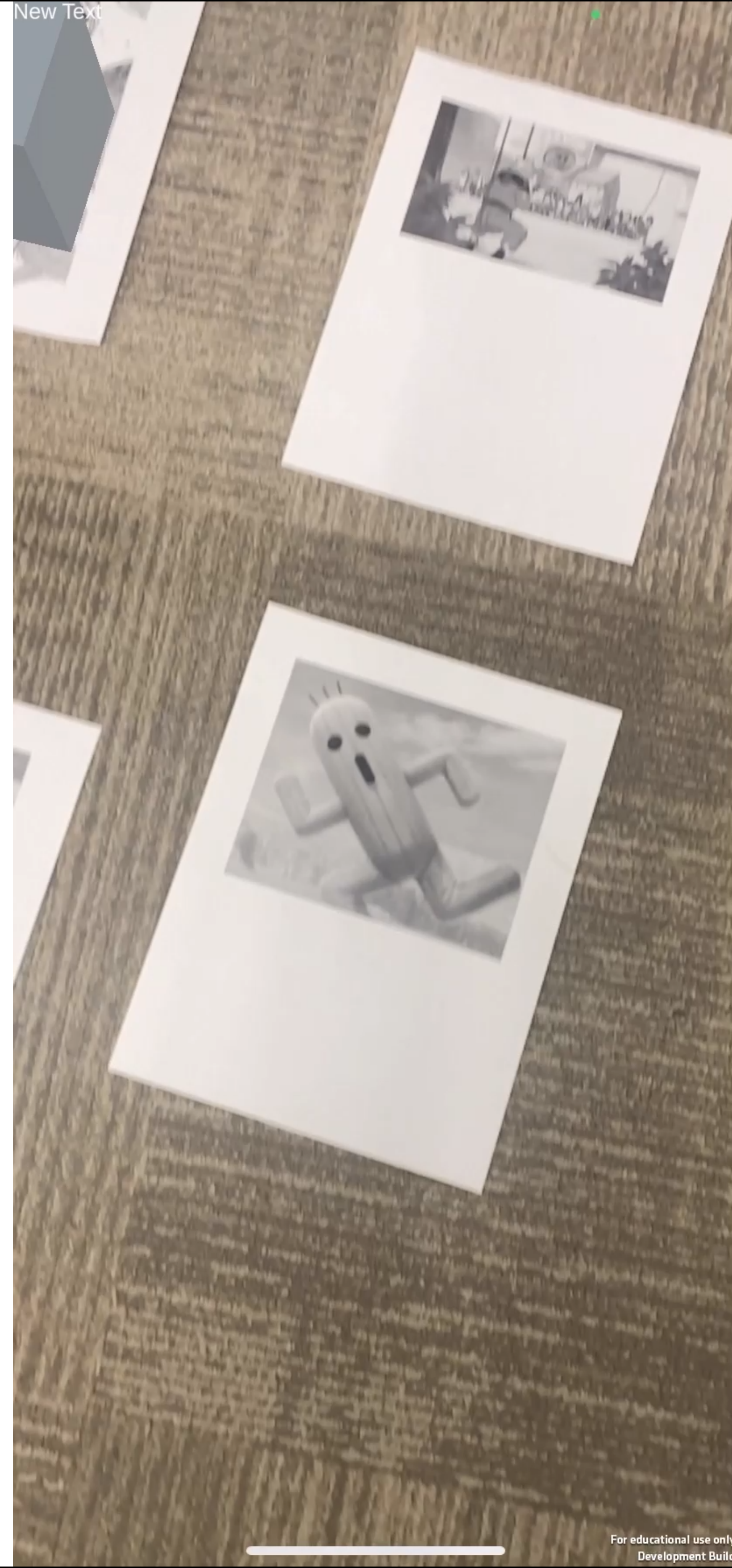
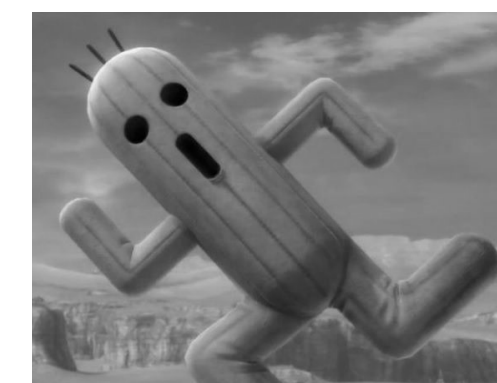
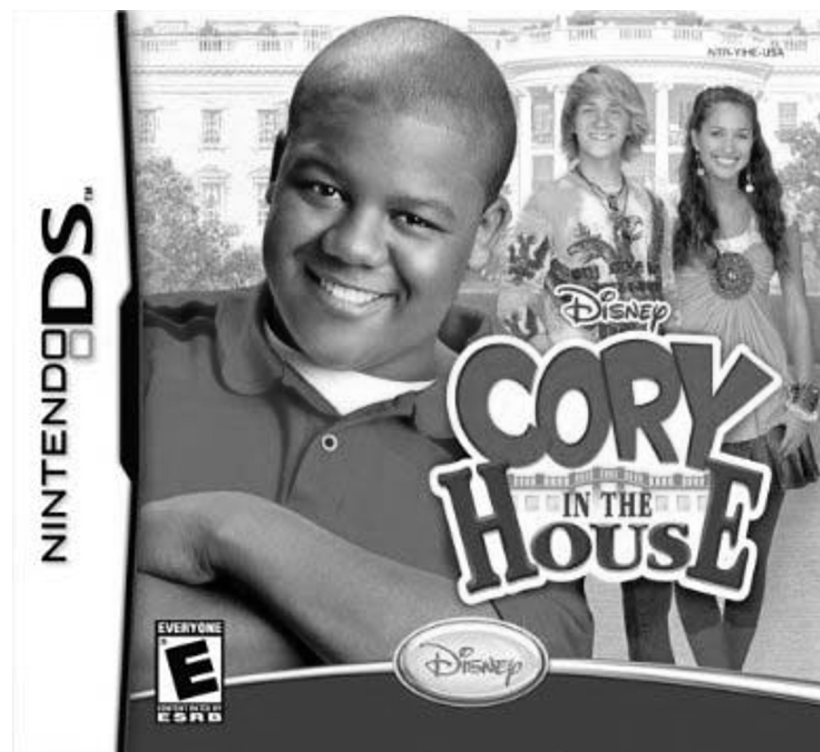
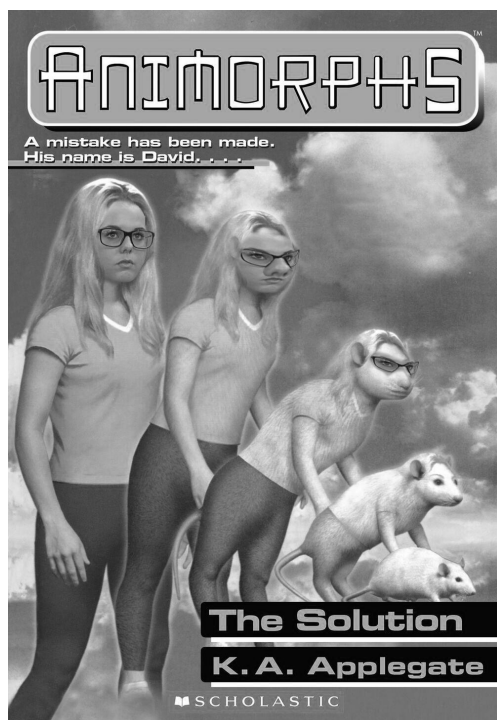
Things we did in order of testing in Unity for functionality:

1. Could we get an AR project onto the phone that **could access the camera**?
2. Could we get an AR project onto the phone **with camera access and something in the space**?
3. Could we get **image tracking to work with a single image**?
 - a. **How big do the images** have to be? **How much of the screen** do they have to take up before an object gets placed?
 - b. What happens **if we have multiple of the same image**, will there be **multiple objects placed**? (No, it just picks the first one it sees)
4. Could we get image tracking to work with multiple images, **but just one model**?
 - a. What images does the app **have trouble with** and which ones **does it easily recognize**?
5. Could we place **different items on different images**?
 - a. Could we get **the items to stay in the space if the image isn't being tracked**? (Yes, surprisingly easily too)
6. Could we **map lists of objects to images**, and could we get the current object to change by **tapping on it**?
 - a. We ended up switching to the new input system for this

Prototype Alpha Version



Our initial test helped us realize what type of imagery the image processing algorithms could detect. Certain features and high contrast helped the process detect the image faster.



Prototype Gold Version

LIVE DEMO

