

Project Deliverable K

November 28th, 2020

Nassar Shakir (300082867)

Adam Jahan (300189300)

Adam Walters (300109768)

Emeric Chanhoun (300211395)

Abdullah Nauman (300136237)

Table of Contents

Table of Contents	2
Introduction	3
Customer Needs	3
Metrics & Target Specifications	3
Metrics	3
Target Specifications	4
Conceptual Designs	4
Interface View	4
App Layout	6
Prototyping, Testing, Customer Feedback	6
Prototype 1	6
Testing Plan	6
Documentation	7
Feedback	9
Prototype 2	9
Testing Plan	9
Documentation	10
Feedback	11
Prototype 3	12
Testing Plan	12
Documentation	12
Feedback	14
Final Solution	14
Instructions	17
App:	17
Navigating AR Interface:	17
Conclusion	18

1. Introduction

User manuals serve as a guide to learn and reproduce a given product or idea. This document will outline the development of **ARchitect** from the very start of the process to the very end. Our group chose to use the Engineering Design Process to develop it. This starts with first identifying the needs of the customer by empathizing with them. The next step is to rank and sort these needs based on importance and functionality. Additionally, we researched similar products on the market, to assess reasonable standards to gauge our product. These two things will serve as the base for the growth of our product, at every further point in the process, we systematically referred to these to enhance clarity. The next step is to start brainstorming and prototyping. This stage takes the longest but includes formulating thoughts, making prototypes, and receiving feedback. The user manual also contains detailed instructions on the operation of the application.

2. Customer Needs

- a. Listed below are the things the client asked for in the application. They are ranked from most important to least based on the meetings with the client.
 - i. Virtual Reality (5)
 - ii. User-friendly interface (5)
 - iii. User manual (4)
 - iv. View multi-disciplinary layers of the model (4)
 - v. Usable with IOS and Android (4)
 - vi. Costs kept to a minimum (4)
 - vii. Increase worker productivity/efficiency (4)
 - viii. Offline usage (3)
 - ix. Open-source app (3)

3. Metrics & Target Specifications

a. Metrics

Metric	Units
Types of Perspectives	<i>num.</i>
The app is free to download	CAD\$
Quality of Communication	<i>subj.</i>
Comprehensive user manual (simple jargon)	<i>subj.</i>
Time to switch between views	Time(s)
Time to load multi-disciplinary	Time(s)

views	
Cost of Software Development	CAD\$

b. Target Specifications

Metric	Marginal	Ideal	Units
Virtual Reality/Augmented Reality	VR	VR/AR	<i>Subj.</i>
Types of Perspectives	$x > 1$	$x = 3$	<i>num.</i>
The app is free to download	$x = 0$	$x = 0$	CAD\$
Quality of Communication	competent	exemplary	<i>subj.</i>
Comprehensive user manual (simple jargon)	competent	exemplary	<i>subj.</i>
Time to switch between views	$x < 5$	$x < 3$	Time(s)
Load time for multidisciplinary views	$x < 5$	$x < 3$	Time(s)
Cost limit	$x < 100$	$75 < x < 95$	CAD\$

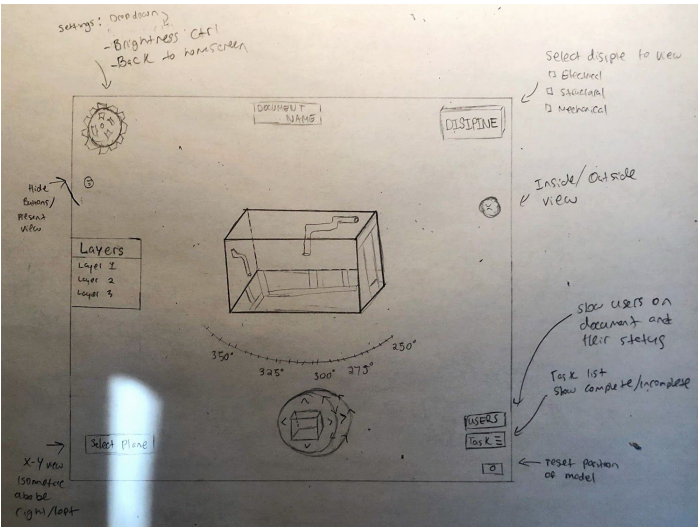
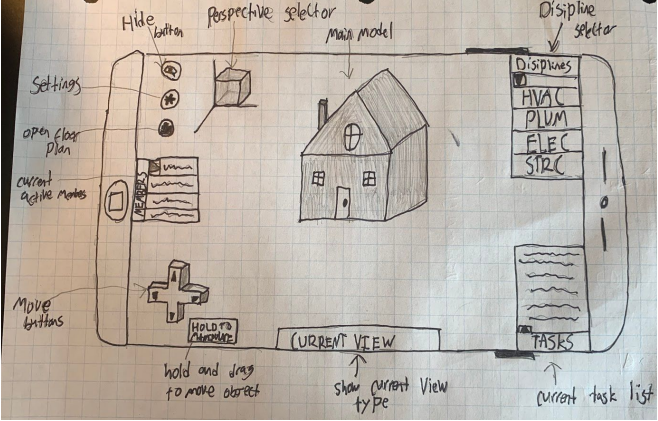
4. Conceptual Designs

The first step in the development of *Architect* was to create conceptual prototypes on paper of what our team desired our app layout and app interface to look like. After we all agreed on a design, our next steps were to bring our ideas to life using various software.

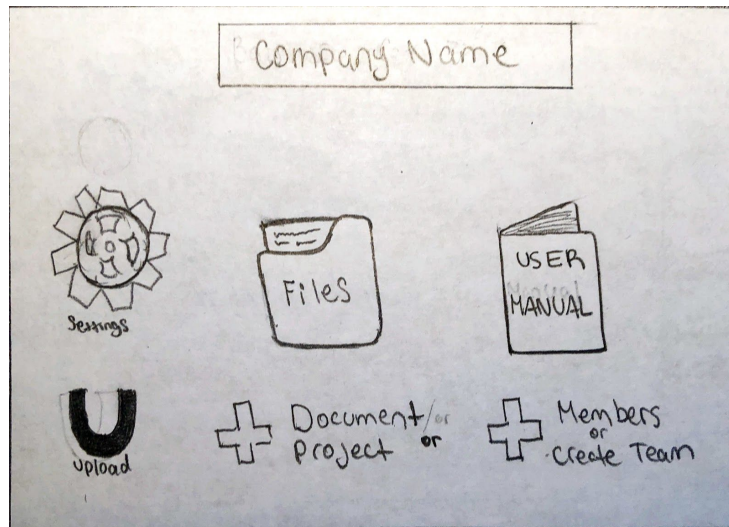
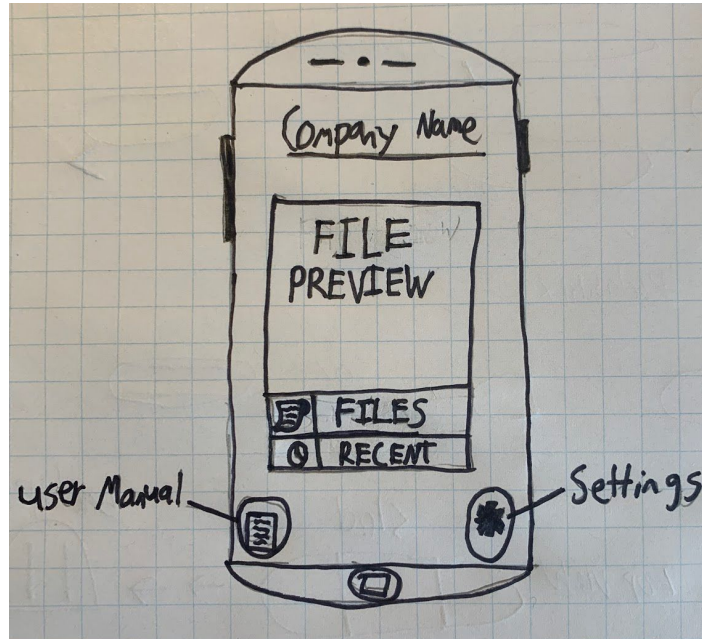
a. Interface View

Featured below are the chosen designs for the Interface view. The key features being, small buttons, task lists, discipline selectors, perspective selector, and a movement function. The usage of small buttons helps keep the screen clutter-free, which increases its simplicity, and ease of use. The perspective and discipline selectors make it easy to switch between views and disciplines, keeping the time to switch and load to a minimum.

Additionally, these functions satisfy the client's need for multi-disciplinary viewing. The movement buttons will utilize simple directions, to make their functionality and usage better. Additionally, the conceptual design for the interface view contains multiple components that were not feasible within the final product. Some of these features include a task list, live collaboration, calibration button, and a degree of rotation manager.



b. App Layout



5. Prototyping, Testing, Customer Feedback

a. Prototype 1

i. Testing Plan

The specific test objective of our prototype is to show a proof of concept for each component of the prototype. The test will include criteria on visual appeal and simplicity, along with technical criteria on the feasibility of the prototype.

ii. Documentation

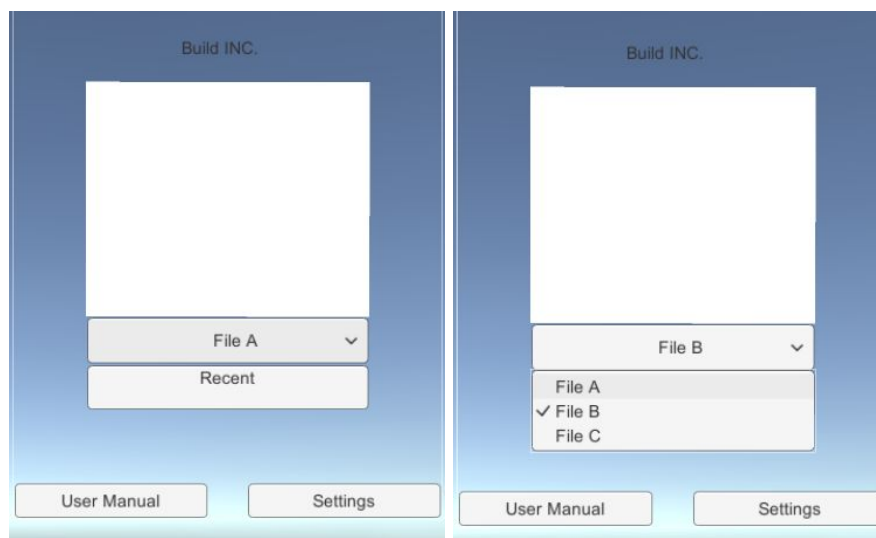
App layout

Featured below is a prototype of our app layout, created using the software Onshape. This is a 3D model of our app layout displayed on an iPhone.



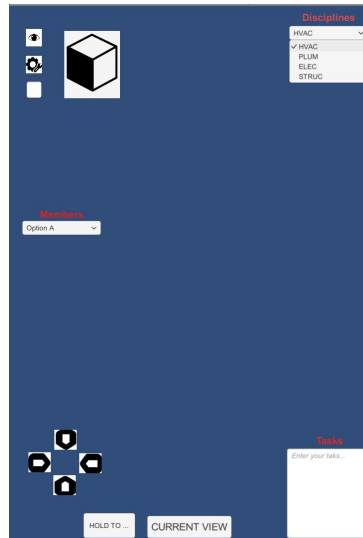
Interactable Buttons

Featured below is a workable dropdown menu along with two clickable buttons.



Interface view

Featured below is the prototype of the interface view of our project. All components of this screen are workable buttons and other UI components.



AR objects

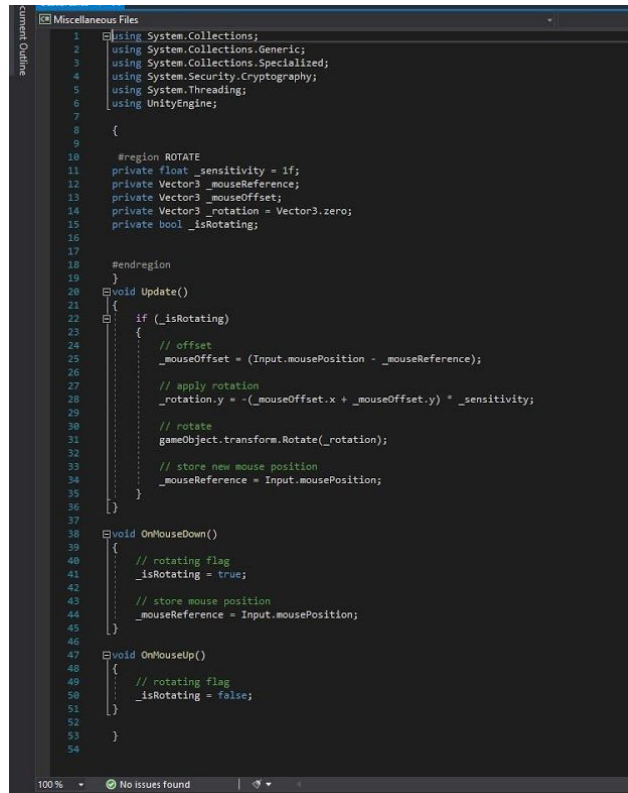
This feature demonstrates the ability to load models into an AR environment in two ways, loading a model into a fixed location in the environment, or attaching a model to a target image which the camera recognizes.



Gesture controls

The code featured below is an attempt to model what the interactions within the app will look like, including zooming in (pinching), panning and rotating the model. So far only a rotation script has been applied to the

game object. Some problems encountered include rotating the object on different axes at once, and applying multiple gestures to the same script proved to be very difficult. The next iteration involves writing multiple scripts with different gesture commands, adding them to the game object however, that does not function properly. The solution we arrived at will be described in prototype 2.



```
1 using System.Collections;
2 using System.Collections.Generic;
3 using System.Collections.Specialized;
4 using System.Security.Cryptography;
5 using System.Threading;
6 using UnityEngine;
7
8 {
9
10 #region ROTATE
11 private float _sensitivity = 1f;
12 private Vector3 _mouseReference;
13 private Vector3 _mouseOffset;
14 private Vector3 _rotation = Vector3.zero;
15 private bool _isRotating;
16
17 #endregion
18 }
19
20 void Update()
21 {
22     if (_isRotating)
23     {
24         // offset
25         _mouseOffset = (Input.mousePosition - _mouseReference);
26
27         // apply rotation
28         _rotation.y = -(_mouseOffset.x + _mouseOffset.y) * _sensitivity;
29
30         // rotate
31         gameObject.transform.Rotate(_rotation);
32
33         // store new mouse position
34         _mouseReference = Input.mousePosition;
35     }
36 }
37
38 void OnMouseDown()
39 {
40     // rotating flag
41     _isRotating = true;
42
43     // store mouse position
44     _mouseReference = Input.mousePosition;
45 }
46
47 void OnMouseUp()
48 {
49     // rotating flag
50     _isRotating = false;
51 }
52
53 }
54
```

iii. Feedback

We decided that feedback from friends and family would be quite useful, as the project is supposed to be user friendly, and simple to navigate. The main feedback we received was to improve the size of our buttons, the text, and button images. We also received feedback that our colour pallet was too light. For our next prototype, we will focus on improving this while enhancing the prototype.

b. Prototype 2

i. Testing Plan

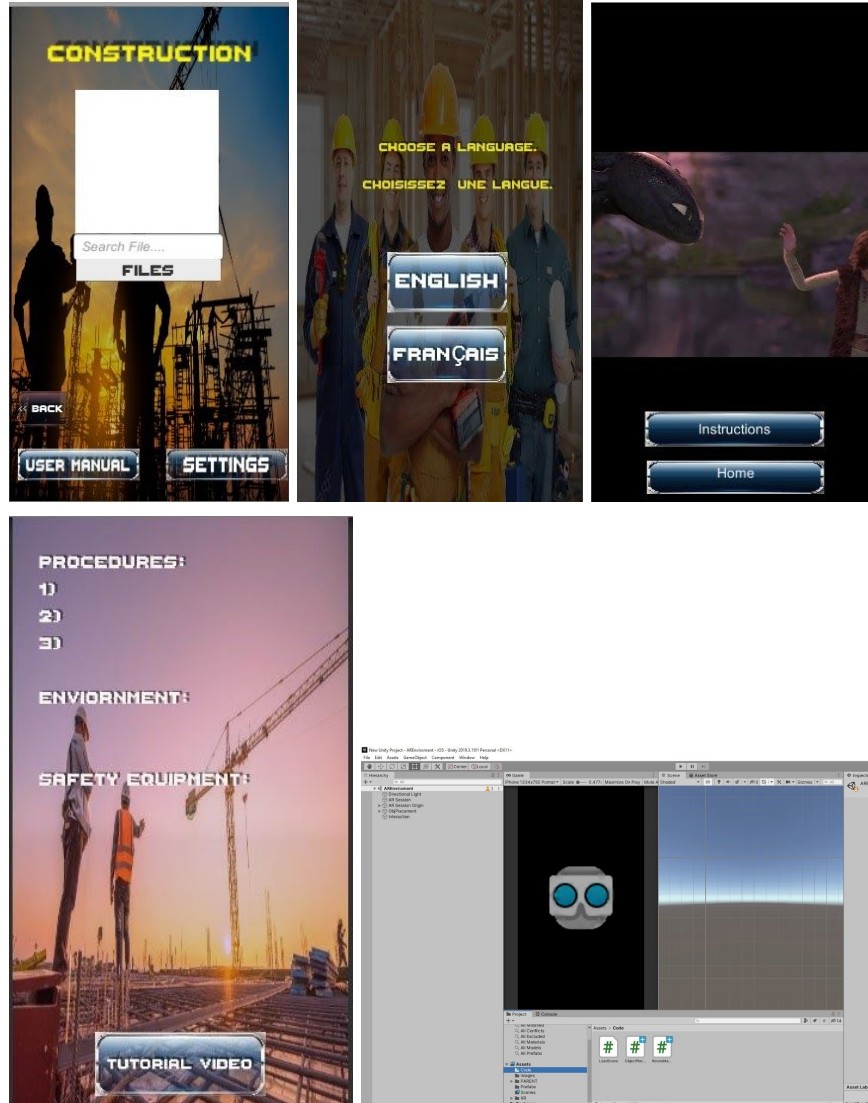
The test objectives for prototype two were to combine the components of prototype one, link multiple scenes together using buttons, and load a 3D

model into unity. The testing also consisted of continued development toward the visual aspects.

ii. Documentation

Featured below are the 4 main screens of the app. Screen one is the language selector. Screen two is the main navigation screen. Screens three and four are both parts of the user manual and contain information along with a video. The last screen is the file screen, where the object can be interacted with. This one can not be documented with a picture, a video is provided in the submission. Also to demonstrate the linked scenes another video has been provided. The final image is the AR environment and the gesture controls associated with it.

In this prototype, we implemented buttons that switched between scenes, this function is essential to every application. To do this in the software Unity, we first had to assign each scene a number by creating a button then adding an event trigger. From there select "pointer up" then change the settings to "runtime only" and "scene loader" as well as add the scene changing code created using language C. All scenes will be given an integer value which can be seen in the build settings, these can be rearranged if desired. The integer value in the event trigger will be the number of the scene to which the button will direct when selected.



iii. Feedback

The client was very happy with the progress made and had no comments on issues with the prototype. Family members that looked at the prototype found it very easy to navigate and visually appealing. One person commented that our french button doesn't work. In our next prototype, this will be one of our biggest focuses along with refining the rest of the app.

c. Prototype 3

i. Testing Plan

The objectives of this prototype were to finalize the app. This included importing fonts, images and button skins to improve the visual aspects. This prototype also includes the inclusion of a french section to increase accessibility.

ii. Documentation

Featured below is the final prototypes split into all its separate components. Each screen is linked together with functioning buttons

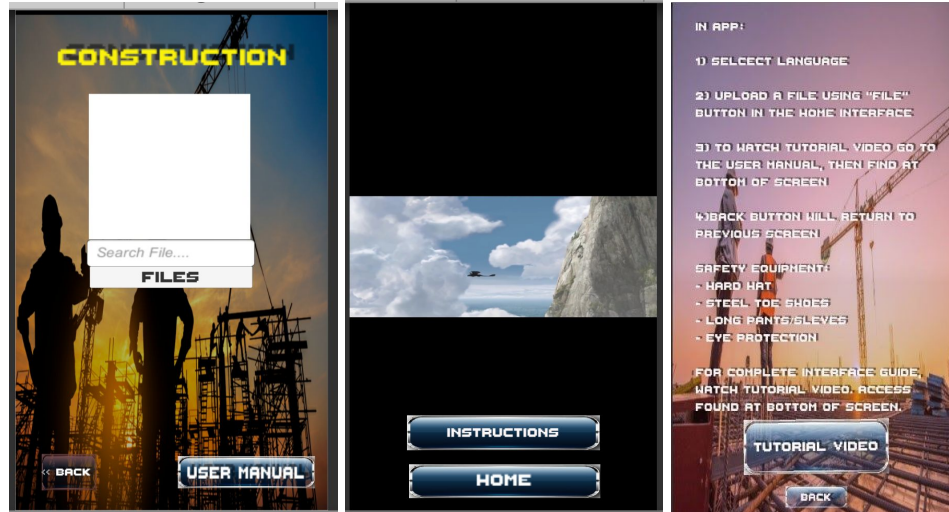
Opening scene

This scene is the first scene and is where the desired language is chosen



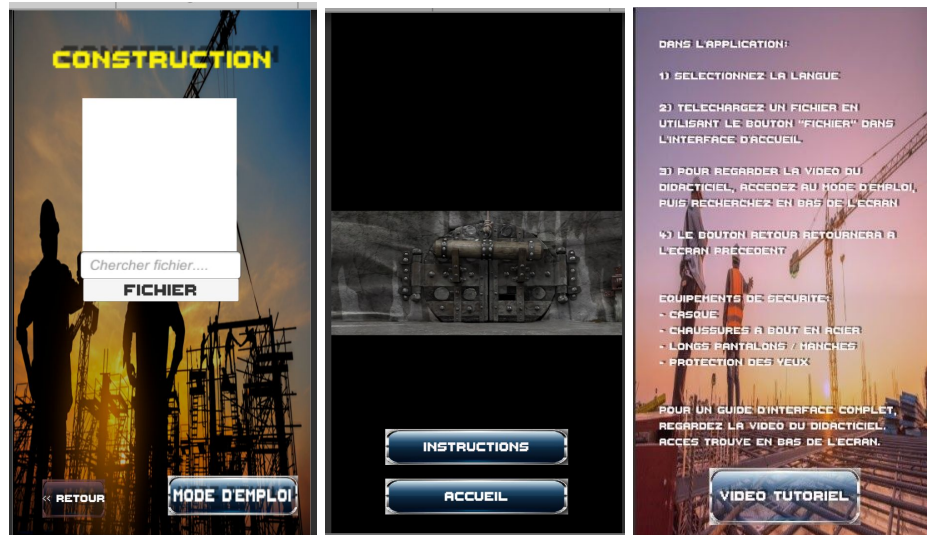
English Scenes

These scenes are the general features of our prototype, they include the home screen which navigates to the tutorial video and instructions



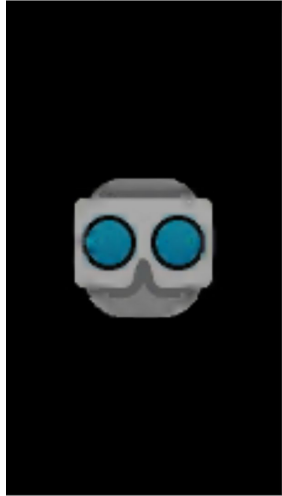
French Scenes

These scenes are identical to the English scenes but are in french



AR Scene

This is a placeholder for the actual AR scene, it is included in the video that was submitted separately.



iii. Feedback

The feedback received on prototype three was overall positive and we have decided that because it is, we have satisfied our stopping criteria. The only feedback we will continue to develop is the development of our name, to help with branding.

6. Final Solution

The final solution is very similar to prototype 3, except it includes an onsite feature, that allows the user to navigate the building in real-time and location. It can run on android and iOS, as per the client's request. To deal with some of the issues from the previous prototypes, the script for the AR environment had to be modified. The following code was utilized:

```
using System. Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
//using UnityEngine.Experimental.XR

using UnityEngine.XR.ARSubsystems;
using System;

public class ARtap2Place : MonoBehaviour
{
    public GameObject placementIndicator;
    public GameObject objectToPlace;
```

```

//private ARSessionOrigin arOrigin;
private Pose PlacementPose;
private ARRaycastManager aRRaycastManager;
private bool placementPoselsValid = false;

void Start()
{
    //arOrigin = FindObjectOfType<ARSessionOrigin>();
    aRRaycastManager = FindObjectOfType<ARRaycastManager>();
}

void Update()
{
    UpdatePlacementPose();
    UpdatePlacementIndicator();

    if (placementPoselsValid && Input.touchCount > 0 && Input.GetTouch(0).phase ==
TouchPhase.Began)
    {
        PlaceObject();
    }
}

private void PlaceObject()
{
    Instantiate(objectToPlace, PlacementPose.position, PlacementPose.rotation);
}

private void UpdatePlacementIndicator()
{
    if (placementPoselsValid)
    {
        placementIndicator.SetActive(true);
        placementIndicator.transform.SetPositionAndRotation(PlacementPose.position,
PlacementPose.rotation);
    }
    else
    {
        placementIndicator.SetActive(false);
    }
}

private void UpdatePlacementPose()
{

```

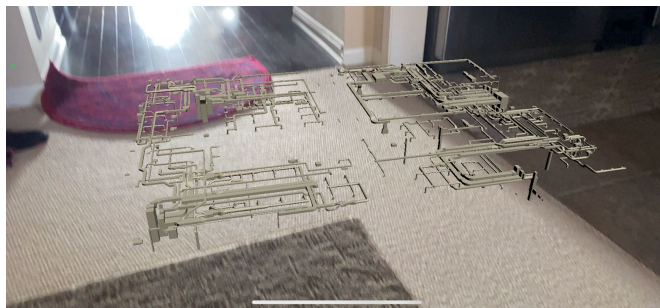
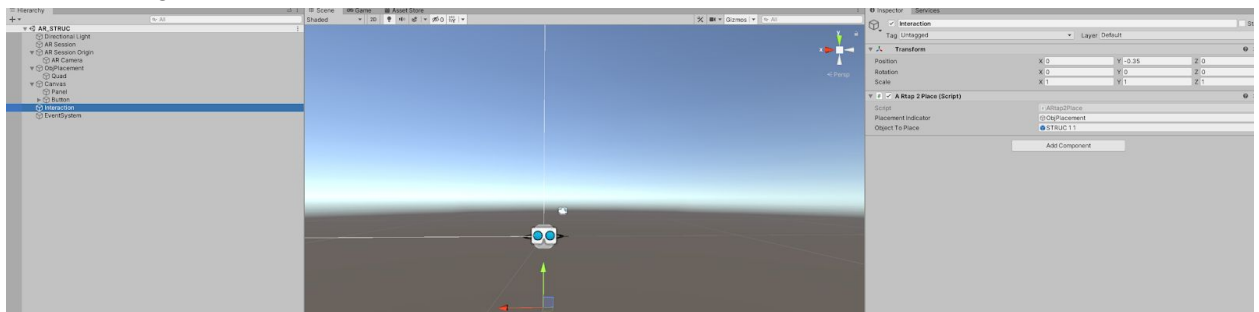
```

var screenCenter = Camera.current.ViewportToScreenPoint(new Vector3(0.5f,
0.5f));
var hits = new List<ARRaycastHit>();
aRRaycastManager.Raycast(screenCenter, hits, TrackableType.PlaneEstimated);

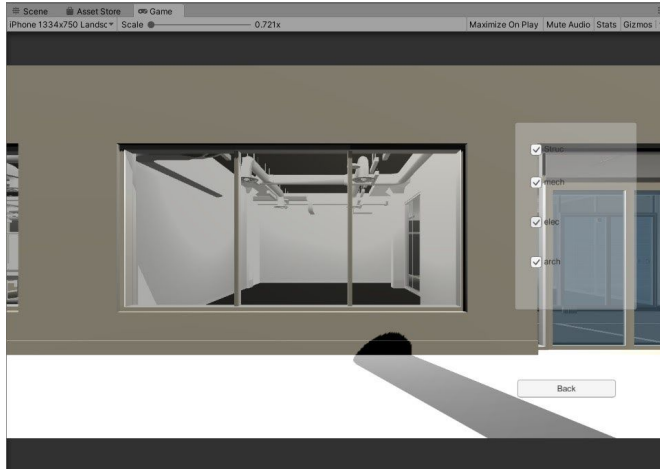
placementPoselsValid = hits.Count > 0;
if (placementPoselsValid)
    {
        PlacementPose = hits[0].pose;
    }
}

```

To script the app, a game object called “Interaction” was created and was supplied with two inputs, one for the model to be placed, and the other for the script regarding the interaction. The interaction script informs the game object that in the AR environment if the user taps the screen, the model is to be placed on the indicator. In the game scene, this interaction object is placed on top of the placement marker so it executes and pastes the model right on top.



ON-SITE feature



This feature was created by overlapping all the layers of the 3D model, adding textures to the walls and windows and adding point light sources throughout all the light fixtures within the model, all within an AR environment within unity. The object placement script was not necessary as the model was preloaded into the scene to allow the user to spawn in and start walking through the building in real-time.

7. Instructions

a. App:

The first screen of the app gives you the option to continue in your preferred language of either English or Francais. After you have selected your language of choice, you will be brought to the main interface of our app. From here, you will find several buttons: File upload, user manual as well as a back button to bring you to the language selection screen. When the user manual button is selected, you will be brought to a screen with details on how to use the app, suggested safety equipment to wear as well as a button that will bring you to the tutorial video. The tutorial video is a complete guide on how to navigate the AR interface. When the video is over, you have the option of going back to the user manual screen or the main interface.

b. Navigating AR Interface:

In order to navigate the AR scenes, one must select the desired model from the file loading screen. The user will then be launched into an augmented reality environment. It is important to ensure you are in a large open space and to clear the room of any debris which may interfere with the plane detection. Next, move the camera around to find the placement indicator, it will be a black target sign. To place the model, tap the screen. Every time the user taps the screen, a model will be placed on the indicator.

8. Conclusion

This document contains a comprehensive outline of the development of **AR***chitect*. It started by outlining the customer needs that were gathered through meeting with the client. These needs were translated into a set of metrics to provide a standard of excellence to abide by during the development process. Once these metrics were completed the group moved on to the planning phase. Each member provided rough ideas through sketches, and the final choices were documented in this report. With the final choices selected, the first prototype was developed. It was tested based upon the outlined metrics and shown to the customer. The results of this prototype were then used to develop a second, and subsequently a third. Through an interactive, well documented prototyping phase, the group was able to create a functional product that the customer was satisfied with. Although functional and complete it can be improved upon, in a few ways. The biggest one is to colour code the separate layers to improve clarity when viewing the model. Additionally, the product could use an improvement on its overall smoothness, including refining the placement and gesture code, as well as implementing VR accessibility.