CS Capstone Project Archive Document

May 24, 2020

Educational AR using WebXR

Prepared for

Intel

Alexis Menard

Signature Date

Prepared by

Group CS19

ARTT - Augmented Reality Teaching Technologies

Andrew Snow

Signature Date

Cody McKenzie

Signature Date

Josh Strozzi

Signature Date

Martin Nguyen

Signature Date

Abstract

Our goal is to create an application for an AR system that will provide virtual tools that the user can use to gain a better understanding of our solar system. We will be coding the application using the WebXR framework which will make the application easily accessible on the web for students and instructors anywhere. We hope this project will provide an easier, quicker and more convenient setup for teachers in their classrooms. We will create documentation to record what we have done and what struggles we had with certain areas. This way if Alexis ever wants to continue the project once we are finished then they will have everything we have done and can share it with the next group so they will know where to start and how everything functions.

CONTENTS

1 INTRODUCTION

During the 2019-2020 school year We worked on developing an AR application using WebXR that will help visually demonstrate core concepts pertaining to our solar system.

2 REQUIREMENTS DOCUMENT

Abstract

Our goal is to create an application for an AR system that will provide virtual tools that the user can use to gain a better understanding of our solar system. We will be coding the application using the WebXR framework which will make the application easily accessible on the web for students and instructors anywhere. We hope this project will provide an easier, quicker and more convenient setup for teachers in their classrooms. We will create documentation to record what we have done and what struggles we had with certain areas. This way if Alexis ever wants to continue the project once we are finished then they will have everything we have done and can share it with the next group so they will know where to start and how everything functions.

2.1 PURPOSE

We are looking to give students a view on how our solar system works. From the beginning of our application students will get a scale view of the sun and the planets in orbit. The student then can zoom in on the earth or any of the other planets to get an idea of how the day/night cycle works as the it rotates on its axis. Students will also be able to click on continents to get information about them. We will then provide an educational quizzes to help students learn.

2.2 SCOPE

Our application will focus on a younger audience but could be used by anyone. The application will be developed to work with Android smartphone devices through the web and may be later extend to IOS and chrome book integration.

2.3 PRODUCT OVERVIEW

The application will be built using the WebXR framework which will provide easy access to the experience by using any device connected to the internet. It will create an environment that is both engaging and productive, with easy to follow controls, and will be built with a school setting in mind. The lesson will also test the user on what they learned from the experience and will help self correct anything that they might not understand by encouraging them to return to the solar system.

2.4 DEFINITIONS

AR: Augmented Reality - a technology that superimposes a computer-generated image on a user's view of the real world, thus providing a composite view.

2.5 INTERFACES

We will be developing a simple interface which will allow the user to interact with the AR experience without getting in the way. The user will be provided with a play/pause button in the upper left side of the screen. A menu drawer in the upper right side that will provide the user with various options such as adjusting the orbit and rotation speeds of the planets. There will also be some text boxes with useful facts and tips for the user and eventually there will be a quiz when the user is finished checking everything out.

2.6 OVERALL DESCRIPTION

2.6.1 PRODUCT PERSPECTIVE

This application is a follow up product on the previous year's VR for Education Application using WebXR. This time the application will use WebXR to demonstrate how AR can be used as a tool to help students learn. The application will serve as a standalone experience that will showcase how AR could be used in an educational setting.

2.6.2 PRODUCT FUNCTIONS

The application will allow students to view the solar system and explore the Sun, Earth, and Moon System. The application will aim to teach users about the day/night cycle, seasons, lunar phases, and the continents/countries on Earth. The application will also include a quiz mode that will test if the user has learned from the AR experience.

2.6.3 USER CHARACTERISTICS

The general user of this product will be students K-12. As such, the application should be designed to be intuitive and not require any special knowledge to use.

2.6.4 CONSTRAINTS

The application will be constrained by the power of the hardware that will run the application. These devices will generally be smartphones or tablets with ARM processors, as such the application will need to be designed not to require overly powerful hardware to run.

2.6.5 ASSUMPTIONS AND DEPENDENCIES

The application will assume that the device running has a forward facing camera and a browser that supports WebXR, otherwise the application will simply tell the user that the AR experience can't be run on their device.

2.7 SPECIFIC REQUIREMENTS

2.7.1 EXTERNAL INTERFACES REQUIREMENTS

The system will interface with the user through either a smartphone, tablet, or handheld device. The user will interact with the application through touch controls on their screen. The AR application will use to the forward facing camera to know where in space it is located and the screen to show the user the video feed with the AR scene superimposed onto the video feed.

2.8 SYSTEM FEATURES

2.8.1 VIEW THE WHOLE SOLAR SYSTEM

The application when first started will show all the planets orbiting the sun. The user will be able to control how fast time flows, to see how long it takes for the planets to orbit the sun. There will also be a time display to show how long it has been since the start.

2.8.2 SUN AND EARTH MODEL DAY/NIGHT CYCLE

The application will show why and how day and night happen on Earth. The user will be able to control the rotation of the earth to see how night and day work for the whole planet.

2.8.3 SUN, EARTH AND MOON MODEL LUNAR PHASES

The application will show the sun in fixed location outside of the view of the camera and focus on the Earth and the Moon orbiting it. The user will be able to control where the moon is in its orbit and at the same time see what the view of the Moon is from the Earth.

2.8.4 EARTH MODEL CONTINENTS/COUNTRIES INTERACTIVE GLOBE

This mode will focus in on just the Earth. The Earth will have continents clickable. The user will be able to tap on the continent and see information about the continent.

2.8.5 QUIZ MODE

At the end of the experience the system will quiz the student. This will allow us to see if the student has learned anything from this experience. It is crucial that we can gauge how well the application teaches students about the concepts covered in the experience. Most importantly this will show teachers and instructors that learning is occurring and that AR has real uses as and educational tool.

2.9 PERFORMANCE REQUIREMENTS

The application must be able to run on smartphones and tablets, while still delivering 60 or even 90 frames per second to maintain the immersion of the AR experience intact. The application must maintain 60 frames per second at all times, even if the hardware is several generations behind the current models.

2.10 DESIGN CONSTRAINTS

The application will be constrained to devices with a camera and a browser that supports WebXR. On Android devices, the application will also be limited to devices that have ARCore installed.

2.11 SOFTWARE SYSTEM ATTRIBUTES

The code should be written in a way that is easily maintainable. The code base will have a longer lifespan than the timeline of this senior project, it is important the future teams that look at the code can understand and modify the code easily.

2.12 GANTT CHART

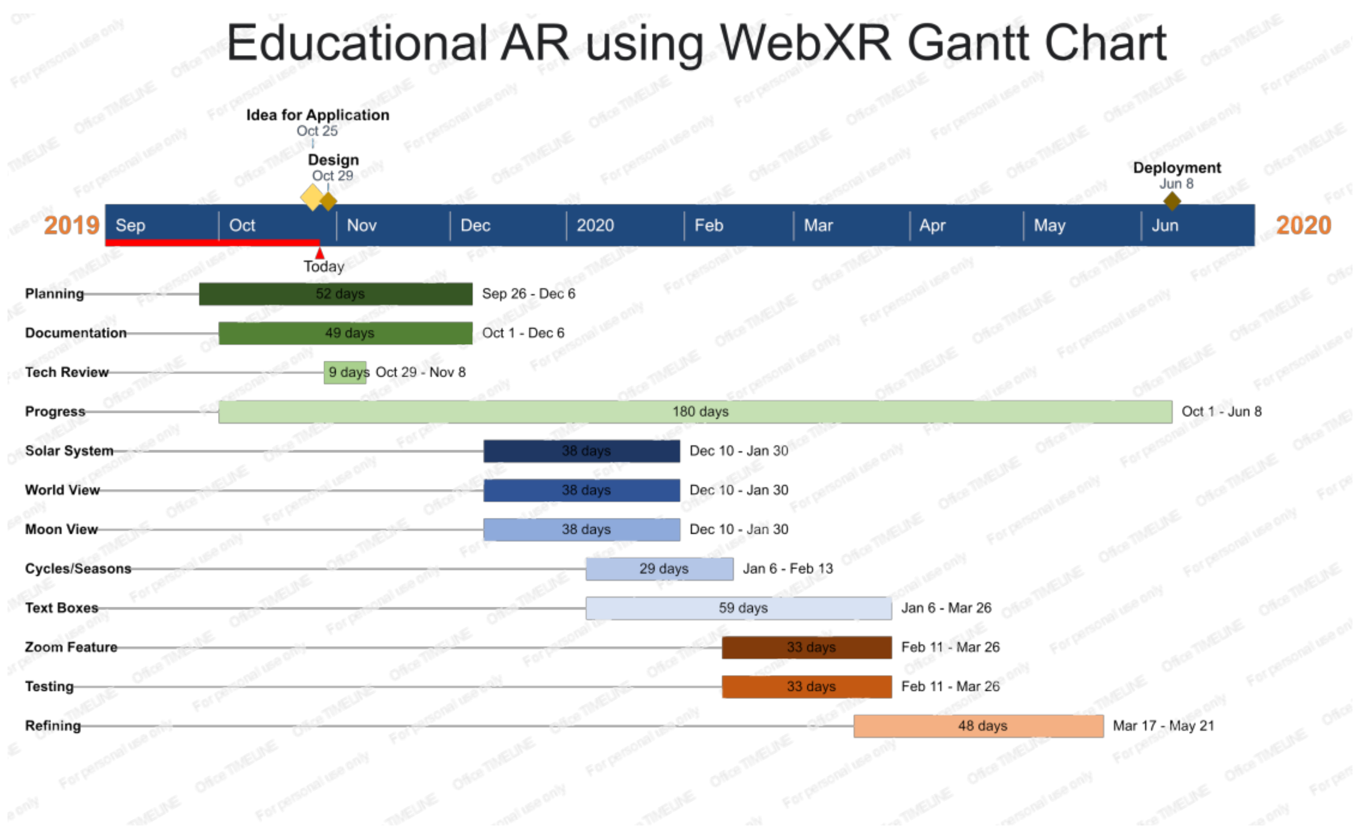


Fig. 1. Gantt Chart for Fall 2019

3 Design Document

Abstract

Our goal is to create an application for an AR system that will provide virtual tools that the user can use to gain a better understanding of our solar system. We will be coding the application using the WebXR framework which will make the application easily accessible on the web for students and instructors anywhere. We hope this project will provide an easier, quicker and more convenient setup for teachers in their classrooms. We will create documentation to record what we have done and what struggles we had with certain areas. This way if Alexis ever wants to continue the project once we are finished then they will have everything we have done and can share it with the next group so they will know where to start and how everything functions.

3.1 Purpose

This document explains the process and procedures to building an educational AR application about the solar system and the Earth. This document is conformed to the IEEE 1016 System Design Document specification, and establishes the design and relationships of all the different components in our project.

3.2 Scope

The Augmented Reality (AR) application is an educational application that will teach the user about the solar system. When fully zoomed in the app will then inform users about the planet. There will be a text box that pops up as the user goes through the app to give some fun facts and other educational tips about planets and the continents on earth. A quiz at the end will ask quiz questions that the user can either take or dismiss. The reason this application is necessary is for classrooms to have a convenient tool that is cost effective, saves space and is easy to set up.

3.3 DEFINITIONS

Item	Definition
API	Application programming interface that acts as an intermediary that allows two applications to talk to each other.
AR	Augmented Reality, an interactive experience with the real world environment that is enhanced with computer generated objects, sounds and effects
ARCore	Software development kit developed by google that enables AR applications to work with the given device.
WebXR	WebXR is a device based API that provides access to input and output capabilities for AR development.
WebGL	A javascript API that can render 3D graphics within a web browser.

TABLE 1: Definitions

3.4 INTRODUCTION

The following sections dive into specific components within the design of our project and the choices we have made on how and why we want to implement it. The sections cover the relational interactions between different components of our project, the use cases for the individual systems, the information gained from the user and environment, and the way we build the AR scene.

3.5 CONTEXT VIEWPOINT

3.5.1 VIEWPOINT DESCRIPTION

The Context view of the educational solar system AR app defines the relationships, dependencies, and interactions between the system and its environment—the people, systems, and external entities with which it interacts.

3.5.2 VIEWPOINT DESCRIPTION

Some of the concerns for the application is how the camera will track with the real world and where the solar system will anchor itself, whether the camera gets covered or is shaking excessively and the fact there has to be some sort of internet connection in order to access a browser and get the application running.

3.5.3 DESIGN ELEMENTS

Design Entities: The application will require a browser that can run WebXR in order to get the app running. The phone/tablet running the application needs to be able to support an ARCore. The phone/tablet will also need a camera so the environment can be seen and we can use it to throw the solar system in the correct place. A gyroscope in the phone would be preferred because then we can have an easier time rotating the phone and tracking the environment.

Finally we need to have the ability to read input from the user. Design Relationships: The application will need to be able to view the real world through the camera and provide a feed to the phone for the user to see. The feed on the phone will show the AR application so the user will then see the solar system and be able to manipulate the app by touching the parts we have setup in the user interface. All of which can be run from the browser that supports WebXR. Design Constraints: The application can only work as well as the user will understand, so all we can do is try to make the interface easy to use and understand so the user will not be confused by what is there. There is only so much that can be done for the unpredictability of what users will try and test when running the application.

3.6 COMPOSITION VIEWPOINT

3.6.1 VIEWPOINT DESCRIPTION

The Composition viewpoint describes the way the design subject is (recursively) structured into constituent parts and establishes the roles of those parts.

3.6.2 DESIGN CONCERN

The project can generally be viewed as three major components. The front-end, the back-end, and the developer tools. The front-end will include the static website files and assets, as well as the JavaScript that will be run on the client's browser to enable the web application to deliver it's AR experience to the user. The back-end will be the server that the web application will use to deliver the static files and assets to the clients. Lastly, the developer tools will enable 4 developers to build, test, and deploy new updates quickly to the production server. The design elements section will go into further detail about how each component will function and be built.

3.6.3 DESIGN ELEMENTS

The front-end will be made of three components: The static website (html, css, and basic js), the AR API, the 3D rendering engine. The static website will be the front page for the web application and will also be how users navigate to the AR experience. The AR API will handle the communication between the web application, the hardware the web application is running on, and the 3D rendering engine. It will handle interpreting the inputs from the camera, touch screen, and gyroscope, and relaying how that will affect the 3D scene to the rendering engine. From there the rendering engine renders the image and passes back to the AR API to be displayed on the user's display. For this application, the AR API will be WebXR and the 3D rendering engine will use three.js. The back-end will be comprised of a NodeJS instance utilizing Express to run as an HTTPS server. Its main purpose will be to reply to HTTP requests from clients with the requested files. Since all the files for the web application are static and do not require any server side processing, no other NodeJS packages are required. The main concerns for the back-end will be to make sure the server has enough bandwidth and processing power to serve all its clients, and to maintain a server up-time of at least 90%. As for the hosting of the server, the application will use ZEIT Now to provide the hardware and URL for the web application. ZEIT Now will run the application's custom NodeJS package on their servers and provide a URL where HTTP requests can be sent to. The developer tools for this project will revolve around tools that are integrated directly with GitHub. They include CirclCI and ZEIT Now. CircleCI will be configured to run build tests whenever a pull request is made to verify the integrity of the changes made to the codebase. After a pull request is merged, ZEIT Now can be configured to immediately deploy the new version to production.

3.7 INTERFACE VIEWPOINT

3.7.1 VIEWPOINT DESCRIPTION

The Interface viewpoint provides information on how different parts of our program communicate with each other.

3.7.2 DESIGN CONCERNS

It is important that all the interfaces between the different components work according to specification. As a web application, many of the interfaces between the different components have already been built into the libraries and packages themselves or can be included with extra libraries. For example for the interface between the https server and the front end client, the https protocol of the internet will handle all communications and make sure the correct files make their way across the internet to make it to the client. On the backend, NodeJS has many packages that can be combined together with the node package manager into a custom package. The node package manager acts as the middleman between all of separate libraries and handles the building, testing, linting and deploying of the web application. On the frontend, the interfaces are much less tested and proven. This is especially true for the WebXR API. An interface between WebXR and three.js has already been created, so the developers will not have to worry about creating an interface for passing the 3D renders between the two components. The interface between WebXR and the browser and the rest of the hardware on the device, were recently finalized. The current version of Chrome supports WebXR if the flag for it is set. The next version of Chrome (version 79) will come with WebXR enabled by default. Currently, Chrome is the only browser that fully supports WebXR and as a result will be the only browser with an interface for WebXR to communicate with the hardware of the device to create an augmented reality experience.

3.8 INTERACTION VIEWPOINT

3.8.1 VIEWPOINT DESCRIPTION

The Interaction viewpoint defines how our program will accept input from the user, and how these inputs will be used.

3.8.2 DESIGN CONCERNS

Our project will revolve around user input. These inputs allow the user to interact with the 3D scene. These inputs will come from two sources, touch from the touch screen and change in the devices orientation and position. Firstly touch will enable the user to interact with the different objects and menus within the 3D scene. Touching a planet will move the camera to a closer position to view the planet.Touch will also allow the user to navigate through the menus, answer questions to the quizzes, and acknowledge dialogue. The user will also be able to manipulate the camera with the movements of their device. The two will be synced providing the user with a seamless experience.

3.9 ALGORITHM VIEWPOINT

3.9.1 VIEWPOINT DESCRIPTION

The Algorithm viewpoint details how curtain components within the 3D scene of our project will operate.

3.9.2 DESIGN CONCERNS

With our project we will use algorithms to impact what takes place in the 3D scene. We will have two main algorithms, one which will set the orbital and rotational movements of objects in the scene and another that will work with the movement of the camera. Every planet and moon in the scene will be set to orbit around some center point. These center points will represent where the centers of the Earth and sun are in the 3D scene. Each orbiting object will be set a distance on how far away the object will reside while orbiting. The distance will be a scaled down version of the solar system. In addition to orbiting around a defined center point, the orbiting objects will also be rotating around a set axis. This again will be in relation to our solar system. The camera is the other algorithm that will be implemented into the program. Because we are creating an AR application, the camera's position and orientation in the 3D scene can be altered when the physical phone is moved. This provides the user with a range of movement to view the 3D scene. The user will also be allowed to interact with the planets in which the camera will move towards the desired object for an up close view. The algorithm will need to find an appropriate location within the 3D scene to view the desired planet and will need to keep the original camera orientation and range displacement to keep from disorienting the user. The algorithm will also need to set a path in which the camera would move that is obstacle free, to prevent clipping through objects.

3.10 CONCLUSION

The educational solar system application we will be programming will follow the above criteria and viewpoints. As mentioned above, this application will be a web based program that way teachers won't have to have anything downloaded previously. It will be as simple as clicking on a link and then using the app. The app will use the forward facing camera and screen of the device in order to display the AR scenario which will allow the user to interact with the solar system. The viewpoints discuss the concerns and entities we expect to have/need for the application to function properly. Updates will be made to this document as changes are made to the design to accommodate for issues and workarounds as they come up.

4 TECH REVIEW - ANDREW SNOW

4.1 Introduction

Our project involves us building a web based AR application. We will need to use assets to represent the planets and parts of the UI visually. Assets can be portrayed in different formats containing bits of data on how the object is to be represented. Different formats can alter what data can be stored and how it is used. The data stored can represent the objects geometry, materials, textures, animations, and more. We need an asset format that is easy to access on a web page and provides the necessary components for our solar system and planetary views.

Our project will also need a graphics library so we can display the different objects in our scene. We need an engine that will work on the web and can cover the necessary functions of our project. Within the scene we will need a graphical library that can create 3D objects in a 3D space, move the main camera, and work with a central light source.

4.2 Asset File Format

4.2.1 FBX

FBX (Filmbox) is a proprietary format that is commonly used within the film and gaming industries due to its support for animation. The file holds support for the objects geometry and textures but when exported does not contain any data on materials that may have been used to provide a shiny, dull, or even reflective surface. 1 Due to the file being proprietary it has its limits, but Autodesk who owns FBX have made it work interchangeably with AutoCAD, Fusion 360, Maya, 3DS Max and other software packages. In addition, it is also used as an exchange format which facilitates high fidelity exchange between 3DS Max, Maya, MotionBuilder, Mudbox and other proprietary software. 2 This format is held in high regard and has a lot of resources that we can use when developing our project.

4.2.2 OBJ

OBJ is a neutral format which means it is open source and can be edited by anyone on the team using appropriate CAD software. OBJ format is primarily used for 3D graphics applications and 3D printing. Dibya Chakravorty describes the OBJ file format, "In a nutshell, the OBJ file format stores information about 3D models. It can encode surface geometry of a 3D model and can also store color and texture information. This format does not store any scene information (such as light position) or animations." 3 This file format can provide unique ways to define / adjust a geometry such as using free-form curves and free-form surfaces. The idea is to use a collection of these predetermined geometries to form the surface of the model. The advantages of using free-form surfaces are similar to the advantages of using free-form curves – they are more precise and they lead to smaller file sizes at higher precision compared to other methods." 4

OBJ files can support a companion file called the Material Template Library (MTL), which can store all information concerning the objects textures, colors, and materials. In addition MTL file can support texture mapping. Texture mapping is a graphic design process in which a two-dimensional (2D) surface, called a texture map, is wrapped around a three-dimensional (3D) object.Thus, the 3-D object acquires a surface texture similar to that of the 2-D surface. 5

The OBJ file format has become one of the most popular for simple 3D applications. It is one of the more lighter file formats when it comes to size and has lots of information online on how it can be implemented into a scene.

4.2.3 GLTF

GLTF is a file format that is neutral format so it works with pretty much any program. This format is used for 3D scenes and models using the JSON standard. This format minimizes both the size of 3D assets, and the run time processing needed to unpack and use those assets. 6 This format supports textures, scene camera, Node hierarchy, and physically based rendering materials. This file format can also use a .bin file to keep track of the geometry's vertices and indices as well as inverse-bind matrices.These two files can be combined into a single file using a GLB file extender.

According to Threekit, "This file format has begun to see strong industry adoption, particularly among web-focused JavaScript frameworks such as Three.js and Babylon, as well as web-focused companies such as Google and Facebook. The format was designed for compact file size, fast loading, run-time independence, and complete 3D scene representation." 7 Since this is web-focused this would be a viable choice for our project.

4.3 WebGL Framework

4.3.1 Three.js

Three.js is a widely used web-based library that has a strong community that has lead to tons of documentation and examples that we can use for our own project. According to aerotwist, "With it you can create cameras, objects, lights, materials and more, and you have a choice of renderer, which means you can decide if you want your scene to be drawn using HTML 5's canvas, WebGL or SVG." 8 It's primary purpose is to display and animate 3D computer graphics which is perfect for our AR project. Below are some of the many features that three.js offers that will be used during the project development if three.js is chosen.

First off three.js comes with a file loader that can be called to add in a 3D model from a file of a variety of different file types. Once a model is loaded in, a number of methods can be accessed to alter the properties of the 3D model including moving the object to a new location in the 3D space. For our project we are going to orbit an object around another object as well as rotate an object upon its axis. These can both be achieved through the rotation method, which allows us to rotate an object along a defined axis. Three.js can also create a camera and move it with appropriate methods that come with the object. For our project we will be using the Perspective Camera object which is most commonly used when displaying a 3D scene. We will also be using a light source and shadows to showcase the day/night cycle of the earth. To simulate this Three.js can create a Point Light to emit light in all directions from our central point of the sun, thus giving us the foundation of our solar system.

4.3.2 Babylon.js

Designed by Microsoft, Babylon.js is a javascript library for displaying 3D graphics in real time on a web browser using HTML5. It contains a lot of functionality for arranging a 3D scene and animating it. Babylon has recently moved to version 4.0 which they say "marks a major step forward in one of the world's leading WebGL-based graphics engines.From the powerful new Inspector, best in class physically-based-rendering, countless optimizations, and much more, Babylon.js 4.0 brings powerful, beautiful, simple, and open 3D to everyone on the web." 9 Similar to three.js, Babylon.js has a community that supports it. There are lots of examples across the web as well as documentation, if a bit scattered, to help get us started.

Like Three.js, Babylon.js provides an easy way to create a scene and fill the scene with 3D models. It also provides a very similar way for an object to rotate along a local axis and orbit around another object. However Babylon.js differs with its far superior support of different camera models. The camera we would be using would be the Device Orientation Camera. A camera specifically designed to react to device orientation events such as a modern mobile device being tilted forward or back and left or right. 10 Babylon provides the same light settings as Three.js, in which case we would be using the point light.

4.3.3 A-Frame.js

A-Frame.js is an open source javascript library that was originally built to simulate VR experiences. They have now started to move into applying AR to their applications as well. Already a community has grown from it, and with the community has come a slew of examples, tutorials, and documentation. The interesting part of A-Frame.js is it focuses its AR experiences to a defined marker that the system can pick up. From these predetermined markers your scene will appear. A-Frame is commonly used with AR.js to provide more features that can be used throughout the scene. This seems great for simple AR projects.

Sence A-Frame was built for VR integration, all the necessary components that we need for our project will be present. The only part we would have to figure out is transferring those parts over to an AR scene.

4.4 Conclusion

For our project we want an asset file format that is easy to implement within a web server and gives us enough control over the textures and materials that will be used with it and we will also need a webgl framework that will be easy to work within an AR environment that can also provide the proper level of functionality for us to manipulate the objects in the scene.

For our project we are going to use the GLTF file format. This format provides us with everything we need and sense it is built with web pages in mind, this will be ideal for our project. We are also looking to get our assets from Sketchfab and they export with this file format.

As for our webgl framework, We will be using Three.js. This framework provides the most examples and easy to find documentation to lend us the most assistance in creating this project. Everything we need to make our solar system can be found using this library.

All the options that we could have gone with were good, but what we have chosen will assist us the most.

5 Tech Review - Cody McKenzie

5.1 Introduction

The goal of this project is to provide students a fun and easy mode of learning using AR. Our application will focus on kids from kindergarten up through high school. Using AR for some important material in classes will free up a lot of space due to lack of clutter and provide more time for education because there is not much to the setup for AR.

My focus will be the implementations of our User interface and how it will look on the phone or tablet. I will also be specializing in the web framework language and the input testing to make sure everything works properly before field testing.

5.2 UI Visualization

5.2.1 *Elix*

Elix is a community-driven collection of web components for user interfaces (UI). The Elix project believes that implementing high-quality, general-purpose components is best done as a community effort. This spreads the cost of creating the components across organizations, and ensures that the resulting components satisfy a broad range of concerns and can be used in many contexts [1]. Elix has a list of over a hundred UI patterns that we would be able to use in our program and majority of the components have a link to a document that explains what they are and how to use them [2]. There isn't much for reviews on Elix but after looking through their site and some of their blogs it seems to be a pretty good component library with a great team that updates and works on the library quite consistently. I am definitely considering them as our got to UI library due to how many components they have and how much detail/help they give for each component.

5.2.2 *Wired Elements*

Wired Elements is a series of basic UI Elements that have a hand drawn look. These can be used for wireframes, mockups, or just the fun hand-drawn look [3]. This UI would be wonderful because of the hand-drawn look it brings which will feel relatable and fun to our audience which will generally be young children. The website for wired-elements has quite a few example components that when clicked on bring you to documentation on each component to see how one will implement them into their application as well as talk about how the component works [4]. ”The Wired Elements collection, from the team wiredjs, has a big reputation amongst web component UI groups” says Binh Bui about wired-elements, according to Binh there is a lot of customizability and ease of use with wired-elements which makes the library extremely useful and why I am considering it [5].

5.2.3 *Angular Material*

Angular Material is another source for finding extremely useful components for our UI when we develop our app. Angular Material has a list of the components they offer, which are sorted by certain categories to help narrow done what is needed for certain areas of applications [6]. Each component is able to be clicked on and will take the user to a documentation page that will then proceed to explain how the component works and how to implement it in your project. In an article by Maina Wycliffe there is mention of how attractive Angular Material is by providing a nice User Experience with its components, Maina says ”What attracted me to Angular Material was the User Experience of its component. How intuitive to user action they were as compared to Bootstrap own components” [7]. This sheds some light on how useful Angular Material is in providing a friendly and fun experience for users. So far this seems to be the most widely known and used option out of the three I have here and might just take the cake on which UI library we go with.

5.3 WEB FRAMEWORK LANGUAGE

5.3.1 REACT

React is currently one of the top choices for web framework languages for job markets and in 2018 it was the most demanded framework [8]. React is so popular due to its ability to make the current view on an application more modular by making reusable components which makes creating large web applications much easier and more efficient. However React does have a decent learning curve as it uses a more intermediate level of understanding for JS concepts. Another thing that is against us on this is the fact Alexis would prefer not using React because he believes it would take away from some of the deeper learning that we would get from using JS itself.

5.3.2 ANGULAR

Angular is another very popular web framework language and it is produced by a team at Google which seems to be where that popularity comes from. Angular is a toolkit for enhancing HTML, it lets you extend HTML with a new vocabulary that will turn a static page into a living and breathing web page without even adding any JS [9]. Another contributor to Angulars popularity and usefulness would be the fact it seems to make hard problems become more trivial. However Angular does not seem to stand well to the time test according to Permalink, as they played with the framework more they seemed to run into a lot of issues and challenges. The first issue being the fact Angular locks you in to only being able to use their own specific directives, templates and code. The second big issue is the fact that the framework gets quite complex due to its steep learning curve and its poor design. Permalink mentioned that if they stuck to the way Angular does things rather then trying to give their application a personal feel then thinks might have gone better, but a framework should not be this limiting and cause a user to feel so trapped in one way of doing things [9].

5.3.3 HTML, CSS AND JS

Finally, our last option that I looked into is just the normal way of programming web-pages, using HTML, CSS and JS to setup all of our framework. This is the framework we will be using most likely because it is what Alexis would prefer so we can develop a better understanding of how our app will function and look. This requires a lot more implementation and work but our application will be the way we want rather than the way one of the frameworks like React or Angular will set things up. At this point we all feel quite comfortable programming in HTML, CSS and JS because we have all learned the basics for web programming. It will be useful to come up with our own setups for our application because it will incorporate a lot of different features, such as zooming, scaling and multiple camera points, so understanding our code thoroughly will be crucial.

5.4 INPUT TESTING

5.4.1 PUPPETEER

The Chrome team bundled their Headless Chrome API into a Node package with a fully fledged API that's callable from Javascript called Puppeteer. This includes navigating and rendering pages, interacting with page elements, taking screenshots, and executing scripts and adding style sheets within the page's context [10]. In other words puppeteer is pretty much a browser that is able to be programmed that way we can test our software by setting up autonomous clicks, typing and even automatic touch controls. This is key for our testing because we can then set Puppeteer up with some commands to have it test our application or even give it random commands to see what happens to our application when things randomly happen. This is a very popular choice for input testing and will most likely be the tester we will implement in our project.

5.4.2 SELENDROID

Selendroid is a test automation framework which drives off the UI of Android native and hybrid applications (apps) and the mobile web. Tests are written using the Selenium 2 client API [11]. Selendroid does have a few system requirements such as Java SDK of 1.6 or later, the latest Android SDK and at least one existing android virtual device installed. Selendroid does work on any operating system (Windows, Mac and Linux), however if it is a 64-bit version of Linux the program will need some extra packages added, which can be found on the Selendroid website. AS mentioned above having an automated input tester is crucial because we can make the automation perform random actions or certain actions we need to test in order to see how our program will handle things. According to the website Selendroid may an old input testing program so I am not sure how likely we will be for using this one but it seems like a good input tester and has a lot of documentation on how to work the program on their website so I am still considering it.

5.4.3 TESTCOMPLETE MOBILE

TestComplete according to their site is ”the industry's first automation testing tool with a hybrid object and visual recognition engine to test every desktop, web, and mobile application with native BDD-style Gherkin syntax and script or scriptless flexibility” [12]. SmartBear, the company that developed TestComplete, was named a leader in Gartner Magic Quadrant for software test automation which is a pretty admirable feat since Gartner is quite a large business with a lot of important software to test. TestComplete can provide us with continuous testing for DevOps, building of automated UI tests automated test reporting/analysis which are pretty useful. This is definitely going to be our runner up next to Puppeteer, we will be moving to this one if Puppeteer gives us to many issues with our testing. The requirements for TestComplete are easily accomplishable as most every new computer (or even majority of the old ones) have the requirements taken care of. The main downside for this though is that it seems to be a pay to use system, and the amount is not cheap. Since we don't need all of it's components paying a hefty price just to use some pieces would not be worth the program.

5.5 CONCLUSION

Since our application will be web based each one of these topics must be able to work with web based code, which all of what I researched should be good there. I have decided to go with Angular Materials for our UI visualization library because it is highly recommended and seems to have the best options that we will be needing for our AR application. AS for the web framework language we will be using the typical HTML, CSS and JS because it will give us the most useful experience and allow us all the Customizability we will need, Alexis also expressed he would prefer this to be our framework language. Lastly, I have decided that we will be using Puppeteer for our input testing because it is free, useful and not too difficult to understand/implement in our application.

References

[1] J. Miksovsky, "Elix." https://github.com/elix/elix?source=post_page-----9d4476c3f103----------------------, 2018.

[2] J. Miksovsky and R. Bearman, "Component.kitchen." https://component.kitchen/, 2018.

[3] wiredjs, "wired-elements." https://github.com/wiredjs/wired-elements?source=post_page-----9d4476c3f103----------------------, 2019.

[4] Bit, "wired-elements." https://bit.dev/wiredjs/wired-elements?source=post_page-----9d4476c3f103----------------------, 2019.

[5] B. Bui, "Wired elements, a set of hand-drawn ui elements, part 1 (button, checkbox, combo card)." https://vaadin.com/blog/wcw-18-wired-elements-part-1, 2018.

[6] Angular, "Angular material." https://material.angular.io/, 2010.

[7] M. Wycliffe, "What i learned when i switched to angular material from bootstrap." https://codinglatte.com/posts/angular/what-i-learned-when-i-switched-to-angular-material/, 2017.

[8] D. Kostrzewa, "Is react.js the best javascript framework in 2018?." https://hackernoon.com/is-react-js-the-best-javascript-framework-in-2018-264a0eb373c8, 2018.

[9] Permalink, "An unconventional review of angularjs." https://www.letscodejavascript.com/v3/blog/2015/01/angular_review, 2015.

[10] A. Sapozhnikov, "Using puppeteer to create an end-to-end test framework." https://medium.com/uptake-tech/using-puppeteer-to-create-an-end-to-end-test-framework-f1e7e008c793, 2019.

[11] Selendroid, "Selendroid, selenium for android." http://selendroid.io/, 2012.

[12] SmartBear, "The easiest-to-use automated ui testing tool with artificial intelligence." https://smartbear.com/product/testcomplete/overview/, 2019.

6 Tech Review - Martin Nguyen

6.1 Introduction

The goal of our project is to provide an example for future developers interested in interfacing with the WebXR API. This API gives developers access to end users' Virtual Reality and Augmented Reality systems, so that they can make interactive VR and AR experiences for the web.

One of my focuses in the project will be project setup. This includes choosing the best language suited to our style of development, the build tools (in our case an asset bundler and development environment), and the continuous integration tool we will use to build, test, and deploy our application.

6.2 Language

6.2.1 Plain ES2018 JavaScript

One of the requirements for our project is to have the Virtual Reality experience hosted within modern web browsers. Since all modern web browsers use JavaScript, and the WebXR API is implemented in JavaScript, we will need to use JavaScript or some derivation of it to create our project.

Plain JavaScript does not have any compile-time type checking. The only type checking it does is if there is an error. As with other dynamically typed language, plain JavaScript allows developers to rapidly create projects and get products out the door quickly. However, although it is easy to quickly create projects in plain JavaScript, it is much more challenging to write code that is maintainable over the long run. This is because without static types, much more documentation must be written for more than one person to write code in the same code base, since it all must interact with itself.

Every year, the ECMA standards organization makes updates to the JavaScript standard that is implemented in each browser. However, it takes time for browsers to implement these standards, and for users to actually update their browser. Therefore, if developers want to use the latest standard, they will need to use a preprocessor such as Babel to create polyfills for them so that user on older browsers can still use all the functionalities of the website [**?**].

6.2.2 TypeScript

TypeScript was created by Microsoft as a way to provide static typing for JavaScript. It does this by forcing users to add type definitions, either explicitly or implicitly (if possible in the situation) to all of their variables, functions, classes, and objects. It also adds interfaces, which are used to tell the compiler (and the developer) about the shape of JavaScript objects and classes, including whether properties are optional or mandatory, and what the type of each of those properties can be [**?**].

TypeScript does add some overhead to the developer. First, the code must be compiled into regular JavaScript to be run in the browser. Also, each library that is referenced within a TypeScript file must have a type definitions file. This means that for any smaller libraries that don't have type definitions, developers won't be able to get the benefits of typescript when interacting with that library [**?**].

6.2.3 Flow

Flow is similar to TypeScript in that both tools are used to provide static types to JavaScript. While TypeScript was made and is supported by Microsoft, Flow is made by Facebook. The type annotation system that Flow uses is very similar to TypeScript, but there are a few differences between the tools as well. Flow will build a deeper understanding of the code and does interprocedural analysis on the code. TypeScript, on the other hand, helps developers by providing intellisense (auto complete), code navigation, and refactoring [**?**].

6.3 Asset Bundlers

6.3.1 Webpack

Webpack has been the most used bundler for the past few years, which means that there is widespread support for it, as well as plenty of tutorials on configuring it. In the past, it has had a bad reputation for being complicated to configure, but in more recent versions, it is significantly easier to have work right out of the box. However, there is configuration required for features such as development server and code splitting. One thing Webpack excels at is allowing the developer full control over their build process, including specifying specific tools for specific filetypes (JavaScript, CSS, Images, etc.) [**?**].

6.3.2 Parcel

Parcel is a newer asset bundler that requires almost no configuration. It works by having the user directly include the entry file (that still needs to be compiled) in their HTML file, and then it resolves the dependency tree from there and outputs a modified HTML file, along with other dependencies. This could beneficial for the WebXR demo project, since the team wouldn't have to worry about spending the time to configure Webpack. However, it doesn't have as widespread of support since it is a newer package. It does have out of the box support for TypeScript and Rust, along with many other languages [**?**].

6.3.3 Rollup

Rollup is a recent addition to the Asset Bundler scene. It requires some configuration, and has works very similar to Webpack in that users specify an entry point, a destination, and an array of plugins that will be applied to the assets. However, Rollup's configuration is a little bit easier to use, but has less documentation than Webpack, especially when it comes to third party tutorial articles. Rollup also seemed to be much quicker than Webpack and Parcel, from tests done by X-Team in their article titled Rollup Webpack Parcel Comparison [**?**].

6.4 Continuous Integration

6.4.1 Circle CI

Circle CI (Continuous Integration) is a tool that automatically runs unit tests on developers' code whenever a commit is pushed to a branch on a source control server. In our case, we will be using GitHub. It is compatible with Node.js (which will help for running unit tests of client-side code), along with many other languages. It works by listening for notifications from GitHub (or other source control providers), then pulling the branch into their cloud cluster and preforming a set of user-specified tasks. Typically, these tasks include building, testing, and publishing. These tasks are configured within a YAML config file [**?**]. Circle CI is free for open source and closed source projects, but one can pay to upgrade the resources the build is given.

Using CI such as Circle CI allows the developer to ensure code quality and that the tests are all passed. Developers can also create releases from within continuous integration, which means that the developer doesn't have to do it themselves manually [**?**].

6.4.2 Travis CI

Travis CI is very similar to Circle CI in that users can specify a YAML config with the tasks they want to execute on their code (typically building, testing, and publishing), and then have Travis run them after every source control push or merge. Although Travis CI has a different config schema, they both allow users to do similar tasks the same way. Both also run in the cloud, rather than on one's local computer or build server so there is little setup required. There are a few things that Travis supports that Circle CI does not. Travis CI has build matrixes, which means that one can specify different environments (such as language versions and operating systems) to build and test their code on [**?**]. Travis is free for open source projects.

6.4.3 Jenkins

Jenkins is one of the most well-known continuous integration systems in existance, partly because it has been around for a long time. Unlike Travis CI and Circle CI, it is a program that developers download to a build machine. This means that to use Jenkins, we would have to have our own build machine to run it on, then configure it to hook into GitHub. This could be a benefit for closed source projects that do not want to risk their source getting out into the world, but for open source projects, this is just another cost. Jenkins is also known to be challenging and time consuming to configure, so there is a time cost associated with using it as well. Other than those costs, it is very configurable and has a plugin system built into the tool. The software itself is also free [**?**].

6.5 Conclusion

One of the requirements of our project is for it to be able to run in the browser. Hence, the language we will need to use must compile to JavaScript. Three options for this are plain JavaScript (using the latetest specification so we get the most features), TypeScript, or Flow. My personal recommendation is to use TypeScript since there is widespread support for it, and it makes the code more maintainable in the long run. That will be beneficial to us since our project will be spanning the entire school year.

In terms of Asset Bundlers, there are three leading options in the community currently - Webpack, Parcel, and Rollup. I believe we should try Parcel first to see if it works for us, since there is no configuration and everything we need should work right out of the box. If Parcel does not give us enough granularity of control, My second recommendation would be Webpack because it is widely used and has great documentation.

For Continuous Integration, there are several options. I narrowed it down to three popular choices: Circle CI, Travis CI, and Jenkins. Circle CI and Travis are both reasonable options for us because they come at no cost for open source projects and they do not require the developer to supply the build machine. They are also highly similar, and we do not need to worry about the code running on multiple versions of node, since we will be putting our code through Babel to compile it to ES5. Jenkins, on the other hand, is free for the software but requires a build machine, which will increase our costs. Therefore, it is not the best option for an open source project.

7 Tech Review - Josh Strozzi

7.1 Introduction

The goal of this project is to provide students a fun and easy mode of learning using AR. Our application will focus on kids from kindergarten up through high school. Using AR for some important material in classes will free up a lot of space due to lack of clutter and provide more time for education because there is not much to the setup for AR.

My focus will be the implementations of our the AR frameworks, platforms and APIs that enable movement tracking for rending 3D scene in a real environment in real time. I'll also be looking into potential web server frameworks for connecting to the host server that will be running these AR experience, as well as the host server method we will be using to run those experiences.

7.2 Web Framework / API

7.2.1 WebXR

WebXR is a Mozilla developed JavaScript API built for enabling people to create and distribute virtual reality (VR) and augmented reality (AR) projects across the web [1]. This API supports the rendering of 3D scenes or adding graphically rendered objects to the real world, all while managing user motion. This is accomplished through four main tasks handled through the API: Find compatible XR device, render the 3D scene to the device at an appropriate frame rate, create and manage motion vectors representing movements of the user through input mediums, and in the case of VR only mirror the output to a 2D display [2]. WebXR can be used with WebGL libraries in rendering those 3D scenes and setting up the virtual camera that will be used to navigate said scene. WebXR, although the most ubiquitously used web based XR API used right now, is currently only fully supported on Chrome and Chrome Android.

7.2.2 8th Wall Web

8th Wall Web is a platform that can be used for developing web based AR experiences, bolstering it's own extra features that could make creating, prototyping and launching much easier and quicker. Also, 8th wall is supported on more browsers than WebXR since the AR content runs directly on the hosting website.[3] 8th is built using the SLAM engine, Simultaneous Localization and Mapping engine, which is an indoor mapping technology that 8th Wall Web has optimized for AR use on mobile browsers. The platform has a cloud editor for quick implementation, prototyping and collaboration, as well as built-in hosting, allowing for immediate publication and deployment of projects hosted on their own secure network. Just like WebXR, 8th Wall Web can be integrated into 3D javascript frameworks. Although this is a very powerful tool and environment for multiple aspects of this project, WebXR is the API our client has chosen for us to use and is what we will ultimately be using.

7.2.3 AR.js

Lastly, AR.js is a javascript library that enable developers to make augmented reality web apps and fast. It's cross browser, desktop and mobile, and works by wrapping together different frameworks into one VR solution, those frameworks being a-frame and three.js [4]. AR.js is a marker-based AR library, meaning in order for the AR scene to be chosen, rendered and tracked, a QR code must be printed and placed down on the chosen reference surface. These markers are at most 16x16 in resolution, square shaped, and only use black and light gray without any symbols like numbers, letters or what have you. This constraint, needing to use markers, goes against the design goals of this project so it is likely we will be avoiding all libraries having such requirements.

7.3 Web Server

7.3.1 Node.js

Node.js is a JavaScript runtime environment built on Chrome's V8 JavaScript engine for various platforms (Windows, Linus, Unix, etc.) designed to build scalable network applications [5]. It's not built to be a server specifically, but modules can be included in it to make it into one. Using nodes built in https module, developers can create a server that serves as a middle-ware from a web page users visit to access the web app to the actual files needed for said web app to run. This is super useful as node serves as our medium for deploying the app to users wanting to user that visit our site, and there's plenty of documentation on node as it is very widely implemented.

7.3.2 Apache

Apache is a freely-available implementation of an HTTP web server. Apache HTTP Server enables developers to serve content on the web, like any server, and powers around 46 percent of all websites. It establishes a connection between the server holding content and the browser while transferring files between them. It is highly customizable, being module-based similarly to node, having modules for security, caching, etc., and works is cross-platform. Advantages include being stable, actively supported, flexible, easy to implement, etc. However, it can suffer performance issues during heavy traffic, and too many configuration options may lead to security vulnerabilities.

7.3.3 NGINX

Often brought up in the same context as Apache, NGINX (pronounced engine-ex), is an open source web server software for HTTP web serving, among other functions. As Apache and NGINX do mostly the same thing, it would serve better to describe how it is they differ. Both are indeed cross-platform solutions, however NGINX struggle to deliver the same performance on Windows than other platforms. As far as user-support, NGINX receives better active support from its creator company. Performance-wise, the decision between the two leans towards NGINX. NGINX can run an exuberant number of static content connections much faster than Apache using little memory, but when comparing their performance for dynamic content they do equally well[7].

7.4 Server Host

7.4.1 Docker

Docker is an open source tool for making, deploying, and running applications by using containers, containers being a medium by which an applications are packaged up entirely, sources, libraries, dependencies, everything regarding the app not already on the host computer [8]. This container of the app ensures the apps ability to run on any other

Linux machine, even if said machine doesn't hold any of the dependencies the app needs. This could be used in tandem with cloud web hosting to distribute multiple instances of the running app [9]. In the case of using Docker, containers and cloud hosting go hand in hand. Cloud web hosting is a method of storing and sharing multiple resources among a network of servers that can dynamically share those resources in the event they need the extra power. Docker has a simple distribution hub for this exact use case, Docker Cloud hub namely, for easy distribution to any machine. Cloud web hosting is very powerful, but the costs of leasing resources of a server farm vary and they are subject to traffic spikes.

7.4.2 Shared Web Server

A shared web server is a server solution wherein multiple web clients utilize a shared single server that is maintained by the hosting service. Each web client website will be allocated a portion of the shared servers resources, like storage, RAM, processing power and bandwidth [10]. Disadvantages involve the fact that sharing resources among multiple web clients means our web page is more susceptible to traffic spikes than some other options, affecting the user experience during those times, as well as not being as easily scale-able as some other hosting options, like cloud hosting. However, the main advantage of this would be how much cheaper it is than other options, as well as being a good fit for our web apps use case, as we have one web page with one web app and few users will access it at any given time.

7.4.3 Single Dedicated Server

Single dedicated server hosting is a service wherein a whole server unit is allocated for a single client, meaning the customer renting the server to host their website or web app shares none of the servers resources with another customer, they have full control. This type of hosting solution is very reliable as there is never an instance where multiple customer processes are vying for the same resources. Also, dedicated servers have unparalleled speed and great bandwidth for whatever you may need to use it for [11]. Many game developers make use of this hosting for online multiplayer as it is currently the best solution for low-latency connection between clients for gaming. Dedicated servers lower security risk, customize-able static resources, and 100-percent up time, but the cost can be too much for most. This could potentially be overkill for our purposes as we have one web page for a single web app that very few users at any given time will be accessing.

7.5 Conclusion

For our WebXR project to deliver the type of experience we have planned to create, we need to use effective AR API, efficient Web server software solutions, and possibly powerful web hosting technologies. The objective is to run a robust AR experience for users to learn from and that requires supported API that manage users movement and translates the 3D objects accordingly, and quick transfer of the assets for said experience so anyone can use their phone to access it. We will be using WebXR to achieve our goal on the AR implementation side since its the premiere technology, being supported by chrome and chosen by our client, and we'll be using Node.js to deliver those files to the phones that try to access the demo as it's very modular and lightweight. As for Web hosting, we need somewhere to store said assets we are transferring with the Node.js server software, and for this task, given our needs, a shared web server is most likely our best option, as it's more affordable and we don't need to large enough volumes of assets to require the storage of a dedicated server, nor will there be so much traffic to the point we require that level of bandwidth.

7.6 References

[1] Bergstrom, Lars. “New API to Bring Augmented Reality to the Web – Mozilla Hacks - the Web Developer Blog.” Mozilla Hacks – the Web Developer Blog, 10 Sept. 2018, hacks.mozilla.org/2018/09/webxr/.

[2] “WebXR Device API.” MDN Web Docs, developer.mozilla.org/en-US/docs/Web/API/WebXR Device API.

[3] “8th Wall Documentation.” 8th Wall Documentation, www.8thwall.com/docs/web/introduction.

[4] Carpignoli, Nicolo. “AR.js - The Simplest Way to Get Cross-Browser Augmented Reality on the Web.” Medium, ‘The Startup, 2 Aug. 2019, medium.com/swlh/ar-js-the-simplest-way-to-get-cross-browser-augmented-reality-on-theweb-10cbc721debc.

[5] “About.” Node.js, Node.js Foundation, nodejs.org/en/about/.

[6] https://www.hostinger.com/tutorials/what-is-apache

[7] https://www.hostinger.com/tutorials/what-is-nginx

[8] “What Is Docker?” Opensource.com, opensource.com/resources/what-docker.

[9] https://www.docker.com/blog/federated-application-management-in-docker-ee/

[10] https://www.techradar.com/web-hosting/what-are-the-different-types-of-web-hosting [11] https://www.atlantic.net/hipaacom
dedicated-server-hosting/what-is-a-dedicated-server/

8 Blog Posts

8.1 Andrew Snow

8.1.1 Fall 2019

Fall	Blog Post
Week 4	This week the most important thing we ended up doing was visiting with Alexis face to face. Together we were able to brain storm and really get an idea of what we were going to develop. Now that we have a better idea we know we can come out with a filled out Requirements Document.
Week 5	This week as a team we were able to concrete more ideas about our project after having the initial visit with Alexis at Intel. We focused on what technology's would be used and divided up the research to those technology's which in turn would be our tech review. Once we get the tech review finished we'll be moving on to the design of our project.
Week 6	Progress: This week we busted out the tech review. This was an individual assignment were we talk about specific components of our project. I've gotten some feedback and have finished the final draft. Otherwise as a team we were able to get a look for how we want application to look. I think we are all on the same page which will help with the design document that is due next week. Problem: There were not many problems. Next week will be a different story as I have an important midterm I need to study for. Plans: As it stands we need to get the design document done by Tuesday. We'll be working pretty hard to get that done. Once the draft is done well be able to work on filling it out more fully with all the details we need.
Week 7	Progress: This week we were able to get the tech review finished. We then started the design draft. Talking with Alexis we elaborated on the ideas we had and agreed on the basic design for our project. Problem: The problem this week will be getting the design doc in order. Plans: The big thing is to fill up the design doc with more details. I will be doing more research and adding what I find to the doc.
Week 8	Progress: Finished up the design document. We have submitted it into canvas and will be waiting for Alexis to review it and provide feedback. Problems: No new problems. Plans: Once Alexis looks over our document and reviews it, we will apply the necessary changes and then start looking on starting small components of our project. I plan to start looking at just getting a sphere to rotate around a center point in a web page.
Week 9	Progress: Didn't do anything really due to Thanksgiving. We did get word from Alexis that he liked what the design doc looked like. Problems: No new problems to report Plans: We will talk to Alexis to finalize the design doc and get started on finalizing all the pieces to the entire document.

TABLE 2: Fall 2019 Blog Post Andrew Snow

8.1.2 Winter 2020

Winter	Blog Post
Week 1	Progress: Meet with the teem and got everything organized and got together on what we should start working on. Problems: No foreseen problems yet. Plans: Will begin working on the solar system part of our project.
Week 2	Progress: Through out the week I worked on getting the solar system components working together, and making it so the camera could be controlled to view all the elements. Problems: The tricky part is I cant do to much till we have the AR component working. For example the camera will be very different when AR is implemented . Plans: I will be helping many of my team members to get them going and help them with any problems that arise. I will also tinker with Ideas within the solar system.
Week 3	Progress: I have merged with Cody's code, got a couple more camera angles to work, and added a couple event handlers. Problems: No major problems other then knowing how some of these dependencies we want to implement work. Plans: This week I am planning to host the web-page on github.io and make sure everything still works. I will also be adding a couple dependencies that our client suggested to implement.
Week 5	Progress: Continent selection is now implemented and working. Also got a test with roll-up and service worker going. Problems: No problems yet. Problems may occur with roll-up and service worker Plans: Need to merge with Josh and Cody and will work on getting the roll-up and service worker packages to work in the main app. Also will start working on poster draft
Week 6	Progress: Been working on getting all the teams code together and working. Also have been working on the poster. Problems: Having trouble with the AR implantation. Plans: Gonna be helping with the AR implantation and getting the poster finished.
Week 7	Progress: AR is now working and we got our presentation done. Problems: The only major problem we have is with the adjustment to AR. Currently it seems only the UI has been effected but as we continue to develop more problems may occur. Plans: Will continue to work on the AR integration
Week 8	Progress: This week I continued to get our scene fully implemented with the AR. I got all the planets rotating properly and now I just got raycasting to work so we now can click on the individual planets. Problems: No forseen problems at this time Plans: I need to create an end event for the XRsession and get the scene to move when a planet it picked on

Winter	Blog Post
Week 9	Progress: Was able to get the solar system click events all functional except for the moon and the sun itself. Problems: No major problems at this time. If something will come up it is mostly due to lack of experience and time in these areas. Plans: This coming week I will be help out a lot to progress the project more. I will also be looking at working in the sun and moon click events once I've talked to the team about what could be done.
Week 10	Progress: Got a pull request going with our client which will catch him up to our current beta build. Before doing all that just worked on getting the solar system to work a bit smother when viewing a planet up close. Problems: No major problems. Just need to get everything together before the due date. Plans: Working on the video and getting the final paper together

TABLE 3: Winter 2019 Blog Post Andrew Snow

8.2 Cody McKenzie

8.2.1 Fall 2019

Weeks	Progress	Problems	Plans
Week 4	We have met with our client Alexis Menard and plan to go meet him in person at Intel on October 25. We have also got our problem statement second draft going and finished the requirements rough draft document.	Lack of knowledge at this point for the requirements document is the biggest thing and working around everyone's schedules.	Meet Alexis Friday October 25, edit our documents as we develop a better understanding of our project and goals and meet weekly with each other, Alexis and the TA (Richard).
Week 5	We have met with our client Alexis Menard in person at Intel. Now are starting our Requirements second draft because we have an idea on what we will be doing for our app.	There aren't any problems right now as we are just working to get the documents together and going.	We plan to start looking into the tech we will need and maybe any assets we will need to purchase license for in order to use for our app.

Weeks	Progress	Problems	Plans
Week 6	Finished my Tech Review, waiting on some Peer Reviews so I can make sure it looks good. Flushed out our general design for our Ar app and how it will function on the phone for each phase.	There are a lot of midterms happening right now which makes meeting up difficult which in turn makes the work harder since we aren't able to discuss things as well.	I plan to finish up my tech review final edition, begin the design documentation and discuss the application with the team and Alexis some more.
Week 7	Submitted the Final Tech Review, discussed the Tech Reviews with Alexis and brought up the Design Document in our conference which he gave us some good advice on and would like to review when we feel it is prepped enough.	Lack of time due to work and other classes but other than that no problems this week.	I plan to fix up the Design document up enough for it to look good and able for Alexis to review it. I also plan to work with the group to see how we want our world map to work as that is the last asset we have to figure out.
Week 8	Fixed up the Design Document and prepared it for Alexis to review and sign.	Meetings are still difficult to setup besides on Wednesdays.	Have Alexis review and communicate with us on the Design Document.
Week 9	Letting Alexis review the documents that we need him to sign.	Thanksgiving week so everyone will be pretty busy this week.	Work on our Progress document and communicate with Alexis if we or he has any questions.

TABLE 4: Fall 2019 Blog Posts

8.2.2 *Winter*

Weeks	Progress	Problems	Plans
Week 1	Working on our AR application and getting our scenery setup as well as the assets. Had a meeting with Alexis to discuss what we have accomplished so far and what we plan to work on next which will be functionality.	Figuring out how our assets are loading in and hooking the astronaut up to the camera is proving to be a bit of a challenge due to point of origin being in weird locations on our objects.	Continue working on the application and flush out our functionality and basic setup so we can begin working on the harder parts.

Weeks	Progress	Problems	Plans
Week 2	Continuing to work on our AR application and getting our scenery setup, the assets are pretty much all added in now. Had a meeting with Alexis to discuss what we have accomplished so far and what we plan to work on next which will be functionality.	Getting the astronaut to go to the position we require him to be in so we can start toying with his transitions.	Get the astronaut to transition properly, get the UI started and do more research about implementing everything to the AR state.
Week 3	Got the astronaut stuff finished, now we just need to implement all the stuff to our solar system portion.	No problems this time around.	Going to start on the text boxes that the astronaut will have bound to him for facts about the planets and the continents of Earth.
Week 5	Working on the alpha version of our app, got the text box implemented and saying facts about each planet. Also working on the poster, hashed out parts of the poster to each member.	The only problem I see coming into play here would be time, we are still working on fixing some big parts but we are real close to getting it done as long as no new snags come up.	Get the poster done and the alpha version of our app finished up.
Week 6	Working on the alpha version of the app and editing the poster to turn it in tonight.	Getting time to work on these items together as a group.	Get the poster done and the alpha version of our app finished up.
Week 7	Got AR working and got the progress presentation taken care of. Currently setting up 3D text to be placed on each continent of Earth so we can have the names of the continents.	We lost a lot of UI functionality when implementing AR so now we have a bit of rework to do to get UI back.	Fix the UI and textboxes and get the continent names on Earth properly.
Week 8	Got AR working for the whole solar system and now just editing the controls and other parts such as UI.	TIME! So much to work on in and out of the class which is making this quite stressful.	Work on the continents and spice up the homepage some more so it can explain what the app is and talk about controls/functionality.

Weeks	Progress	Problems	Plans
Week 9	Got a great start to the homepage, almost done with it and started the quiz page.	Determining how to make things flow properly and smoothly, it won't be hard just time consuming and a bit tricky to figure some parts out.	Finish up the homepage completely and then hopefully get the quiz finished or at least most of the way finished.
Week 10	Homepage and quiz page officially finished, beta wise anyway. We will definitely be editing parts of them later but it has the functionality we need.	We have an issue with the AR application not going directly to the quiz, instead it makes a brief stop at the homepage which does not look very good.	Help teammate tweak whatever we need to in order for our Beta to be completely finished by the end of this week.

TABLE 5: Winter 2020 Blog Posts

8.3 Martin Nguyen

8.3.1 Fall 2019

Weeks	Progress	Problems	Plans
Week 4	We met up with our client at Intel in person on Friday and finalized what kind of AR experience we want to create. We have decided to do a Solar System Model in AR. It will focus on the Sun, Earth, and Moon System. We plan to cover topics such as the day/night cycle, lunar phases, seasons, and the continents/countries. Our client, Alexis, also showed us some libraries and APIs we can could use to implement our website. We got an extension on our Requirements Document, and we look to turn it it on Sunday.	We are still having trouble with LaTeX. We can make it function as we want for the most part, but LaTeX is very picky about how it wants us to do do things and making it work the way we want it to is hard. We'll keep fiddling with it.	We plan to start figuring out what specific systems and libraries we plan to use for our project. The previous year's group used a lot of libraries that we would like to look into.

Weeks	Progress	Problems	Plans
Week 5	We finished the second draft of the requirements and got feedback from our client and our TA. We need to work on the professionalism and details of our requirements document. We also divided our project into 9 topic/technologies we can research for the tech review. Last Friday, we met up with our client at Intel in person and finalized what kind of AR experience we will create. We decided on making a Solar System/Earth, Moon, Sun AR experience. We also discussed potential technologies and libraries we could use in our project.	We've been having issues finding the LaTeX template and getting our documents to be in the correct format, but today we found the template in canvas. The template is in a link in the other resources page on canvas. Before we were only searching through the files on canvas and could never find it. We are having an issue finding enough technologies for all 4 of us to have 3 different technologies to do research on. We have decided on either letting two people cover one really crucial technology separately, or have one member only research 2 technologies but write 1000 words for one really important technology.	This weekend I plan on doing research and writing up my section of the tech review. I also plan on updating the our requirement's document with the new template and also improve the professionalism of the paper.
Week 6	I did the peer review and worked on improving my tech review. I am still not confident in my technical writing. I feels like I am using too many words to say too little.	There are a lot of sections for the Design document and figuring out what we should write for each viewpoint and how each viewpoint differs from the one another is proving to be a challenge.	I plan on working on the first draft of the design document this throughout this weekend and working on improving past documents.
Week 7	We got the first draft of the design document done. We have all the viewpoints we want to talk about decided on as well.	I don't know exactly what we should talk about for each viewpoint. I also don't know if we have included all the details of the entire project with the current viewpoints we have selected.	I plan on working on improving my section of the design document. I also want to flesh out a single format/style we should follow for the design document.

Weeks	Progress	Problems	Plans
Week 8	We finished working on the finishing touches for the design document.	No problems showed up this week. Everything went as planned.	I plan on refining the requirements document now that we know more about what our client is asking of us.
Week 9	Not much progress was made this week with thanksgiving taking place, but the requirements document was looked at. I don't know if we should redo our problem statement, since I am unsure of it will be included in the final document.	There we're no problems with week. However, we had some miscommunication about our TA meeting this week where we didn't show up thinking the meeting was cancelled when in fact it wasn't.	I plan on finalizing all the documentation and compiling it all into one document.

TABLE 6: Fall 2019 Blog Posts for Martin Nguyen

8.3.2 Winter 2020

Weeks	Progress	Problems	Plans
Week 1	N.A.	N.A.	N.A.
Week 2	I've been working on getting WebXR to to work on our phones. It turns out that although Chrome version 79 supports WebXR, there is a bug that stops it from working, so we had to switch to chrome dev or beta.	We had problems with getting the astronaut to show up correctly at the right distance at the right angle in front of the camera.	This week I plan on getting the AR and hittest to work with the 3D scene that my group has made of the solar system.
Week 3	I have continued working on getting the AR portion of the application working. It has been slow, but steady progress.	Our client showed some new technologies/libraries he wants us to include in our project. We will have to learn these new libraries and incorporate them into our code base. One of these technologies is a library for caching files, so that our users will only have to download the models once to use our application multiple times.	We plan on merging all of our code into the master branch in GitHub. The code in this state should be set up to work as a node package and will use github.io for web hosting.
Week 4	N.A.	N.A.	N.A.
Week 5	I've been working on getting the hit test to work for WebXR and AR.	No problems have really popped, everything is going smoothly.	We plan on merging all the code together for an alpha demo, and plan to get the first draft of the poster done.
Week 6	I continued debugging the AR functionality, and worked on my part of the poster.	AR is proving to be more difficult than we thought it would be. Also the testing pipeline we have for testing the website on mobile is becoming cumbersome, because we are using github.io to serve the files.	I plan on preparing for the presentation we have to do. I also plan on further debugging AR functionality, so that we can demo it during the presentation.

Weeks	Progress	Problems	Plans
Week 7	We finished doing the design review.	We learned that all our HTML code that we planned to use as a UI and as click listeners does not work when WebXR is active, so we will need to complete redo all of it without using HTML DOM elements.	I plan on refactoring the code and working on a way to remove the node_modules from our git repository.
Week 8	I continued work on refactoring the code base.	When we attempted to remove the node_modules folder from history, the repo got bigger instead.	I plan on finishing refactoring the code and adding a linter.
Week 9	I finished refactoring the code into functions to make it more readable.	Our git repo is now massive. It is 500+ MB large. It makes cloning the repo really slow.	I plan to refactor the AR branch to use LitElement and ESLint.

TABLE 7: Fall 2019 Blog Posts for Martin Nguyen

8.4 Josh Strozzi

8.4.1 Fall 2019

Weeks	Progress	Problems	Plans
Week 4	As of today we have met with our client face to face and decided together a course of action for the coming months. We've gone over in greater depth the exact technology we will be using as well as the product we will be developing using said technology to achieve a specific experience.	Due to us having to meet with our client the same day that our requirements document is due meant we had to talk with our project liaison(?) to move our due date for that assignment back to the Sunday after since we only knew the details we need for that assignment as of that day.	We will be working on class assignments for the remainder of the term and trying to learn the technology we need for the assignment until a time in which we can fully delve into taking on the project fully, sometime around the beginning of Winter break, or earlier if we can manage.

Weeks	Progress	Problems	Plans
Week 5	We got our Requirements doc rough draft in on time, and we have separated out sections among the group for the technical review which we'll be working on this weekend. We also got feedback from our client about the requirements doc and what needs to be changed.	Not problems to report this week. Things went pretty smoothly.	Plans are to work on the technical review this weekend, researching all the tech we will be using and options within each. Should be a lot of work, but very important work since this is probably when we learn a good chunk about how to do each part of our project.
Week 6	Worked on the Tech review assignment and worked on further design details about our app as a team after our weekly meeting with our project adviser.	When working on the tech review, I didn't really understand how we were going to be potentially using some of the technologies within our projects, so I didn't have direction for how to research some of the technologies and ended up writing sections that I'm not proud of and don't represent the direction of our project accurately.	I will be re-researching sections that I didn't do well on the first time and writing a better final draft.
Week 7	We finished our final draft of the tech review and turned it in, defined in a greater detail the design of our doc and finished the first draft of our design document and discussed the design with our client as well as a UI framework that we could use that'd be much better than those we found on our own.	No problems to speak of this week.	We will be working on our next draft of the design document.
Week 8	There wasn't much to work on this week. We met with our adviser and discussed what we'd be working on moving forward.	There were no problems this week since nothing was needing done this week.	In the next week we will be working on our next drafts of the design doc.
Week 9	N/A	N/A	N/A

TABLE 8: Fall 2019 Blog Posts for Josh Strozzi

8.4.2 Winter 2020

Weeks	Progress	Problems	Plans
Week 1	This weeks progress included figuring out weekly meeting times as a team, with our client and with our adviser. Also one of our team members has gotten started on the more simple aspects of the assignment.	No problems thus far. Just figured out our schedules together. It all went pretty seamlessly.	Get together this Sunday to hammer out in a more detailed manner our plan for completing this project over the coming weeks. Who does what, and the sort. We'll also get started on pieces of the project.
Week 2	We decided on who will be working on what over the next couple weeks. I will be doing UI for the app.	Problems are that some people have aspects of their part of the project that are waiting on other people. Also one of the group members is vocal about not wanting us to merge to the master branch but not giving us enough information about why.	Plans are to bunker down and get the work we need to get done done so we can launch our alpha build. Just have to research how to do UI and then implement it on my branch of the project.
Week 3	The group as a whole is currently making a lot of progress towards the alpha release of the project.	I am seriously struggling with getting started with my part of the assignment. I've been having serious paralyzing fear of starting my part and I'm falling far behind the team. I'm trying to get started this weekend, but I'm really worried I won't be able to get anything done and what that might mean.	I'm going to write SOMETHING no matter what, but I'm going to meet with one of the group members and try to work through it with him.
Week 4	N.A.	N.A.	N.A.
Week 5	This week was mostly light, since last week I completed what I could on UI at this time, there wasn't much left to work on. We did meet to discuss the creation of the first draft of our post and we gave everyone a portion of the poster to work on.	No problems this week. Things went as swimmingly as possible.	Complete the rough draft of the poster and begin working on the phases of the moon portion of the project, which was recently appointed to me.

Weeks	Progress	Problems	Plans
Week 6	This week, we've mainly been waiting for the AR implementation, I believe, but we made our poster draft.	I got sick this last week and couldn't work on much as I was dizzy in bed coughing a lot.	I believe we are, as a team, going to meet up this weekend and go over the code base to prepare for our presentation and for further work on said code base.
Week 7	N/A	N/A	N/A
Week 8	The AR aspect of the app and the 3D scene have been successfully merged.	Realizing our UI has to now be entirely recreated from scratch was a bit rough, but we are currently looking into way to remedy this.	We found a ThreeJS UI module that we are going to be experimenting with this weekend called Poki.
Week 9	We figured out how to implement a UI within a ThreeJS scene without using any external modules	We thought we could use an NPM module but what we were looking at on github turned out to be a separate product from the one on NPM, same company, but they don't actually include the UI they built in the module.	Finish the UI to the point of feature completeness within the scene, then work on making it pretty. Good thing about the old UI from back when we could use html, we can still use some of that code so not all of my previous work is gone, so that's good.

TABLE 9: Fall 2019 Blog Posts for Josh Strozzi

9 Poster

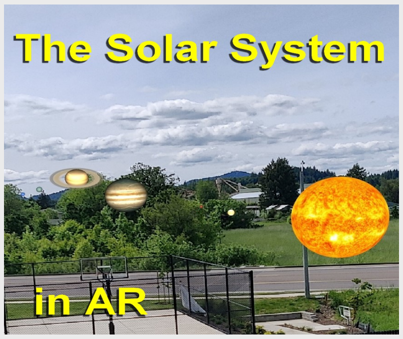
COLLEGE OF ENGINEERING

Electrical Engineering and Computer Science

CS 19

ARTT - Augmented Reality Teaching Technologies

A new interactive learning experience available to anyone with a smartphone!



Purpose

We are looking to give students a view on how our solar system works. From the beginning of our application students will get a scale view of the sun and the planets in orbit. The student then can zoom in on the earth to get an idea of how the day/night cycle works as the earth rotates on its axis.

What is Augmented Reality?

Augmented reality is an interactive experience with the real world environment that is enhanced with computer generated objects, sounds, and effects.

AR as a teaching platform

Visualizing many subjects properly can require materials that teacher potentially don't have to the resources for. AR powered teaching modules are capable of equipping all classrooms with the tools to visualize any subject too grand in scale for teachers to properly describe or display themselves.

Oregon State University

Features:

- View the solar system in action as the planets orbit around the sun.
- View each planet individually and learn unique facts about it.
- Visit the earth and learn about the 7 continents.
- Learn about the day/night cycle.
- Watch the moon lunar phases as it rotates around the earth.
- Adjust the speed at which the solar system works.

WebXR

- WebXR is an API that connects web applications running on browsers with the hardware on the device to display Virtual Reality and Augmented Reality to the user.
- Using the WebXR API requires three steps:
  1. Check if the browser and device support immersive content with WebXR.
  2. If the device supports WebXR allow the user to start the immersive content.
  3. Create an XRSession.
  4. Replace the normal renderer() function from the WebGL library beign used with the requestAnimationFrame() function.

From left to right:

Cody McKenzie, Martin Nguyen, Andrew Snow and Joshua Strozzi

Technology

- WebXR - An API that allows developers to program AR and VR experiences through the web.
- Three.js - A widely used web-based library.
- Service Worker - A programmable network proxy that allows control of how network requests are handled on pages.
- Roll Up - A module bundler for JavaScript that will compile small parts of code inot larger or more complex parts of code in order to help build libraries or applications.
- Node.js - An open source and cross-platform JavaScript runtime environment that is able to execute the code outside of a web browser.

10 Project Documentation

The application was built using the WebXR framework which means only curtain devices will be able to run our project. The following web page can help indicate if the users device is compatible:

https://developers.google.com/ar/discover/supported-devices

Since our application is web based the user will simply need to run the following link on a compatible device to experience our application.

https://osu-2019-capstone-cs19.github.io/Educational-AR-WebXR-App/dist

11 Recommended Technical Resources for Learning More

The most useful resources for this project was the online documentation for the APIs and libraries we used to implement our application. Our client, Alexis Menard, was very helpful with the process of choosing the which technologies and libraries we would use for our application.

List of Useful Site Resources:

- W3 WebXR Device API Docementation: https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API
- MDN WebXR Device API Documentaion: https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API
- three.js Documentation: https://threejs.org/
- Rollup Documentation: https://rollupjs.org/guide/en/
- MDN Service Worker API: https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API
- Node.js Documentation: https://nodejs.org/docs/latest-v13.x/api/
- NASA Models: https://github.com/nasa/NASA-3D-Resources

12 Conclusion

12.1 Andrew Snow's Reflection

Q: What technical information did you learn?
A: Throughout the development of this project I have learned a great deal about working in 3D environments, collaborating ideas with other team members, and working with the webXR frame work to get AR to work. It was a joy to work on the project and watch it slowly shape into what we have now; a fun, educational learning experience. All lot of what we accomplished were due to hard work and dedication towards the project.

Q: What non-technical information did you learn?
A: Most of the work we did was technical work. For the non-technical work we worked on a couple documents that in my opinion didn't help our project.

Q: What have you learned about project work?

A: Big projects like ours take a great deal of work and effort. I found an important aspect of it was just to keep at it and get a little done everyday. By doing that I was able to get the parts I was working on done quickly and efficiently.

Q: What have you learned about project management?
A: One important aspect of project management is establishing a good plan for when parts are going to be finished. Communication is key for keeping everything moving smoothly.

Q: What have you learned about working in teams?
A: Having a good team helps makes the workflow go quickly. Being able to bounce ideas off of each other and establish a good plan helped a lot to make our project what it is.

Q: If you could do it all over, what would you do differently?
A: A lot of what we did took a lot of trying and failing. If we were to do this differently I feel like we would ultimately end up with a similar result due to the learning curve we all had to go through.

12.2 Cody McKenzie's Reflection

Q: What technical information did you learn?
A: I learned some technical information but already knew a bit thanks to previous jobs and classes. I learned how to write for certain perspectives better by practicing through the documents we have had in this class. I also learned how to code in WebXR which was interesting to learn and see what all breaks and has to have rework from javascript to become augmented reality.

Q: What non-technical information did you learn?
A: I did not learn much non-technical information because most of our assignments and work on this project were quite technical.

Q: What have you learned about project work?
A: I learned that project work can be really difficult work but is also incredibly rewarding once we reach the finish line because seeing the final version of our project and seeing most everything functioning the way we wanted it to is amazing.

Q: What have you learned about project management?
A: I have learned project management can save a lot of time and effort/rework as you plan everything out before beginning on the project. It is also useful to think about some fire that may start later down the road and have contingencies for them prepared ahead of time.

Q: What have you learned about working in teams?
A: Working in a team is really nice as you can ask questions and work together to solve issues when things get rocky. There are some downfalls though if schedules end up conflicting with one another and group meetings end up being

pushed off.

Q: If you could do it all over, what would you do differently?
A: I would change quite a bit if I could do it all over again, I would look more in depth through the different types of project management models. Fix up documentation as soon as possible and really plan things out some more before beginning to code or devise how to tackle the problems. Other than that I don't think I would have done much else different.

12.3 Josh Strozzi's Reflection

Q: What technical information did you learn?
A: I learned a good bit about manipulating and maintaining a 3D scene using threejs, initializing a scene with variables stored in a JSON file, collaborating with a team of individuals all handling separate portions of the project and much more.

Q: What non-technical information did you learn?
A: This course wasn't really non-technical so trying to think of something specifically related to this course and it's work load that is ”non-technical” feels nigh impossible.

Q: What have you learned about project work?
A: Quick iteration depends, in large part, on the team handling their work quickly and efficiently. If one member's portion is dependent on the completion of another team members portion, that halt in the process, at each step of the process adds up to massive amounts of time lost in waiting for the next iteration.

Q: What have you learned about project management?
A: I learned that having on person sort of direct everyone can really help drive progress. One of the team members really stepped up to help push us to do the work we needed to do, and managed to do it in a way that felt like we weren't being boss around, but being properly led.

Q: What have you learned about working in teams?
A: The team dynamic, however collaborative, will always naturally become a leadership dynamic as without one filling the role, work will progress at a snails pace and the vacuum of this will will inherently shine a light on those persons who are most qualified and informed enough to lead.

Q: If you could do it all over, what would you do differently? A: I'd try harder to get over my code paralysis sooner. In a lot of my projects, I find myself to afraid to start the code, feeling like I won't be able to do it or figure it out. But, once I begin, if I can find the proper resources then, I can usually finish anything.

12.4 Martin Nguyen's Reflection

Q: What technical information did you learn?
A: I learned a lot of technical information about what technologies are being used on the web nowadays. I learned about

the es6 standards for JavaScript and how to effectively use them. Among other things, I learned about web bundlers, Babel, Service Worker, and the WebXR API.

Q: What non-technical information did you learn?
A: I did not learn much non-technical information, for the most part all the things I learned in these courses were technical whether it be coding or document writing. I did learn about how to communicate with group members and advisors.

Q: What have you learned about project work?
A: Among other things, I've learned that using the issues tracker, using branches, and using GitHub to resolve merge conflicts makes life a lot easier when developing in a group.

Q: What have you learned about project management?
A: I have learned that when managing a project with waterfall can lead to some big unexpected pitfalls into development later on. For example in this project, we assumed we would be able to create a UI using HTML elements layered over the AR element. This turned out to be wrong, and we had to completely redo our UI code from scratch. I have learned that getting started early and dividing up work into smaller pieces between group members is important to get progress started.

Q: What have you learned about working in teams?
A: I have learned that in person communication is much better than online communication. There are just somethings that are hard to express and share over the internet. I learned that communicating regularly and frequently is vital for group work and that asking for help when you've taken responsibility for something you can't handle by yourself.

Q: If you could do it all over, what would you do differently?
A: I would have started writing code earlier and written more code in general. The document specifications given by the IEEE are very esoteric, and as undergraduate students, they made very little sense. I would have gone to office hours more to talk about the expectations about the documents that we had to create.

Appendix

Text of Appendix A is Here

Text of Appendix B is Here