Universidad
Rey Juan Carlos

ESCUELA DE INGENIERÍA DE FUENLABRADA

GRADO EN INGENÍERIA EN SISTEMAS AUDIOVISUALES Y MULTIMEDIA

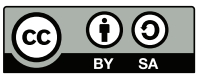
TRABAJO FIN DE GRADO

TÍTULO DEL TRABAJO CON LETRAS MAYÚSCULAS PARA SUSTANTIVOS Y ADJETIVOS

Autor : Juan José Arias Rojas

Tutor : Dr. Jesús María González Barahona

Curso académico 2024/2025



Dedicado a

todos aquellos que me animaron y apollaron

incluso en mis momentos de debilidad

II

Agradecimientos

Resumen

Summary

Índice general

Índice de figuras

Capítulo 1

Introducción

Capítulo 2

Tecnologías utilizadas

En esta sección se muestran las distintas tecnologías que han sido necesarias para el desarroyo de este proyecto tanto de forma directa como indirecta. Del mismo modo, se explicará su funcionalidad y la manera en la que contribuyeron al proyecto.

2.1. Tecnologías principales

Aquí se nombran y explican aquellas tecnologías que han tenido una implicación directa con el proyecto y que han tenido un papel crucial e imprescindible con las cuales, sin ellas, no se habria popido llegar a los resultados obtenidos. Las principales tecnologias utilizadas han sido HTML5, JavaScript y A-Frame, estas tres tecnologias han sido el nucleo principal del proyecto. El resto de las tecnologías principales son aquellas en las que se apollan esas tres, permitiendo que funcionen como lo hacen.

2.1.1. A-frame

A-Frame [1] es un web framework diseñado para construir experiencias de realidad virtual. A-frame esta basado en HTML y en JavaScript, permitiendo realizar cualquier escena que uno pueda imaginar sin necesidad de conocimientos avanzados en gráficos 3D. Al estar basado en HTML es posible realizar escenas simples directamente desde un archivo HTML simplemente importando la libreria de A-frame y añadiendo dentro de la etiqueta `<a-scene>` cualquier elemento que desee el usuario.

Uno de los ejemplos más simples que se puede llegar a realizar utilizando A-frame es el siguiente:

```
<html>
  <head>
    <script src="https://aframe.io/releases/1.6.0/aframe.min.js"></script>
  </head>
  <body>
    <a-scene>
      <a-box position="-1 0.5 -3" rotation="0 45 0" color="#4CC3D9"></a-box>
      <a-sphere position="0 1.25 -5" radius="1.25" color="#EF2D5E"></a-sphere>
      <a-cylinder position="1 0.75 -3" radius="0.5" height="1.5" color="#FFC65D"></a-
          cylinder>
      <a-plane position="0 0 -4" rotation="-90 0 0" width="4" height="4" color="#7BC8A4"></a
          -plane>
      <a-sky color="#ECECEC"></a-sky>
    </a-scene>
  </body>
</html>
```

Listing 2.1: Escena A-Frame básica

Este código de HTML genera una de las escenas más básicas que se pueden hacer en A-frame. La cual consiste en un plano y, encima, una serie de figuras geométricas, como se aprecia en la figura 2.1.

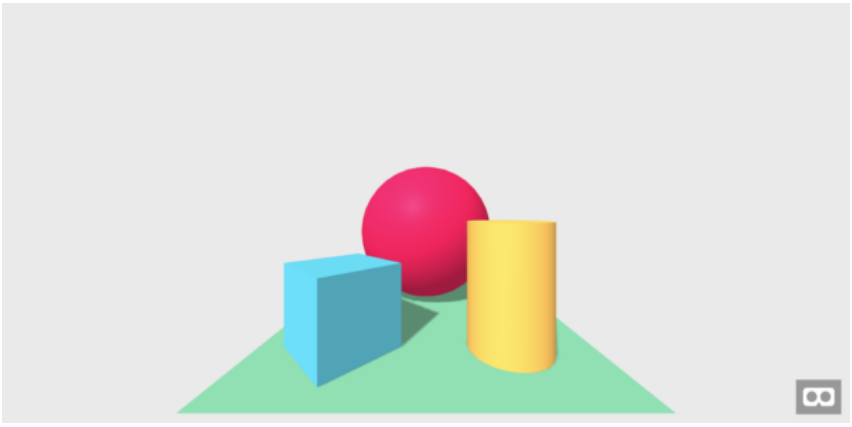


Figura 2.1: Escena básica de A-frame

Una de las principales características de esta tecnología es su arquitectura basada en un modelo de entidad-componente (*Entity Component System*), donde cada objeto dentro de la escena es una entidad que luego es integrada con Three.js para crear la escena. Una entidad en A-frame es un objeto HTML que se crea añadiendo la etiqueta `<a-entity>` y el componente

es la apariencia, comportamiento y funcionalidad que se le asigna mediante JavaScript.

Para hacer uso de un componente se requiere de una serie de pasos previos. Si el componente es externo es necesario importarlo en la cabecera del HTML del mismo modo que se importa A-frame y posteriormente añadirlo a la entidad en la que uno desee usarlo. Por otra parte, si el componente es de nuestra creación, primero hay que registrar y definir el componente. Esto se hace en un archivo externo de JavaScript mediante el método de `A-Frame.registerComponente`. A este método, del mimo modo que una función primero se le asigna un nombre, este sera el nombre del componente. Luego, este metodo consta de varias funciones internas las cuales se encargan de definir la apariencia y lógica de dicho componente. Alguna de estas funciones son la función `schema`, `init` o `tick`. La función `schema`, define las propiedades principales del componente, estas se pueden modificar dependiendo de la lógica del componente desde el HTML a la hora de llamarlo. La función `init` se ejecuta una única vez al inicio cuando se inicializa el componente y la función `tick` se ejecuta a cada frame de la escena. Una vez el componente esta registrado y programado importamos el archivo que contiene el componente al HTML y lo insertamos a la escena en la entidad u objeto que deseemos.

Un ejemplo de la llamada de un componente dentro de un archivo de HTML se puede apreciar en 2.2

```
<html>
  <head>
    <script src="https://aframe.io/releases/1.6.0/aframe.min.js"></script>
    <script src = "componente1.js"></script>
    <script src = "componente2.js"></script>
  </head>
  <body>
    <a-scene>
      <a-entity componente1></a-entity>
      <a-box componente2></a-box>
    </a-scene>
  </body>
</html>
```

Listing 2.2: Ejemplo de llamada de entidad

En este ejemplo primero se muestra como a una entidad vacia se le añade un compoennte llamado `componente1` y también se muestra como a una entidad de cubo se le añade el componente `componente2`. Del mismo modo, para la creación de un componente, en el archivo `.js` que posteriormente importaremos en la escena, unicamente es necesario registrar el

componente y definir sus caracteristicas como en el siguiente ejemplo:

```
  AFRAME.registerComponent('componente1', {
  init() {
    const box = document.createElement('a-box');
    box.setAttribute('position', '-1 0.5 -3');
    box.setAttribute('rotation', '0 45 0');
    box.setAttribute('color', '#4CC3D9');

    const sphere = document.createElement('a-sphere');
    sphere.setAttribute('position', '0 1.25 -5');
    sphere.setAttribute('radius', '1.25');
    sphere.setAttribute('color', '#EF2D5E');

    this.el.appendChild(box);
    this.el.appendChild(sphere);
  }
});
```

Listing 2.3: Ejemplo de creacion de entidad

Este fragmento de código registra el componente `componente1` el cual crea un cubo y un cilindro con unos colores, posición y rotación especificos y los añade a la escena.

Para este proyecto, A-Frame ha sido la tecnología principal, permitiendo elementos fundamentales como la creación y gestión de entornos de realidad virtual. También, permitio la integración de componentes diseñados mediante JavaScript y permitio el manejo de interacciones mas complejas a la vez que optimizaba el rendimiento de la escena 3D.

2.1.2. HTML5

HTML5 [3] es la quinta iteración del lenguaje de HTML (*marcado de hipertexto*), el cual es la estructura básica y principal de toda página web. Desarrollado conjuntamente por W3C (*World Wide Web Consortium*) [8] y WHATWG (*Web Hypertext Application Technology Working Group*) [9]. HTML5 introduce mejoras sustanciales respecto a sus anteriores versiones, incluyendo nuevas etiquetas semánticas, soporte multimedia mejorado y compatibilidad ampliada con diversos navegadores y dispositivos, dándole mayor flexibilidad y dinamismo a la creación de páginas web.

Una de sus nuevas introducciones clave es la introducción de etiquetas semánticas como: `<header>`, `<article>`, `<section>` y `<footer>`, que facilitan una estructuración clara y accesible del contenido web. Estas etiquetas no solo ayudan a mejorar la organización de la

información, si no que también facilitan la búsqueda de información por parte de los motores de búsqueda y facilitan la interpretación por parte de los desarrolladores. Además, HTML5 incorpora nuevas interfaces de programación de aplicaciones (*API*), destacándose las funcionalidades de almacenamiento local mediante `localStorage` y `sessionStorage`, que permiten la gestión eficiente de datos directamente en el navegador, eliminando la necesidad de bases de datos externas.

HTML5 también introduce mejoras significativas en la gestión de contenido multimedia, eliminando la necesidad de complementos externos. Para esto se implementaron las etiquetas `<audio>` y `<video>` las cuales permiten la incorporación directa de archivos de sonido y video en las páginas web, permitiendo mayor dinamismo en el desarrollo. Otro elemento que se añadió fue el elemento `<canvas>` el cual habilita la generación de gráficos y animaciones en tiempo real a través de JavaScript, permitiendo el desarrollo de aplicaciones interactivas y videojuegos en línea.

HTML5 se ha establecido firmemente como un estándar en el desarrollo de aplicaciones web progresivas (*PWA*) y en entornos multiplataforma. Su integración con tecnologías como CSS3 y JavaScript permite la creación de interfaces dinámicas y adaptativas, compatibles con una amplia gama de dispositivos, desde ordenadores hasta tabletas y teléfonos.

Gracias a que HTML es una tecnologia extensible, tecnologias como A-Frame o WebXR han sido posibles de utilizar ya que son extensiones de HTML. WebXR permitio poder trabajar con entornos VR desde el navegador mientras que A-Frame fue el eje central del proyecto, donde se estructuro la escena. HTML fue una de las tecnologías mas importantes ya que ofrece una estructura esencial para el desarroyo del proyecto, haciendo uso de sus extensiones de A-Frame y WebXR para poder mostrar los elementos de la escena en la realidad virtual.

2.1.3. JavaScript

JavaScript es un lenguaje de programación ligero y multiplataforma utilizado principalmente para la creación de contenido dinámico e interactivo para páginas web. Su felixibilidad permite desarrollar elementos para mejorar la interacción del usuario en páginas web tanto del lado del servidor como del lado del cliente. En 1997, JavaScript fue estandarizado por *ECMA* como ECMAScript y poco después como un éstandar *ISO*.

JavaScript, creado por Brendan Eich de Netscape en 1995 bajo el nombre de *Mocha*, pos-

teriormente se renombró a LiveScript y finalmente a JavaScript. En el año 2000, JavaScript se extendió al lado de los servidores con la introducción de tecnologías como *Node.js*, y posteriormente su popularidad creció con la llegada de *AJAX*. Y Con la llegada de ECMAScript 6 en 2015, se introdujeron características mas avanzadas y actualizaciones anuales, haciendo que hoy en día sea uno de los lenguajes mas importantes en el desarrollo web.

JavaScript es un lenguaje de programación de alto nivel, lo que significa que su leguaje esta diseñado para ser lo mas 'humano' posible, permitiendo una rapida comprensión del codigo sin necesidad de un nivel alto de conocimientos. Es un lenguaje basado en eventos ya sean entradas de ratón o entradas de teclado, permitiendo la creación de interfaces de usuario interactivas. JavaScript puede integrarse en un HTML dentro de la etiqueta `<script>` de forma directa, escribiendo el codigo. También, se puede introducir codigos de JavaScript, los cuales estén en archivos distintos con la extensión correspondiente al lenguaje (.js), esto es posible vinculando dicho archivo en la cabecera del HTML.

Para este proyecto formo un papel crucial, ya que fue con JavaScript que se desarrollaron los distintos componentes de A-Frame que se usaron para el funcionamiento del proyecto. Permitio diseñar e implementar la lógica que hay tras cada componente permitiendo alcanzar los resultados obtenidos.

2.1.4. Three.js

Three.js [10] es una biblioteca de codigo abierto de JavaScript utilizada para la creación de gráficos 3D en los navegadores web. Three.js funciona sobre WebGL [2], la cual permite generar y visualizar animaciones y escenas 3D en el navegador sin necesidad de complementos. La tecnologia de WebGL tambiénpuede usarse junto con el elemento `<canva>` de HTML.

Esta biblioteca proporciona una API de alto nivel la cual permite al usuario la creación y manipulación de geometrías 3D como cubos, esfera o planos, así como la aplicación de texturas o la aplicación de tanto camaras como de efectos de iluminación en las escena, permitiendo que el desarrollo de dichas escenas sea mucho mas simplificado. También, Three.js permite la posibilidad de importar modelos 3D desde archivos distintos creados desde aplicaciones distintas.

La biblioteca de Three.js ofrece una alta variedad de herramientas que permiten la gestión y control de usuario, facilitando asi tanto la navegación como la interacción dentro de las escenas 3D. También, Three.js soporta animaciones suaves y dinámicas tanto para objetos en la

escena como para camaras, lo cual permite la creación de efectos visuales mas dinámicos e impresionantes al igual que juegos interactivos.

Gracias a la amplia variedad de posibilidades que permite la biblioteca de Three.js las opciones de los elementos o escenas que se pueden llegar a crear son basicamente ilimitadas. Siendo capaz de crear escenas complejas como la de la figura 2.2

Figura 2.2: Escena de Three.js

Three.js permitio añadir una capa mas simplificada a WebGL, simplificando asi el proceso de creación y manipulación de los elementos 3D. Permitió crear un entorno 3D con mayor dinamismo y control para el proyecto.

2.1.5. WebXR

WebXR [7] es una tecnología desarrollada por W3C (*World Wide Web Consortium*) que permite crear experiencias inmersivas desde el navegador. Esta tecnología combina la Realidad Aumentada (*AR*) y la Realidad Virtual (*VR*) con la accesibilidad de un navegador web. Lo cual permite a los usuarios experimentar e interactuar con entornos tridimensionales a través de cualquier dispositivo con acceso a un navegador compatible.

A medida que la tecnología de WebXR ha ido evolucionando, se han desarrollado distintos tipos de experiencias para poder adaptarse a distintos contextos y necesidades. Hoy en día, se podria clasificar los distintos tipos en 3 categorias:

- **WebXR AR:** Este tipo de WebXR combina el mundo virtual y el mundo real, permitiendo que distintos elementos del mundo virtual puedan superponerse en el entorno físico a

través de la cámara de un dispositivo compatible, pero sin que llegue a influir el mundo real en los elementos virtuales.

- **WebXR VR:** Este modo permite a los usuarios sumergirse en el entorno virtual creado, y con la ayuda de dispositivos suplementarios como auriculares, la experiencia es aún mayor. Esta tecnología permite a los usuarios interactuar y experimentar en primera persona distintos entornos virtuales, desde juegos y entretenimiento, hasta simulaciones virtuales y visualizaciones en 3D.

- **WebXR MR(Mixed Reality):** Esta tecnología combina el mundo real y el virtual de forma mas profunda que la tecnología AR. Permite a los usuarios poder interactuar con elementos tantos virtuales como físicos y que estos puedan interactuar entre si. Este tipo de combinación proporciona un nivel mayor de interacción entre el usuario y y su entorno, dando mayor numero de posibilidades para la creación de contenido.

Estos distintos tipos de WebXR ofrecen distintos tipos de experiencias dependiendo del entorno virtual que se quiera diseñar.

2.1.6. WebGL

WebGL (*Web graphics Library*), se trata de una tecnología de bajo nivel multiplataforma usada para la renderización de gráficos tanto tridimensionales como bidimensionales dentro de cualquier navegador que sea compatible.

WebGL fue lanzado en 2011 por el grupo Khronos. Esta tecnología se fundamenta en OpenGL ES, la cual es una variante simplificada de OpenGL, diseñada para dispositivos moviles. WebGL ha sufrido numerables actualizaciones, lo cual ha permitido una evolución continua que ha ido mejorando tanto su funcionalidad como compatibilidad tanto en navegadores de escritorio como en navegadores en dispositivos móviles.

WebGL esta diseñado para trabajar directamente con la GPU (*Graphic Processing Unit*) del dispositivo, lo cual permite un mayor aprovechamiento de la computación para poder generar gráficos detallados y de alta calidad. Además, la tecnología de WebGL esta diseñada para poder integrarse de manera fluida con otros estandares de desarrollo web como HTML, CSS o DOM.

2.2. Tecnologias auxiliares

En esta sección se detallan y describen las distintas tecnologías que aunque no hayan influido de manera directa en los resultados del proyecto, han tenido un papel crucial para el desarroyo de este.

2.2.1. Visual Studio Code

Visual Studio Code (*VS Code*) es un potente editor de código abierto el cual esta disponible para Windows, Linux, MacOS y versión web. VS Code es una de las plataformas de edición de codigo más utilizadas a nivel mundial por toda clase de desarrolladores de software debido a su versatilidad, flexibilidad y amplia gama de extensiones que facilitan la edición y depuración de código.

También, gracias a su pestaña de **Source Control**, permite ir guardando el código en GitHub para tener un control de las distintas versiones del código.

Gracias a las numerosas extensiones que hay disponibles facilitan en gran medida la creación y depuración de código, permitiendo cosas como autocompletar palabras o expresiones o permitir compilar o depurar con algun lenguaje que VS Code no tenga por defecto. Algunas de las extensiones utilizadas en este proyecto han sido:

- **A-frame completition**: Permite autocompletar rapidamente a la hora de escribir los distintos elementos o Snippets que posee A-frame
- **Error lens**: A la hora de depuración, permite visualizar dentro del código si hay algun error de sintaxis o lógica
- **LaTeX Workshop**: Permite la escritura, compilación y previsualización de la memoria de este proyecto.

2.2.2. Github

GitHub es una plataforma basada en la nube que permite almacenar distintos repositorios y en cada repositorio codigo. Github esta basado en Git [4], el cual fue creado en 2005 por Linus Torvalds como un sistema de control de versiones de código abierto. Este sistema fue

desarroyado con la intención de ayudar a los desarrolladores a tener en cualquier dispositivo con acceso a internet todo el historial de código que estan desarrollando.

Esta tecnología es ampliamente utilizada a nivel mundial por desarrolladores de todo tipo debido a su capacidad de almacenar, soportar y gestionar proyectos de toda clase. Una de las funciones clave que presenta la plataforma, es la capacidad de crear ramificaciones (branches). Esta opcion permite a los desarroyadores poder crear copias del codigo del repositorio en el que estan trabajando para poder trabajar en paralelo y posteriormete poder integrar los cambios realizados en paralelo al codigo principal. Esta opcion también permite que mas de un desarrollador pueda trabajar en el mismo codigo, haciendo que cada uno trabaje en paralelo en ramas distintas.

GitHub aprovecha todas las ventajas que permite Git, permitiendo crear repositorios en la nube para sus proyectos Git, que los desarrolladores pueden llegar a compartir con otras personas. Esta plataforma tambien facilita la colaboración entre desarrolladores y aparte de las ventajas ya mencionadas de Git permite también el uso de herramientas de gestión de proyectos al igual que acciones automaticas.

2.2.3. Meta Quest 3

Meta Quest se trata de un dispositivo inalámbrico diseñado para disfrutar de contenido de realidad virtual (VR) por Meta [6]. Dicho dispositivo esta compuesto por unas gafas, un micrófono y auriculares integrados en un unico dispositivo que el usuario pone en su cabeza y un par de mandos inalámbricos. La primera version fue lanzada en 2020 con las Meta Quest 2 y 3 años mas tarde, en 2023 se lanzaron las Meta Quest 3.

Para este proyecto se han utilizado unas Meta Quest 3, las cuales coontaban con una memoria de 8 GBB de memoria RAM, una resolición de 2064x2208 pixeles por cada uno de los ojos. También, cuenta con mejoras respecto al campo de visión (FOV) respecto a su version anterior. Para este proyecto jugaron un papel cricial ya que al enfocarse en la detección de mandos el proyecto, sin las gafas no se habria podido probar los resultados que se iban obteniendo.

Figura 2.3: Meta Quest 3

2.2.4. LaTeX

LaTeX [5] es un software de uso libre de creación de documentación de alta calidad. Este sistema es altamente utilizado para la creación de documentación tecnica o cientifica de media o larga extensión, aunque gracias a su versatilidad se puede utilizar para cualquier tipo de documentación.

Entre las distintas caracteristicas de LaTeX cabe destacar su capacidad de manejar expresiones matematicas complejas, lo cual es una herramienta indispensable para cualquier documento cintifico. También gracias a su software motor TeX LaTeX es capaz de convertir los comandos de texto, utilizados para expresar los resultados tipográficos, en un archivo PDF profesional.

Además de facilitar la creación de formulas matematicas complejas, LaTeX también ofrece otras facilidades avanzadas, como puede ser la creación de un índice del contenido, un índice de las figuras utilizadas, gestión de bibliografias o referencias simplificando la organización del documento y la citación de figuras o elementos externos en la bibliografia.

Para este proyecto, LaTeX facilito considerablemente el trabajo a la hora de crear este documento, ayudando con la creación y gestión de indices, imagenes y bibliografia al igual que a estructurar el texto del documento de forma correcta.

Capítulo 3

Desarrollo del proyecto

En el siguiente capítulo se describe de forma detallada como fue el desarrollo del proyecto. Dicho desarrollo se describe en forma de sprints, desde los inicios y planteamiento del proyecto hasta obtener los resultados finales.

3.1. Sprint 0

Puesta en marcha del proyecto, planteamiento con el tutor y dominio del uso basico de A-Frame.

3.1.1. Objetivo Principal

El objetivo principal de este sprint se puede dividir en 2 partes principales.

- **El planteamiento con el tutor**, donde se identificaron las necesidades para el proyecto del mismo modo que planificar las distintas etapas y duración de cada una y los objetivos que se deseaban obtener.

- **Dominio del uso basico de A-Frame**. Aprender a usar de forma correcta como funcionan las escenas al igual que la creación e implementacion de componentes o el uso de librerias de A-Frame.

3.1.2. Implementación

La identificación de las necesidades y requisitos de un proyecto es una parte fundamental. Durante las primeras charlas con el tutor se concretaron los distintos objetivos y los posibles caminos que puede tomar este proyecto. Al principio se plantearon varios caminos, pero la idea general de todos ellos era la creación de un sistema de handtracking que sea capaz de interactuar con distintos elementos dentro de la escena.

Para lograr llegar a ese objetivo primero fue necesario dominar los elementos basicos de A-Frame como biene a ser la organización de la escena, el uso y creación de componentes y el uso de eventos. Para ello, primero se creo primero el repositorio de *Git*[1] con el que trabajaremos durante todo el proyecto. Se crearon una serie de escenas sencillas con el objetivo de dominar el uso de componentes y eventos simples.

3.1.3. Resultados

Para familiarimarme aun mas con el entorno de A-FRAME se crearon una serie de escenas muy sencillas donde se creaba y utilizaba un componente customizado y se hacia uso del evento nativo de A-Frame `click`. Estas escenas permitieron un mayor entendimiento del funcionamiento tanto de componentes como de eventos, que fueron fundamentales para el resto del proyecto.

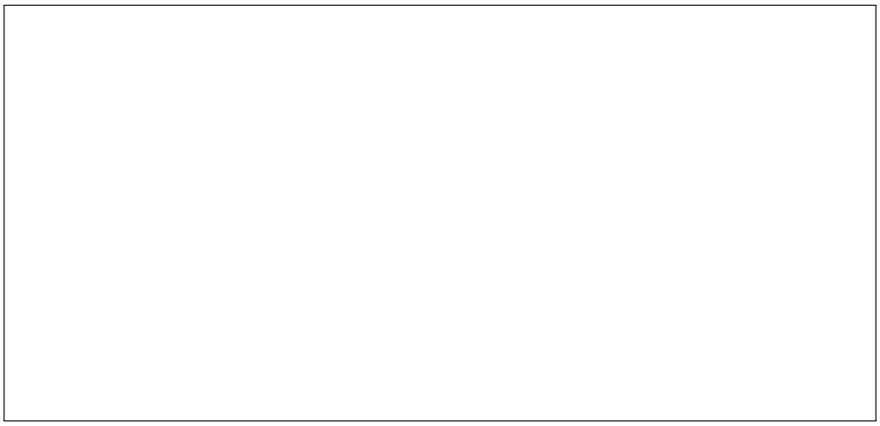


Figura 3.1: Escena de test de A-Frame

Tras la familiarización con el entorno de A-Frame y sus características el proyecto fue for-

[1] `https://github.com/JuJoarias/TFG`

malizado formalmente. Tras ellos se planteo el siguiente sprint del proyecto donde ya se empezarian los primeros pasos formales de este.

3.2. Sprint 1

texto pendiente

3.2.1. Objetivo Principal

texto pendiente

3.2.2. Implementación

texto pendiente

3.2.3. Resultados

texto pendiente

3.2.4. Planificación

texto pendiente

3.3. Sprint 2

texto pendiente

3.3.1. Objetivo Principal

texto pendiente

3.3.2. Implementación

texto pendiente

3.3.3. Resultados

texto pendiente

3.3.4. Planificación

texto pendiente

3.4. Sprint 3

texto pendiente

3.4.1. Objetivo Principal

texto pendiente

3.4.2. Implementación

texto pendiente

3.4.3. Resultados

texto pendiente

3.4.4. Planificación

texto pendiente

3.5. Sprint 4

texto pendiente

3.5.1. Objetivo Principal

texto pendiente

3.5.2. Implementación

texto pendiente

3.5.3. Resultados

texto pendiente

3.5.4. Planificación

texto pendiente

3.6. Sprint 5

texto pendiente

3.6.1. Objetivo Principal

texto pendiente

3.6.2. Implementación

texto pendiente

3.6.3. Resultados

texto pendiente

3.6.4. Planificación

texto pendiente

3.7. Sprint 6

texto pendiente

3.7.1. Objetivo Principal

texto pendiente

3.7.2. Implementación

texto pendiente

3.7.3. Resultados

texto pendiente

3.7.4. Planificación

texto pendiente

Capítulo 4

Resultados

Capítulo 5

Pruebas y experimentos

Capítulo 6

Conclusiones

6.1. Aplicación de lo aprendido

6.2. Lecciones aprendidas

6.3. Trabajos futuros

Bibliografía

[1] A-Frame. A-frame - introduction, 2024.
https://aframe.io/docs/1.6.0/introduction/.

[2] EncodeBiz. Parte 1: Webgl y su impacto en el desarrollo web moderno, 2024.
https://www.encodebiz.com/blog/parte-1-webgl-y-su-impacto-en-el-desar
cG9zdDozMjc=.

[3] E. Freeman and E. Robson. *Head First HTML and CSS*. O'Reilly Media, 2nd edition, 2018.

[4] Git SCM. Branching and merging, 2024.
https://git-scm.com/about/branching-and-merging.

[5] LaTeX Project. About the latex project, 2025.
https://www.latex-project.org.

[6] Meta. Company information, 2024.
https://www.meta.com/es-es/about/company-info/.

[7] Onirix. Webxr: Ejemplos y desarrollo en realidad extendida, 2024.
https://www.onirix.com/es/webxr-ejemplos-desarrollo-realidad-extendid

[8] W3C. The w3c mission, 2024.
https://www.w3.org/mission/.

[9] WHATWG. Whatwg - web hypertext application technology working group, 2024.
https://whatwg.org/.

[10] Wikipedia contributors. Three.js, 2024.
https://es.wikipedia.org/wiki/Three.js.