



**ESCUELA DE INGENIERÍA DE FUENLABRADA**

**INGENIERÍA TELEMÁTICA**

**TRABAJO FIN DE GRADO**

**EXTENDER BABIAXR CON MENÚ PARA AÑADIR  
VISUALIZACIONES**

Autor : Mario Cobo Martínez

Tutor : Dr. David Moreno Lumbreras

Curso académico 2024/2025





©2025 Mario Cobo Martínez

Algunos derechos reservados

Este documento se distribuye bajo la licencia

“Atribución-CompartirIgual 4.0 Internacional” de Creative Commons,

disponible en

<https://creativecommons.org/licenses/by-sa/4.0/deed.es>



*Dedicado a mí,  
por aquellas veces que dudé,  
por cada paso dado con miedo...  
y por seguir subiendo,  
incluso cuando la cima parecía tan lejana.*



# **Agradecimientos**

Aquí vienen los agradecimientos... Aunque está bien acordarse de la pareja, no hay que olvidarse de dar las gracias a tu madre, que aunque a veces no lo parezca disfrutará tanto de tus logros como tú... Además, la pareja quizás no sea para siempre, pero tu madre sí.



# Resumen

Este Trabajo de Fin de Grado se centra en el desarrollo de experiencias inmersivas de visualización e interacción en realidad virtual (VR) y aumentada (AR), mediante la creación de escenas interactivas que integran menús navegables y visualizaciones de datos utilizando la biblioteca BabiaXR sobre el framework A-Frame.

El proyecto incluye dos demostraciones principales. La primera de ellas consiste en una escena introductoria en la que, a través de un sistema de menús interactivos, el usuario puede explorar opciones que desencadenan la reproducción de sonidos o la generación de formas geométricas, las cuales pueden ser modificadas dinámicamente en color, tamaño y posición. La segunda escena, de mayor complejidad, permite representar datos en gráficos de barras y circulares mediante una interfaz basada en menús jerárquicos que guían al usuario hasta la visualización final. Estas visualizaciones son generadas en tiempo real utilizando BabiaXR y pueden ser manipuladas mediante gestos con las manos, sin necesidad de controladores físicos.

Ambas experiencias han sido diseñadas para ser compatibles con dispositivos como Meta Quest 3, haciendo uso de la API de WebXR y el seguimiento de manos para ofrecer una interacción más natural y accesible. Este trabajo no solo explora el potencial de la realidad extendida en el ámbito de la visualización de datos, sino que también busca facilitar el acceso a estas tecnologías a través de interfaces intuitivas y adaptables.



# **Summary**

This Final Degree Project focuses on the development of immersive virtual reality (VR) and augmented reality (AR) experiences for data visualization and interaction. It consists of creating interactive scenes that integrate navigable menus and data visualizations using the BabiaXR library built on the A-Frame framework.

The project includes two main demonstrations. The first is an introductory scene where users can explore a system of interactive menus that trigger the playback of sounds or the creation of geometric shapes, which can be dynamically modified in color, size, and position. The second, more complex scene enables users to represent data in bar and pie charts through a hierarchical menu interface that guides them to the final visualization. These charts are generated in real-time using BabiaXR and can be manipulated through hand gestures, without the need for physical controllers.

Both experiences are designed to be compatible with devices such as the Meta Quest 3, leveraging the WebXR API and hand tracking to offer a more natural and accessible interaction. This project not only explores the potential of extended reality in the field of data visualization, but also aims to make these technologies more approachable through intuitive and adaptable interfaces.



# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Contexto . . . . .	2
1.2. Objetivos del proyecto . . . . .	2
1.2.1. Objetivo general . . . . .	2
1.2.2. Objetivos específicos . . . . .	3
1.2.3. Objetivos complementarios . . . . .	3
1.3. Motivación . . . . .	4
1.4. Estructura de la memoria . . . . .	5
1.5. Disponibilidad del Software . . . . .	6
<b>2. Tecnologías utilizadas</b>	<b>7</b>
2.1. Tecnologías principales . . . . .	8
2.1.1. JavaScript . . . . .	8
2.1.2. HTML . . . . .	9
2.1.3. A-Frame . . . . .	9
2.1.4. WebXR . . . . .	10
2.1.5. BabiaXR . . . . .	12
2.1.6. Three.js . . . . .	13
2.1.7. Meta Quest 3 . . . . .	14
2.2. Tecnologías adicionales . . . . .	15
2.2.1. Glitch . . . . .	15
2.2.2. Github . . . . .	16
2.2.3. Latex . . . . .	16

<b>3. Desarrollo del proyecto</b>	<b>19</b>
3.1. Sprint 1 . . . . .	19
3.1.1. Objetivos del sprint . . . . .	19
3.1.2. Tareas realizadas . . . . .	20
3.1.3. Resultados obtenidos . . . . .	21
3.1.4. Conclusiones del sprint . . . . .	22
3.2. Sprint 2 . . . . .	23
3.2.1. Objetivos del sprint . . . . .	23
3.2.2. Tareas realizadas . . . . .	23
3.2.3. Resultados obtenidos . . . . .	24
3.2.4. Conclusiones del sprint . . . . .	33
3.3. Sprint 3 . . . . .	34
3.3.1. Objetivos del sprint . . . . .	34
3.3.2. Tareas realizadas . . . . .	34
3.3.3. Resultados obtenidos . . . . .	35
3.3.4. Conclusiones del sprint . . . . .	37
3.4. Sprint 4 . . . . .	38
3.4.1. Objetivos del sprint . . . . .	38
3.4.2. Tareas realizadas . . . . .	39
3.4.3. Resultados obtenidos . . . . .	40
3.4.4. Conclusiones del sprint . . . . .	43
3.5. Sprint 5 . . . . .	44
3.5.1. Objetivos del sprint . . . . .	44
3.5.2. Tareas realizadas . . . . .	44
3.5.3. Resultados obtenidos . . . . .	45
3.5.4. Conclusiones del sprint . . . . .	46
3.6. Sprint Final . . . . .	47
3.6.1. Objetivos del sprint . . . . .	47
3.6.2. Tareas realizadas . . . . .	48
3.6.3. Resultados obtenidos . . . . .	50
3.6.4. Conclusiones del sprint . . . . .	52

<b>ÍNDICE GENERAL</b>	<b>XI</b>
<b>4. Resultados</b>	<b>53</b>
4.1. Descripción funcional . . . . .	53
4.2. Arquitectura general . . . . .	57
4.2.1. index.html . . . . .	58
4.2.2. menu.js . . . . .	61
4.2.3. scene.json . . . . .	65
4.3. Guía de usuario . . . . .	68
4.4. Caso de uso La habitación de los diagramas . . . . .	73
4.5. Caso de uso La habitación a diseñar VR/AR . . . . .	75
4.6. Caso de uso La habitación de los sonidos . . . . .	76
<b>5. Conclusiones</b>	<b>79</b>
5.1. Consecución de objetivos . . . . .	79
5.2. Aplicación de lo aprendido . . . . .	80
5.3. Lecciones aprendidas . . . . .	81
5.4. Trabajos futuros . . . . .	82
<b>A.</b>	<b>85</b>
<b>Bibliografía</b>	<b>87</b>



# Índice de figuras

3.1. Imagen script A-frame básico . . . . .	25
3.2. Imagen escena dos. Antes y después de pulsar el botón. . . . .	26
3.3. Escena 3: Representación de datos filtrados. . . . .	27
3.4. Cambio de color en la esfera y de posición en la caja. . . . .	31
3.5. Cambio de color y posición en el cilindro con latencia. . . . .	31
3.6. Creación de figuras hijas y cambio de color y posición. . . . .	32
3.7. Creación de figuras hijas y cambio de forma y color. . . . .	33
3.8. Escena con botón y menús interactivos. Diagrama de barras y circular. . . . .	37
3.9. Escena con mandos como controladores. Esfera con cambio de color. . . . .	40
3.10. Escena con mandos como controladores. Menu con creación de cajas. . . . .	41
3.11. Escena con manos como controladores. Mini menu. . . . .	42
3.12. Escena con mandos como controladores. Panel agarrable. . . . .	42
3.13. Escena con mandos como controladores. Prototipo de menú final. . . . .	43
3.14. Aspecto final del menú interactivo . . . . .	46
4.1. Arquitectura General. . . . .	58
4.2. La habitación de los diagramas con visualizaciones dinámicas activadas. . . . .	75
4.3. La habitación a diseñar en modo AR y VR. . . . .	76
4.4. Demo 3 - La habitación de los sonidos. . . . .	78



# **Capítulo 1**

## **Introducción**

En el ámbito de la visualización de datos, la incorporación de tecnologías inmersivas como la realidad virtual (VR) y la realidad aumentada (AR) ha abierto nuevas posibilidades para representar información de forma más intuitiva, interactiva y envolvente. Estas tecnologías permiten superar muchas de las limitaciones presentes en los enfoques tradicionales basados en gráficos 2D, facilitando una mejor comprensión de datos complejos y multidimensionales gracias a la espacialidad, profundidad y libertad de movimiento que ofrecen los entornos tridimensionales.

El presente Trabajo de Fin de Grado se centra en extender las funcionalidades de BabiaXR —una biblioteca de código abierto para la creación de visualizaciones inmersivas en A-Frame— mediante la implementación de menús interactivos que permiten al usuario filtrar información y construir visualizaciones de datos personalizadas. A través del desarrollo de dos demos, se han creado escenas compatibles con realidad virtual y aumentada, en las que se emplean dispositivos como Meta Quest 3 para ofrecer una experiencia interactiva sin necesidad de controladores físicos, gracias al seguimiento de manos.

Estas escenas permiten al usuario navegar por diferentes opciones mediante gestos naturales y seleccionar visualizaciones en función de sus preferencias, todo dentro de un entorno inmersivo. La combinación de A-Frame, WebXR y el potencial de BabiaXR ha permitido diseñar una solución accesible, escalable y centrada en la experiencia del usuario, que aporta valor tanto en contextos educativos como profesionales, donde la interpretación de datos puede beneficiarse enormemente de estas nuevas formas de interacción.

## 1.1. Contexto

La realidad virtual (VR) y la realidad aumentada (AR) han dejado de ser tecnologías emergentes asociadas exclusivamente al entretenimiento para convertirse en herramientas clave en áreas como la educación, la medicina, la arquitectura o el análisis de datos. Su capacidad para generar entornos tridimensionales interactivos y envolventes ha transformado la manera en que las personas acceden, manipulan y comprenden la información.

Con la evolución de dispositivos como las Meta Quest 3 o las Apple Vision Pro, y el desarrollo de tecnologías como el seguimiento de manos mediante WebXR, la interacción en estos entornos se ha vuelto más natural, precisa y accesible, permitiendo al usuario prescindir de controladores físicos y utilizar únicamente sus gestos. Esta nueva forma de interacción no solo incrementa la inmersión, sino que abre la puerta a nuevas experiencias de exploración de información.

Uno de los campos que más se ha beneficiado de estas capacidades es la visualización de datos. Gracias a bibliotecas como BabiaXR, ahora es posible representar datos complejos en entornos 3D accesibles desde un navegador web, facilitando un análisis más dinámico, comprensible y participativo. En este contexto, el presente trabajo explora cómo combinar estas tecnologías para construir experiencias inmersivas interactivas, centradas en la creación de visualizaciones de datos a través de menús manipulables en VR y AR.

## 1.2. Objetivos del proyecto

### 1.2.1. Objetivo general

La finalidad de este proyecto es diseñar y desarrollar una aplicación interactiva en realidad virtual y aumentada que permita al usuario, mediante el uso de menús manipulables con las manos, explorar contenidos y generar visualizaciones de datos dinámicas utilizando la biblioteca BabiaXR sobre el framework A-Frame.

### 1.2.2. Objetivos específicos

- **Investigar tecnologías adecuadas** para el desarrollo de experiencias inmersivas en VR y AR, priorizando el uso de A-Frame, BabiaXR y WebXR con seguimiento de manos.
- **Desarrollar escenas inmersivas** with distintos levels of complejidad, including desde interacciones básicas (como la reproducción de sonidos o la creación de figuras geométricas) hasta representaciones de datos en diagramas.
- **Diseñar e implementar menús interactivos** en entornos tridimensionales que permitan al usuario navegar por opciones, submenús y funcionalidades mediante gestos manuales, sin necesidad de controladores físicos.
- **Garantizar la compatibilidad del sistema** con dispositivos como las Meta Quest 3 y navegadores compatibles con WebXR, optimizando la experiencia tanto en VR como en AR.
- **Integrar la biblioteca BabiaXR** para generar visualizaciones de datos en tiempo real dentro del entorno inmersivo, permitiendo su modificación (tamaño, color, posición) a través de la interfaz.

### 1.2.3. Objetivos complementarios

- **Aprender a crear componentes personalizados en A-Frame** e integrarlos correctamente en las escenas para modularizar el código y mejorar la reutilización y mantenibilidad del proyecto.
- **Familiarizarse con la biblioteca BabiaXR** y su integración para la visualización de datos en VR/AR.
- **Mejorar la experiencia visual y de usuario** mediante el diseño del front-end de las escenas.
- **Profundizar en el manejo de tecnologías web como HTML, JavaScript y CSS** aplicadas a entornos tridimensionales.

- **Familiarizarse con el uso y configuración de las gafas Meta Quest 3** comprendiendo sus capacidades, limitaciones y particularidades para el desarrollo y prueba de experiencias inmersivas tanto en realidad virtual como aumentada.
- **Consolidar y aplicar los conocimientos adquiridos a lo largo del grado** mediante el desarrollo de un proyecto práctico centrado en la integración de tecnologías web, realidad virtual y aumentada.

### 1.3. Motivación

El desarrollo de este proyecto surge de la motivación por explorar y mejorar las formas de interacción en entornos inmersivos, donde actualmente las interfaces de usuario, especialmente en realidad virtual, se encuentran aún en una etapa primitiva. A pesar del avance tecnológico en dispositivos como las Meta Quest 3, muchas experiencias VR/AR continúan utilizando sistemas de control limitados o poco intuitivos, lo que afecta la accesibilidad y la usabilidad general de las aplicaciones.

En este contexto, se identificó la necesidad de investigar nuevas formas de diseñar interfaces más naturales y eficaces, en particular aquellas que aprovechan el control por gestos y el seguimiento de manos. La creación de menús tridimensionales manipulables sin necesidad de controladores físicos representa un paso hacia una interacción más fluida, intuitiva y versátil, con potencial de ser aplicada en múltiples escenarios: desde la visualización de datos hasta la navegación por entornos complejos.

Además, este proyecto nace de un interés personal por integrar diversas tecnologías web (HTML, CSS, JavaScript, A-Frame) en un entorno práctico y desafiante, aplicando conocimientos adquiridos durante la carrera. También existe un propósito exploratorio relacionado con la biblioteca BabiaXR, que ofrece herramientas potentes para representar datos en contextos inmersivos, permitiendo construir experiencias visuales más ricas y útiles.

En conjunto, las motivaciones del proyecto no solo responden a un interés académico y técnico, sino también a la inquietud de contribuir al avance en el diseño de interfaces naturales dentro de la realidad extendida, abriendo posibilidades para aplicaciones futuras en educación, análisis de datos, simulación o entretenimiento.

## 1.4. Estructura de la memoria

- **Introducción** En este capítulo se presenta el objetivo general del proyecto, el contexto y los antecedentes. Se define el propósito de la aplicación en realidad virtual y aumentada, así como los objetivos planteados desde el inicio.
- **Tecnologías utilizadas** Este capítulo describe todas las tecnologías empleadas en el proyecto, organizadas en dos categorías:
  - Tecnologías principales: A-Frame, BabiaXR, WebXR, JavaScript, HTML, CSS y las herramientas asociadas que han sido fundamentales para la implementación de la aplicación.
  - Tecnologías adicionales: Glitch, Github, Latex, son otras tecnologías relacionadas que han sido clave para la interacción y la creación de código para realizar la interfaz de usuario.
- **Desarrollo del proyecto** En este capítulo se examina todo el proceso de desarrollo de la aplicación. Se sigue un enfoque metodológico, destacando las fases de planificación, diseño e implementación. Además, se abordan los retos y dificultades encontrados durante el desarrollo, como la integración de la biblioteca BabiaXR, la creación de menús interactivos y la adaptación para dispositivos VR.
- **Resultados** Este capítulo presenta el sistema final que se ha desarrollado, detallando las funcionalidades de la aplicación. Se describen los menús interactivos, la creación de sonidos y objetos en 3D, y la integración con bandas sonoras de películas. Se explica la arquitectura y los componentes clave que hicieron posible la implementación del sistema. Además, se discute la experiencia del usuario final al interactuar con la escena de realidad virtual. En cuanto a los resultados obtenidos, se presentan los comportamientos observados durante las pruebas de usuario, destacando aspectos como la navegación por los menús, la reproducción de sonidos y las interacciones con los objetos 3D. También se comenta la compatibilidad del sistema con dispositivos como las Meta Quest 3 y los navegadores compatibles con WebXR.
- **Conclusiones** Se evalúan los objetivos del proyecto, reflexionando sobre los logros alcan-

zados y las lecciones aprendidas durante el desarrollo. También se analizan las habilidades adquiridas durante el proyecto y cómo los conocimientos previos del grado han sido aplicados. Finalmente, se proponen ideas para futuros trabajos, tales como la implementación de nuevas funcionalidades, mejoras en la interacción del usuario, y la optimización del rendimiento. Se sugiere, por ejemplo, la inclusión de características adicionales como el movimiento por la escena con las manos y la capacidad de cambiar la vista del entorno.

## **1.5. Disponibilidad del Software**

Para explorar y acceder al proyecto, se han creado diversos recursos:

- **Página web:** Proporciona una visión detallada del proyecto, con toda la documentación importante y videos de las demostraciones realizadas. <https://mcobom2019.github.io/TFG>
- **Repositorio en GitHub:** Contiene el código fuente completo del proyecto. <https://github.com/mcobom2019/TFG>

# Capítulo 2

## Tecnologías utilizadas

En este apartado se describen las principales tecnologías empleadas para el desarrollo del presente Trabajo de Fin de Grado, así como las herramientas de apoyo que han facilitado la implementación, organización y documentación del proyecto.

La base tecnológica del proyecto se sustenta principalmente en A-Frame, un framework de código abierto basado en HTML y JavaScript, que permite la creación de experiencias inmersivas en realidad virtual (VR) y aumentada (AR) directamente desde el navegador. A-Frame se construye sobre Three.js, una biblioteca ampliamente utilizada para renderizar gráficos en 3D, lo que permite definir escenas complejas de forma más accesible y con mayor control sobre los elementos tridimensionales.

Para habilitar la compatibilidad con dispositivos de realidad virtual y aumentada, se ha utilizado WebXR, la API estándar que permite la integración de experiencias inmersivas dentro del navegador. Esta API facilita el uso de tecnologías como el seguimiento de manos, crucial en este proyecto para la interacción sin controladores físicos.

Uno de los elementos clave del desarrollo ha sido la utilización de BabiaXR, una librería que extiende las funcionalidades de A-Frame y permite la creación e integración de componentes visuales orientados a la visualización de datos, como gráficos de barras y diagramas circulares. BabiaXR ha sido fundamental para dotar al proyecto de capacidades interactivas avanzadas, especialmente en la escena principal dedicada a la representación de datos.

Como dispositivo principal para pruebas y despliegue, se ha empleado Meta Quest 3, unas gafas de realidad mixta que permiten ejecutar experiencias tanto en VR como en AR de forma autónoma. Estas gafas ofrecen capacidades avanzadas de interacción mediante seguimiento de

manos, lo que ha sido aprovechado para implementar una navegación natural a través de menús interactivos en las distintas escenas del proyecto.

Además de las tecnologías mencionadas, se han utilizado herramientas complementarias que han facilitado el flujo de trabajo y la organización del código. Entre ellas se encuentran Glitch, una plataforma online que permite desarrollar y desplegar proyectos web de forma sencilla y colaborativa; GitHub, utilizado para el control de versiones y almacenamiento del código fuente del proyecto; y LaTeX, empleado para la redacción y maquetación del documento de la memoria del TFG.

## 2.1. Tecnologías principales

### 2.1.1. JavaScript

JavaScript es un lenguaje de programación interpretado, orientado a objetos y basado en prototipos, ampliamente utilizado en el desarrollo web para crear experiencias dinámicas e interactivas. Es uno de los pilares fundamentales del desarrollo frontend junto a HTML y CSS, y se ejecuta en el navegador del cliente, permitiendo la manipulación del DOM (Document Object Model), la gestión de eventos, la comunicación con servidores y, en general, la creación de aplicaciones web interactivas.

En el contexto de este proyecto, JavaScript ha sido la tecnología central para implementar la lógica de funcionamiento de las escenas de realidad virtual y aumentada. Gracias a su compatibilidad con frameworks como A-Frame, ha sido posible definir componentes personalizados, gestionar la interacción del usuario con los elementos de la escena (como botones, gráficos o menús), y responder a eventos como el seguimiento de manos o la navegación por submenús.

Además, JavaScript ha permitido integrar librerías externas como BabiaXR y controlar dinámicamente la generación y modificación de objetos tridimensionales en tiempo real. Por ejemplo, se ha utilizado para cambiar el color o tamaño de una figura geométrica según la interacción del usuario, o para crear y actualizar gráficos de barras y diagramas circulares de forma interactiva.

La elección de JavaScript como lenguaje principal está motivada también por su compatibilidad con la API WebXR, lo que facilita el desarrollo de experiencias inmersivas directamente

en navegadores web compatibles, sin necesidad de plugins adicionales.

Gracias a su flexibilidad, facilidad de integración y amplio ecosistema de herramientas, JavaScript se posiciona como una de las tecnologías más adecuadas para el desarrollo de experiencias inmersivas accesibles y multiplataforma, tal como se ha demostrado en este proyecto.

### 2.1.2. HTML

HTML (HyperText Markup Language) es el lenguaje de marcado estándar utilizado para estructurar y presentar contenido en la web. Su función principal es definir la estructura de una página web mediante etiquetas, que organizan el contenido en elementos como títulos, párrafos, listas, enlaces, imágenes, formularios o scripts.

En el marco de este proyecto, HTML cumple un papel fundamental como soporte estructural para las escenas desarrolladas en realidad virtual y aumentada. Gracias a frameworks como A-Frame, que extienden HTML con un conjunto de etiquetas personalizadas para representar elementos tridimensionales (por ejemplo, `<a-box>`, `<a-entity>`, `<a-scene>`), se facilita la creación de entornos inmersivos sin necesidad de escribir código 3D complejo desde cero.

Esta integración entre A-Frame y HTML permite que los desarrolladores diseñen escenas VR/AR declarativamente, manteniendo una sintaxis intuitiva y accesible. Por ejemplo, a través de simples etiquetas HTML es posible añadir objetos geométricos, luces, cámaras, gráficos o menús interactivos, todo ello dentro del navegador web.

Además, HTML ha sido esencial para incorporar componentes interactivos como botones o interfaces de usuario (UI), que forman parte de los menús del proyecto. Estos elementos se complementan con JavaScript para dotarlos de comportamiento dinámico, permitiendo, por ejemplo, navegar por submenús, activar animaciones o modificar objetos en la escena.

### 2.1.3. A-Frame

A-Frame es un framework de código abierto desarrollado por Mozilla que permite construir experiencias de realidad virtual (VR) y aumentada (AR) directamente en el navegador, utilizando una sintaxis basada en HTML. Se apoya en Three.js para el renderizado 3D y en WebXR para acceder a las capacidades de realidad virtual y aumentada del dispositivo, lo que lo convierte en una herramienta especialmente potente para el desarrollo de experiencias inmersivas.

accesibles desde la web.

Una de sus principales ventajas es la simplicidad que ofrece al permitir crear y manipular objetos tridimensionales mediante etiquetas HTML personalizadas. Esto facilita el desarrollo incluso a quienes no tienen conocimientos avanzados en gráficos 3D o programación gráfica, lo que lo hace ideal para proyectos educativos, prototipos rápidos o entornos interactivos como el desarrollado en este Trabajo de Fin de Grado.

En este proyecto, A-Frame ha sido la tecnología principal sobre la que se ha construido toda la estructura y lógica de las escenas VR/AR. Gracias a su ecosistema modular y extensible, ha sido posible:

- Crear entornos 3D en los que los usuarios pueden interactuar mediante seguimiento de manos o dispositivos como Meta Quest 3.
- Integrar menús tridimensionales con botones, elementos interactivos y componentes visuales personalizables.
- Utilizar librerías adicionales como BabiaXR para la creación de gráficos interactivos y WebXR para habilitar la compatibilidad con dispositivos inmersivos.

Además, A-Frame permite una fácil integración de componentes personalizados, lo cual ha sido clave en este proyecto para extender funcionalidades específicas como la creación dinámica de gráficos, cambio de colores de objetos, escalado interactivo o manipulación de geometrías en tiempo real.

Por otro lado, su compatibilidad con tecnologías web estándar como JavaScript y HTML, ha facilitado una integración fluida con otras herramientas utilizadas durante el desarrollo, como Glitch para despliegue online, y GitHub para control de versiones.

En resumen, A-Frame ha sido el núcleo del desarrollo VR/AR del proyecto, permitiendo la creación de experiencias ricas e interactivas con una curva de aprendizaje accesible y una gran flexibilidad técnica.

#### 2.1.4. WebXR

WebXR es una API (Interfaz de Programación de Aplicaciones) desarrollada por el World Wide Web Consortium (W3C) que permite a las aplicaciones web acceder a funcionalidades de

realidad virtual (VR) y realidad aumentada (AR) directamente desde el navegador. Su nombre proviene de la combinación de “Web” y “XR”, donde XR es un término general que abarca tanto VR como AR.

Esta tecnología es fundamental para el desarrollo de experiencias inmersivas multiplataforma, ya que permite que una misma aplicación web funcione en una gran variedad de dispositivos, como gafas de realidad virtual (por ejemplo, Meta Quest 3), móviles compatibles con AR, ordenadores con visores conectados, etc.

Entre sus principales funciones destacan:

- Acceso a sensores de posición y orientación del dispositivo, permitiendo que las escenas reaccionen al movimiento de la cabeza o del cuerpo del usuario.
- Renderizado estereoscópico, que crea la sensación de profundidad mediante el uso de diferentes imágenes para cada ojo.
- Seguimiento de manos y controladores, permitiendo interacciones naturales como pulsar botones, agarrar objetos o manipular menús 3D.
- Renderizado inmersivo, adaptando el contenido de la escena a las capacidades del dispositivo y el entorno del usuario.

En el marco de este proyecto, WebXR ha sido esencial para permitir la compatibilidad y ejecución de las escenas VR y AR directamente desde el navegador web, sin necesidad de instalar aplicaciones externas. Gracias a esta API:

- Se ha podido utilizar el seguimiento de manos con las Meta Quest 3 para interactuar con menús y elementos 3D sin necesidad de mandos físicos.
- Se ha conseguido una integración fluida de experiencias inmersivas en plataformas accesibles, manteniendo una alta calidad visual y rendimiento.
- Se ha garantizado la portabilidad del proyecto a otros dispositivos compatibles con WebXR, ampliando su alcance y utilidad.

En combinación con frameworks como A-Frame, WebXR ha simplificado el desarrollo de aplicaciones XR al abstraer la complejidad técnica del hardware subyacente, permitiendo centrarse en la experiencia de usuario y la funcionalidad del proyecto.

### 2.1.5. BabiaXR

BabiaXR es una librería de código abierto desarrollada por investigadores de la Universidad Rey Juan Carlos (URJC), concretamente por Jesús M. González-Barahona, David Moreno-Lumbreras y Andrea Villaverde Hernández. Esta herramienta nace con el objetivo de facilitar la creación de visualizaciones de datos inmersivas en entornos de realidad extendida (XR), aprovechando tecnologías web abiertas como A-Frame y WebXR.

BabiaXR proporciona una serie de componentes predefinidos que permiten al desarrollador construir interfaces interactivas en realidad virtual y aumentada sin necesidad de implementar desde cero toda la lógica de interacción o visualización. Entre sus funcionalidades destacan la creación de gráficos 3D (diagramas de barras, circulares, etc.), menús, botones interactivos, filtros de datos, así como elementos configurables por el usuario (como el color o tamaño de los objetos). Todo ello permite convertir conjuntos de datos en experiencias inmersivas que pueden ser manipuladas y exploradas de manera intuitiva.

Una de las características más valiosas de BabiaXR es que ha sido diseñada pensando tanto en la representación de datos como en la investigación en interacción humano-computador. Por ello, incluye herramientas para el registro del comportamiento de los usuarios, lo que permite realizar estudios empíricos sobre cómo interactúan con la información en entornos XR.

En el marco de este proyecto, BabiaXR ha sido una tecnología central para el desarrollo de la escena principal. Gracias a esta librería, ha sido posible:

- Crear diagramas de datos (barras y circulares) en un entorno tridimensional completamente interactivo.
- Implementar menús y submenús navegables mediante botones virtuales seleccionables con las manos en dispositivos como Meta Quest 3.
- Permitir al usuario modificar en tiempo real aspectos visuales como el tamaño o color de los elementos de la escena, o cambiar el conjunto de datos representado.
- Diseñar una experiencia inmersiva en VR/AR accesible desde cualquier navegador compatible con WebXR.

BabiaXR ha contribuido notablemente a simplificar el desarrollo de interfaces XR avanzadas, permitiendo centrarse en la lógica y diseño de la experiencia, en lugar de en los aspectos técnicos

más complejos del renderizado o la interacción 3D.

### 2.1.6. Three.js

Three.js es una biblioteca de JavaScript que permite crear y renderizar gráficos 3D de manera sencilla utilizando WebGL, una API que permite renderizado acelerado por hardware directamente en el navegador sin necesidad de plugins. Esta herramienta actúa como una capa de abstracción sobre WebGL, proporcionando una interfaz más amigable y accesible para desarrolladores que deseen trabajar con gráficos tridimensionales.

Three.js ha sido fundamental en el desarrollo de experiencias web inmersivas, ya que ofrece una amplia gama de funcionalidades, como:

- Creación y manipulación de geometrías 3D (cubos, esferas, planos, etc.).
- Aplicación de materiales y texturas realistas.
- Iluminación y sombreado con distintos tipos de luces.
- Cámara y controles de navegación.
- Animaciones y efectos visuales.
- Soporte para carga de modelos 3D complejos en formatos como glTF, OBJ o FBX.

En el contexto de este proyecto, Three.js no ha sido utilizada de forma directa, pero es una tecnología base esencial, ya que A-Frame, el framework principal empleado, está construido sobre ella. Esto significa que toda la representación gráfica 3D que se observa en las escenas —ya sean objetos, geometrías, o elementos interactivos— se genera internamente utilizando las capacidades de Three.js.

Gracias a Three.js, A-Frame y otras herramientas como BabiaXR pueden ofrecer entornos tridimensionales ricos, dinámicos e interactivos sin que el desarrollador tenga que preocuparse por la complejidad técnica de trabajar directamente con WebGL. Así, aunque no se haya programado directamente con esta librería, Three.js ha sido clave para el renderizado 3D del proyecto, proporcionando una base gráfica sólida, estable y optimizada para dispositivos como las Meta Quest 3 o navegadores compatibles con WebXR.

### 2.1.7. Meta Quest 3

Meta Quest 3 es un visor de realidad virtual (VR) desarrollado por Meta (anteriormente conocida como Facebook), lanzado como parte de la serie Quest. Este dispositivo se destaca por ser completamente autónomo, es decir, no requiere un PC ni cables para funcionar, lo que lo hace más accesible y cómodo para una experiencia inmersiva. La Meta Quest 3 está equipada con un procesador potente, pantallas de alta resolución y sensores avanzados que permiten una experiencia visual fluida y realista.

Una de las características más relevantes de la Meta Quest 3 es su capacidad de ofrecer experiencias tanto de realidad virtual como de realidad aumentada (AR), gracias a su sistema de passthrough en color, que permite ver el entorno físico con una visualización de alta calidad mientras se interactúa con elementos virtuales. Esto es crucial en el contexto del desarrollo de entornos interactivos en 3D, como el que se ha utilizado en este proyecto, ya que facilita la transición entre la realidad y el mundo virtual de una forma más natural y cómoda.

En términos de interacción, las Meta Quest 3 permiten un seguimiento preciso de las manos y del cuerpo mediante su sistema de cámaras externas, lo que elimina la necesidad de controladores tradicionales para muchas interacciones. Esto ha sido aprovechado en este proyecto para permitir que los usuarios interactúen con los menús y elementos de las escenas a través de gestos manuales y movimientos corporales, mejorando la inmersión y haciendo la experiencia más intuitiva.

Al ser compatible con tecnologías como WebXR, la Meta Quest 3 ofrece una plataforma ideal para el desarrollo de aplicaciones de realidad virtual y aumentada basadas en navegador, lo que se alinea perfectamente con las herramientas utilizadas en este trabajo. Además, su accesibilidad y facilidad de uso la convierten en un dispositivo ideal para probar y desplegar experiencias interactivas sin la necesidad de equipos adicionales.

En este proyecto, la Meta Quest 3 ha sido utilizada como el dispositivo principal para probar y validar las escenas de realidad virtual (VR) y aumentada (AR) desarrolladas, aprovechando su potencia de procesamiento, capacidad de interacción sin controladores y su compatibilidad con las tecnologías web.

## 2.2. Tecnologías adicionales

### 2.2.1. Glitch

Glitch es una plataforma en línea que permite el desarrollo, edición y despliegue de aplicaciones web directamente desde el navegador. Su principal ventaja es la simplicidad y accesibilidad que ofrece tanto para principiantes como para desarrolladores con experiencia. En el contexto de este proyecto, Glitch ha sido utilizada como entorno de desarrollo y despliegue de las escenas creadas en A-Frame, permitiendo probar de forma rápida y sencilla los avances sin necesidad de configurar un servidor local o servicios de hosting complejos.

Una de las características más útiles de Glitch es su funcionalidad de edición colaborativa en tiempo real, similar a la de Google Docs, lo que facilita trabajar en equipo o recibir feedback inmediato. Además, cualquier modificación realizada en el código se refleja casi instantáneamente en la vista previa del proyecto, lo cual resulta especialmente útil para ajustar visualmente las escenas en realidad virtual y aumentada.

Glitch soporta tecnologías web modernas como HTML, JavaScript, y por supuesto, frameworks como A-Frame y librerías como BabiaXR o Three.js, fundamentales en este trabajo. Gracias a esto, se ha podido mantener un flujo de trabajo ágil, visualizando los cambios directamente en las Meta Quest 3 mediante el navegador integrado del visor.

En resumen, Glitch ha sido una herramienta clave durante el desarrollo del proyecto por su capacidad para:

- Editar código en la nube sin necesidad de instalaciones locales.
- Desplegar escenas VR/AR de forma inmediata.
- Compartir fácilmente los avances mediante URLs públicas.
- Hacer pruebas en dispositivos como las Meta Quest 3 de forma directa.

Su simplicidad y eficacia han facilitado enormemente el proceso de prototipado y presentación de resultados.

### 2.2.2. Github

GitHub es una plataforma de desarrollo colaborativo ampliamente utilizada para alojar y gestionar proyectos de software utilizando el sistema de control de versiones Git. Esta herramienta permite llevar un seguimiento detallado de los cambios realizados en el código fuente, trabajar en distintas ramas de desarrollo, colaborar con otros desarrolladores y mantener el histórico completo del proyecto.

Durante el desarrollo de este Trabajo de Fin de Grado, GitHub ha sido una herramienta clave para organizar, versionar y respaldar el código de las distintas escenas y componentes implementados. Además, ha permitido sincronizar el trabajo entre distintos dispositivos y garantizar la seguridad de los avances mediante almacenamiento en la nube.

Uno de los recursos más relevantes ofrecidos por GitHub y aprovechado en este proyecto ha sido GitHub Pages. Se trata de un servicio gratuito que permite desplegar páginas web estáticas directamente desde un repositorio. Gracias a esto, ha sido posible alojar y mostrar las escenas de realidad virtual y aumentada desarrolladas en A-Frame y BabiaXR, permitiendo que cualquier persona con acceso al enlace pueda visualizar el proyecto desde su navegador, ya sea en modo escritorio, móvil o incluso en dispositivos VR como las Meta Quest 3.

GitHub Pages ha sido especialmente útil para:

- Publicar versiones actualizadas del proyecto de forma inmediata.
- Probar la compatibilidad de las escenas en diferentes dispositivos y navegadores.
- Compartir avances con mi tutor, compañeros o evaluadores de forma sencilla.
- Evitar configuraciones complejas de servidores web, al facilitar el despliegue directamente desde el repositorio.

En conjunto, GitHub ha servido como eje central para la organización del proyecto, mientras que GitHub Pages ha sido el medio ideal para mostrar los resultados de manera accesible, rápida y profesional.

### 2.2.3. Latex

LaTeX es un sistema de preparación de documentos de alta calidad, especialmente utilizado en entornos académicos y científicos para la creación de informes, artículos, libros y otros tipos

de publicaciones técnicas. Se basa en un lenguaje de marcado que permite separar el contenido del formato, lo que facilita la creación de documentos con una estructura clara, coherente y bien organizada, especialmente cuando se incluyen fórmulas matemáticas, referencias bibliográficas y tablas complejas.

A diferencia de los procesadores de texto tradicionales, LaTeX no permite la edición visual directa del documento en el mismo estilo con el que aparecerá finalmente. En cambio, el usuario escribe el contenido utilizando comandos específicos, y luego LaTeX genera el formato final del documento, lo que permite un control preciso sobre la presentación. Esta separación entre contenido y formato resulta especialmente útil cuando se necesita mantener la coherencia en el diseño de documentos largos o complejos.

En este Trabajo de Fin de Grado, LaTeX ha sido la herramienta elegida para la redacción de la memoria del proyecto. Gracias a su capacidad para gestionar referencias bibliográficas mediante BibTeX, organizar secciones con facilidad y generar índices automáticos, LaTeX ha permitido crear un documento estructurado de manera profesional, facilitando la presentación del contenido técnico y académico del proyecto.

Las principales ventajas de utilizar LaTeX para la memoria del proyecto incluyen:

- Alta calidad tipográfica: LaTeX genera documentos con una excelente calidad de impresión, lo que es ideal para la presentación de proyectos académicos.
- Manejo eficiente de referencias y bibliografía: Gracias a herramientas como BibTeX, LaTeX permite gestionar citas y referencias de manera automática, garantizando su formato correcto y evitando errores.
- Soporte para fórmulas matemáticas: En proyectos técnicos como el mío, que pueden requerir fórmulas o expresiones matemáticas, LaTeX proporciona un soporte excelente para incluir ecuaciones de manera precisa y clara.
- Estructura y organización del contenido: LaTeX facilita la creación de documentos largos, con un control total sobre la organización del contenido (secciones, subsecciones, listas, tablas, figuras, etc.), lo cual es crucial en la documentación técnica.

Gracias a LaTeX, he podido crear una memoria ordenada, precisa y profesional, acorde con los estándares requeridos para un Trabajo de Fin de Grado.



# **Capítulo 3**

## **Desarrollo del proyecto**

En este capítulo se expone el proceso de desarrollo llevado a cabo para la realización del Trabajo de Fin de Grado. Desde las fases iniciales de planificación y exploración tecnológica hasta la implementación progresiva de las escenas interactivas en realidad virtual y aumentada. El desarrollo se ha estructurado siguiendo una metodología basada en sprints, lo que ha permitido dividir el trabajo en etapas bien definidas y avanzar de forma iterativa. A lo largo de este capítulo se detallan los principales retos, decisiones técnicas, herramientas empleadas y evolución del proyecto, documentando paso a paso cómo se ha construido el sistema final.

```
@inproceedings{robles2005self,  
    title={Self-organized development in libre software:  
        a model based on the stigmergy concept},  
    author={Robles, Gregorio and Merelo, Juan Juli\'an  
        and Gonz\'alez-Barahona, Jes\'us M.},  
    booktitle={ProSim'05},  
    year={2005}  
}
```

### **3.1. Sprint 1**

#### **3.1.1. Objetivos del sprint**

El objetivo principal de este primer sprint fue establecer las bases conceptuales, técnicas y organizativas necesarias para abordar el desarrollo del Trabajo de Fin de Grado. Esto incluyó

tanto la comprensión profunda del tema del proyecto como la preparación del entorno de trabajo y la familiarización con las herramientas clave que se utilizarán a lo largo del desarrollo.

Además, durante este sprint se definió la planificación temporal general del proyecto, estructurándolo en varios sprints que permitieran una organización ágil y progresiva del trabajo. Esta planificación ayudó a segmentar las tareas en fases concretas, permitiendo así un mejor control del avance, la detección temprana de bloqueos y una gestión más eficiente del tiempo disponible.

En resumen, este sprint sirvió como punto de partida para sentar las bases teóricas, técnicas y temporales del proyecto, permitiendo afrontar con mayor seguridad y claridad las fases posteriores de implementación.

### 3.1.2. Tareas realizadas

- **Investigación de la documentación de las tecnologías principales:**

- Se realizó un análisis detallado sobre las tecnologías que serían utilizadas a lo largo del proyecto, con especial énfasis en A-Frame, WebXR y BabiaXR.
- Se estudió la documentación oficial de A-Frame, así como sus recursos adicionales, con el fin de entender su funcionamiento y sus capacidades para la creación de escenas de realidad virtual.

- **Revisión de recursos proporcionados por el tutor:**

- Se dedicó tiempo a estudiar los recursos proporcionados por el profesor, tales como tutoriales y seminarios específicos sobre A-Frame y BabiaXR.
- Se consultaron recursos como el A-Frame Playground, que proporciona ejemplos prácticos y una base sobre la que construir las futuras escenas de realidad virtual.

- **Investigación sobre BabiaXR:**

- Se investigó sobre la librería BabiaXR, su uso para la creación de interfaces interactivas en realidad virtual y aumentada, y su integración con A-Frame.
- Se revisaron tutoriales y documentación relacionada con BabiaXR para entender su rol en la visualización de datos dentro de entornos VR y AR.

**■ Creación del repositorio en GitHub:**

- Se configuró el repositorio de GitHub para el proyecto, lo que permitió el control de versiones y la organización del código a lo largo del desarrollo.
- Se creó la estructura básica de archivos y carpetas en el repositorio para la implementación de los primeros pasos del proyecto.

**■ Planificación temporal inicial:**

- Se dedicó tiempo a la planificación del proyecto, estableciendo un cronograma preliminar con los tiempos estimados para cada fase de desarrollo y organización de los sprints.

### 3.1.3. Resultados obtenidos

**■ Comprensión general de A-Frame:**

- Se obtuvo una visión clara de cómo A-Frame funciona como un framework para crear experiencias de realidad virtual dentro del navegador. Tras explorar su documentación y realizar ejercicios prácticos, se comprendió cómo crear y manipular escenas 3D usando este framework.
- Se familiarizó con la sintaxis de A-Frame y su integración con otras tecnologías web como HTML y JavaScript.

**■ Conocimiento sobre BabiaXR:**

- Se adquirió una comprensión sólida sobre cómo BabiaXR extiende las capacidades de A-Frame para crear interfaces interactivas y gestionar elementos dentro de escenas VR. BabiaXR permite la manipulación de componentes en tiempo real, lo que será esencial para la creación de visualizaciones interactivas de datos en el proyecto.

**■ Familiarización con la integración de A-Frame y WebXR:**

- A través de los recursos revisados, se consolidó la comprensión de cómo A-Frame se apoya en WebXR para integrar experiencias de realidad virtual y aumentada directamente en el navegador. Este conocimiento será fundamental para el desarrollo

de interfaces interactivas que serán accesibles a través de dispositivos como las Meta Quest 3.

■ **Visión general del alcance del proyecto:**

- Después de realizar esta fase de investigación y formación, se tiene una idea clara de los componentes que formarán parte del proyecto. Se entiende que el desarrollo de interfaces interactivas de datos, usando A-Frame y BabiaXR, será central, y que se trabajará en la integración de estos elementos en un entorno VR/AR.

■ **Revisión del repositorio en GitHub:**

- Se completó la configuración inicial del repositorio, lo que permitió organizar de manera estructurada los archivos del proyecto y habilitar el control de versiones desde el inicio del desarrollo.

### 3.1.4. Conclusiones del sprint

En esta fase inicial del proyecto, se logró una comprensión general de las tecnologías clave que se utilizarán, como A-Frame, BabiaXR y WebXR. Tras explorar los recursos y tutoriales proporcionados, junto con la documentación disponible, se obtuvo una visión clara de cómo estas herramientas interactúan entre sí para permitir la creación de experiencias de realidad virtual interactivas. Además, se configuró el repositorio en GitHub, lo que permitirá organizar el código de manera eficiente a lo largo del desarrollo del proyecto.

Aunque ahora se tiene una idea general de lo que se debe realizar en el proyecto, se es consciente de que solo al comenzar a desarrollar las primeras funciones prácticas y a integrar estos elementos, se podrá comprender completamente los retos y detalles que surgirán. Esta fase de investigación y aprendizaje ha preparado al autor para avanzar con más seguridad en la implementación de las siguientes etapas del proyecto, pero es en los próximos sprints donde los objetivos tomarán forma real y se podrá afinar la planificación en función de los desafíos técnicos que puedan surgir.

## 3.2. Sprint 2

### 3.2.1. Objetivos del sprint

El objetivo principal de este sprint ha sido adquirir un dominio funcional de las bases de A-Frame mediante la creación de múltiples escenas de prueba. Estas escenas han permitido familiarizarse con la estructura y funcionamiento del framework, así como con su sintaxis y capacidades gráficas. Además, se ha profundizado en el aprendizaje y desarrollo de componentes personalizados en A-Frame, sentando así una base sólida para la construcción de escenas más complejas e interactivas en futuras etapas del proyecto.

### 3.2.2. Tareas realizadas

En este segundo sprint, el trabajo se ha centrado en el aprendizaje práctico y progresivo de A-Frame y BabiaXR, mediante la creación de múltiples escenas experimentales con el objetivo de familiarizarse con su funcionamiento y capacidades. Las tareas realizadas incluyen:

- Creación de una primera escena básica en A-Frame compuesta por tres figuras geométricas (cubo, esfera y cilindro) dispuestas sobre un plano, sirviendo como punto de partida para entender la estructura de una escena 3D.
- Realización del tutorial oficial de BabiaXR, en el que se implementa una escena con representación de diagramas de barras y circulares a partir de un fichero .json, incluyendo el uso de filtros interactivos.
- Desarrollo de escenas con pequeños menús interactivos, uno de ellos diseñado para generar cajas en posiciones aleatorias dentro de la escena al pulsar un botón.
- Implementación de un menú de enlaces que, al pulsar diferentes botones, permite reproducir canciones específicas, explorando así la integración de elementos multimedia en realidad virtual.
- Pruebas centradas en la obtención de logs en la consola, tanto de forma inmediata como con temporización, con el fin de facilitar la depuración de futuras escenas.

- Creación de otras escenas en las que se manipulan figuras geométricas para que cambien dinámicamente de color y posición mediante interacción con elementos de la interfaz.
- Desarrollo e integración de componentes personalizados en A-Frame para encapsular funcionalidades, reutilizarlas y adaptarlas a diferentes contextos dentro de las escenas.

### 3.2.3. Resultados obtenidos

Los ejercicios realizados no solo permitieron entender la estructura y sintaxis de A-Frame, sino también experimentar con sus capacidades interactivas y visuales, como la creación de figuras geométricas, la incorporación de menús, la interacción con elementos o la integración de datos externos mediante JSON. A continuación, se describen las escenas más representativas que se desarrollaron durante este sprint.

- **Escena 1 – Menú A-frame inicial sencillo con figuras geométricas.**

Uno de los primeros ejercicios realizados consistió en la creación de una escena simple en A-Frame con tres figuras geométricas básicas: una caja azul, una esfera roja y un cilindro amarillo, todas ellas dispuestas sobre un plano verde que actúa como suelo. Este ejercicio sirvió como introducción práctica a la estructura básica de una escena en A-Frame, entendiendo cómo se declaran las entidades y sus atributos dentro del entorno HTML. El código utilizado para realizar esta escena es el siguiente:

```
<html>
  <head>
    <script src="https://aframe.io/releases/1.7.0/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-scene>
  </body>
</html>
```

```

<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>

```

Como resultado, podemos visualizar la siguientes escena, por la cuál podemos movernos con total libertad:

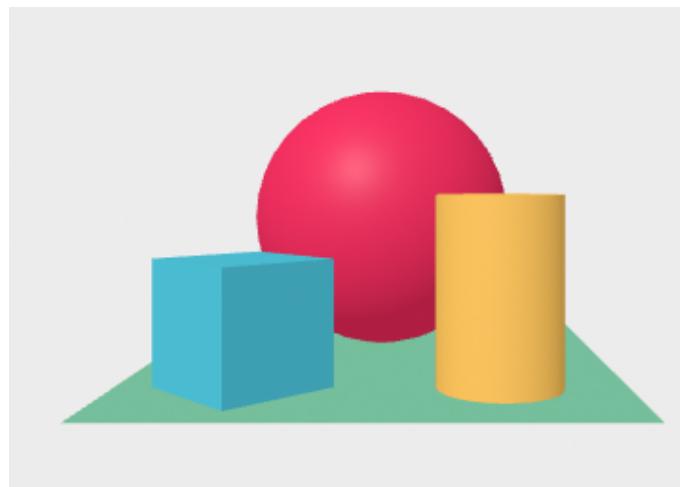


Figura 3.1: Imagen script A-frame básico

Con esta escena, se consolidaron los siguientes aspectos clave:

- La sintaxis y jerarquía de una escena en A-Frame (`a-scene`, `a-entity`, etc.).
- Cómo definir formas básicas mediante primitivas (box, sphere, cylinder, plane).
- Personalización de atributos como color, posición, tamaño y rotación.

#### ■ Escena 2: Menú con botón para añadir cajas en posiciones aleatorias.

En este segundo ejercicio, se desarrolló una escena interactiva en A-Frame que incorpora un pequeño menú funcional, diseñado con el objetivo de probar la interacción básica mediante botones y la generación dinámica de entidades en la escena.

La escena se compone de una caja roja fija, sobre la cual se sitúa un botón interactivo con la etiqueta “Añadir Caja”. Al pulsar dicho botón (usando el sistema de eventos de A-

Frame), se genera una caja azul en tiempo real, cuya posición es aleatoria, pero siempre se sitúa alrededor de la caja roja, en posiciones laterales calculadas dentro de un rango definido.

El resultado de la escena es el siguiente:



Figura 3.2: Imagen escena dos. Antes y después de pulsar el botón.

Este ejercicio permitió:

- Explorar la creación dinámica de entidades en A-Frame utilizando JavaScript.
  - Aplicar funciones matemáticas básicas para definir posiciones aleatorias con lógica espacial.
  - Entender el sistema de eventos de A-Frame, en particular el uso de addEventListener para activar acciones con la interacción del usuario.
- **Escena 3: Exploración de BabiaXR con representación y filtrado de datos.**

En esta tercera escena se lleva a cabo una primera aproximación práctica al uso de BabiaXR, una biblioteca diseñada para facilitar la visualización de datos en entornos de realidad virtual a través de A-Frame.

El objetivo principal de este ejercicio fue comprender cómo se representan datos estructurados utilizando los componentes predefinidos que ofrece BabiaXR, especialmente los destinados a gráficos circulares (pastel) y de barras.

Características de la escena:

- Se cargan datos desde un archivo .json externo, que contiene información estructurada para su representación visual.

- Se generan dos tipos de diagramas:
  - Un diagrama circular donde cada porción representa una categoría específica del conjunto de datos.
  - Un diagrama de barras, utilizado para comparar valores entre distintas categorías.
- La escena incluye controles de filtrado que permiten al usuario seleccionar qué subconjunto de datos visualizar. Esto permite observar dinámicamente cómo cambian los gráficos en función de los filtros aplicados.

Resultados de la escena:

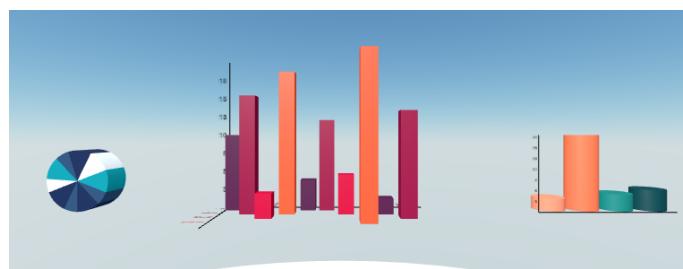


Figura 3.3: Escena 3: Representación de datos filtrados.

Con este ejercicio se ha aprendido lo siguiente:

- Se aprendió cómo se declaran los componentes de BabiaXR en HTML utilizando etiquetas como `<a-entity babia-bar-chart>` y `<a-entity babia-pie-chart>`.
- Se configuraron parámetros como `field`, `groupBy`, `filter`, y `title` para personalizar cada gráfico.
- Se entendió el flujo de trabajo completo: importación de datos → configuración del componente → visualización en VR.
- **Escena 4: Menú de navegación con enlaces a vídeos de YouTube.**

Esta cuarta escena está centrada en el desarrollo de un menú interactivo en A-Frame, compuesto por tres botones que, al ser seleccionados, redirigen al usuario a diferentes vídeos de YouTube. El objetivo principal de este ejercicio fue experimentar con la interactividad básica dentro de la realidad virtual y entender cómo gestionar eventos de clic y redirección a recursos externos.

El menú está conformado por tres botones visualmente diferenciados, colocados en una caja o panel visible en la escena.

Cada botón está asociado a un evento que, al ser activado (mediante clic con controlador o interacción con la mano), abre un enlace externo en una nueva pestaña del navegador o redirige al usuario directamente a un vídeo de YouTube.

Los vídeos son distintos entre sí y se eligieron con el fin de simular una galería o catálogo multimedia en VR.

Valor del ejercicio:

- Este ejercicio fue especialmente útil para reforzar los conocimientos sobre eventos y componentes interactivos dentro de A-Frame.
- Supuso una primera práctica real sobre cómo construir interfaces de usuario funcionales en VR, un aspecto esencial para la navegación en entornos inmersivos.
- Además, permitió experimentar con los límites de la integración de contenido externo dentro del contexto de una escena 3D en el navegador.

#### ■ Escenas 5-7: Creación y uso de componentes personalizados en A-Frame.

En estas escenas, el objetivo principal fue aprender a desarrollar componentes propios en A-Frame, lo que supuso un paso importante hacia la creación de lógica personalizada y reutilizable dentro de las escenas VR. A través de varios ejercicios breves y experimentales, se exploraron aspectos clave como el uso de funciones tick, la integración de scripts embebidos en el HTML y la manipulación directa del DOM, además del uso de logs.

Características comunes de estas escenas:

- Creación de componentes personalizados mediante `AFRAME.registerComponent`, definiendo comportamientos específicos que se pueden aplicar a entidades dentro de la escena.
- Implementación de la función `tick`, que permite ejecutar código de forma continua o periódica mientras la escena está activa. Se utilizó para lanzar mensajes en consola cada cierto tiempo, simulando procesos de seguimiento o monitoreo.

- Uso de scripts embebidos directamente en el archivo HTML para declarar componentes inline, facilitando la organización del código y su vinculación directa con la estructura de la escena.
- Manipulación del DOM, incluyendo la creación dinámica de entidades mediante JavaScript. Se aprendió a generar nodos con `document.createElement`, añadirles atributos y agregarlos a la escena con `appendChild`.
- Uso de logs en scripts para ayudar a la trazabilidad del código.

Ejemplos de desarrollo:

- Un componente que lanza un mensaje en la consola cada segundo, útil para entender el ciclo de vida de una entidad y validar la correcta ejecución de lógica.
- Scripts que generan nuevas entidades en tiempo de ejecución, probando la inserción de objetos en el mundo virtual desde código.

Valor de estos ejercicios:

- Supusieron una introducción práctica al ciclo de vida de los componentes en A-Frame.
  - Ayudaron a comprender cómo A-Frame se apoya en la estructura de HTML y JavaScript, reforzando el entendimiento del framework como una extensión declarativa de tecnologías web estándar.
  - Sentaron las bases para el desarrollo de funcionalidades más complejas y reutilizables, clave en proyectos con lógica avanzada o escenas dinámicas.
- **Escenas 8-11: Interactividad avanzada con figuras geométricas y retraso en la ejecución de eventos.**

En estas escenas, se exploró una mayor interactividad con las entidades en A-Frame mediante la manipulación de figuras geométricas, como cajas, esferas y cilindros. Estas figuras son interactivas y responden a diferentes acciones del usuario, lo que permite modificar su apariencia y comportamiento en la escena. Un aspecto clave de estas escenas es la introducción de un retraso aleatorio en la ejecución de las interacciones, lo que agrega una capa de complejidad y realismo en el comportamiento de los objetos.

Características principales de estas escenas:

- Interactividad con figuras geométricas: Las escenas incluyen figuras básicas como cajas, esferas y cilindros que pueden ser estáticas o moverse de manera circular en la escena.
- Acciones de click: Al hacer clic en una de estas figuras, se ejecutan diversas acciones. Algunas de estas acciones incluyen:
  - Cambio de color: Al hacer clic en una figura, el color de la misma cambia aleatoriamente entre un conjunto predefinido de colores.
  - Cambio de forma: Las figuras pueden transformarse en otras formas geométricas, como cambiar de una caja a una esfera, o viceversa.
  - Creación de figuras hijas: Al hacer clic en la figura principal, se pueden añadir figuras hijas a la escena. Estas figuras hijas, que también pueden cambiar de color o posición, se relacionan directamente con la figura principal.
  - Cambio de posición: Además de los cambios de forma y color, las figuras pueden desplazarse a posiciones aleatorias dentro de la escena.
  - Retraso aleatorio en la ejecución de eventos: Una característica clave que se implementó fue la función de retraso que, al hacer clic en una figura, retrasa entre 1 y 10 segundos la ejecución de la acción deseada. Este retraso aleatorio agrega una diferencia temporal en la respuesta a la interacción del usuario, lo que puede simular efectos como la latencia o un comportamiento más orgánico y menos instantáneo.

Ejemplos de funcionalidades en estas escenas:

- Esfera y caja - Cambio de color y posición: Al hacer clic sobre la esfera, esta cambia de color de forma aleatoria entre varios colores predeterminados. La caja, al igual que la esfera, tiene una interactividad basada en el clic. En este caso, al hacer clic sobre la caja, su posición se mueve a un lugar aleatorio en la escena. La interacción es inmediata, no tiene la componente de retraso aleatorio.

El resultado es el siguiente:

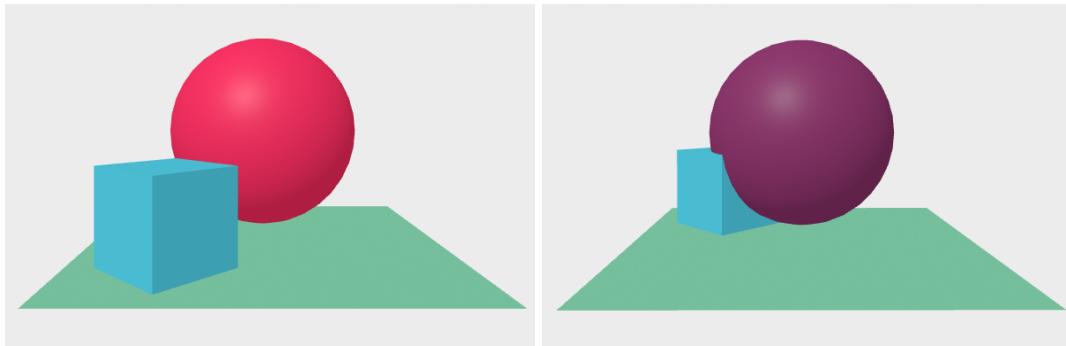


Figura 3.4: Cambio de color en la esfera y de posición en la caja.

- Escena con cilindro interactivo: En este segundo ejercicio, se crea una escena interactiva que contiene un cilindro. Al igual que en los ejemplos anteriores con la esfera y la caja, esta figura tiene dos funcionalidades principales basadas en el clic del usuario: Al hacer clic sobre el cilindro, este cambia tanto de color como de posición. Los cambios no ocurren de inmediato, sino que se retrasan aleatoriamente entre 1 y 10 segundos, añadiendo un elemento de latencia y haciendo la interacción más dinámica.

El resultado es el siguiente:



Figura 3.5: Cambio de color y posición en el cilindro con latencia.

- Escena con esfera que genera esferas hijas y cilindro interactivo: En este tercer ejercicio, la escena se compone de una esfera principal, que al hacer clic genera tres esferas hijas. Además, se incluye un cilindro que, al igual que en los ejemplos previos, cambia de color y posición con un retraso de entre 1 y 10 segundos.

Al hacer clic sobre la esfera principal, esta genera tres esferas hijas dentro de la

escena. Cada una de las esferas hijas puede interactuar de manera independiente con el usuario, permitiendo cambiar su color con un clic.

El cilindro en esta escena, como en el ejercicio anterior, tiene la funcionalidad de cambiar su color y posición con un clic del usuario.

El resultado es el siguiente:

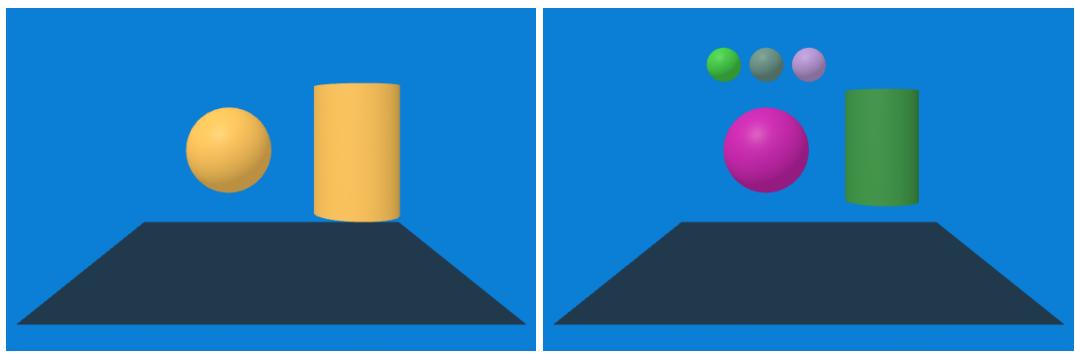


Figura 3.6: Creación de figuras hijas y cambio de color y posición.

- Escena interactiva con conversión de objetos y esferas hijas: En este último ejercicio, se crea una escena más compleja en la que interactúan varias entidades. En ella se combina una caja que se convierte en cilindro, un cilindro que se divide en tres partes, y una esfera en movimiento que genera una esfera hija que cambia de color al hacer clic. Cada una de estas interacciones tiene un retraso aleatorio de entre 1 y 10 segundos, lo que añade un toque dinámico a la escena.

En esta escena, existe una caja que, al hacer clic, se transforma en un cilindro. El cilindro se divide en tres partes separadas las cuales si se cíican cambian de color.

En la escena también hay una esfera que se mueve en un movimiento circular constante. Al hacer clic sobre esta esfera, se crea una esfera hija que es un poco más grande que la esfera principal. Si se clica cualquiera de las dos esferas cambian su color.

Tanto la caja como las esferas cambian su color con una latencia de entre 1 y 10 segundos después del click.

El resultado es el siguiente:

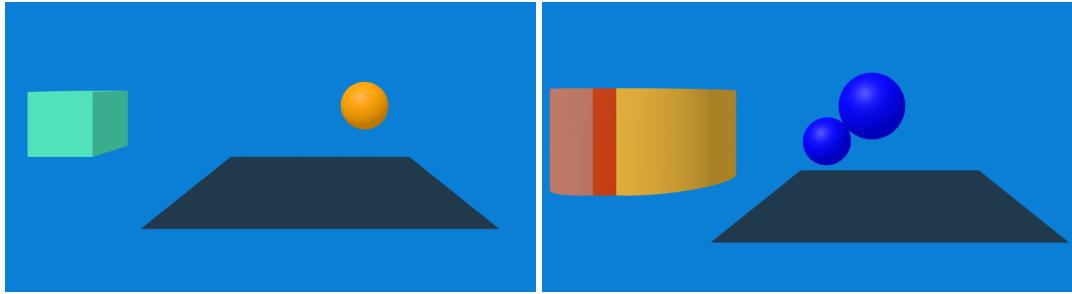


Figura 3.7: Creación de figuras hijas y cambio de forma y color.

### 3.2.4. Conclusiones del sprint

En este segundo sprint, se ha logrado una comprensión significativa del funcionamiento básico de A-Frame y su capacidad para crear experiencias interactivas en entornos 3D. Al realizar una serie de ejercicios prácticos, he podido experimentar con diferentes tipos de entidades y aprender cómo se pueden manipular a través de interacciones como cambios de posición, color y forma, integrando también un comportamiento dinámico mediante el uso de retrasos aleatorios en las acciones.

Los ejercicios realizados han permitido entender los principios fundamentales de la programación con A-Frame y sus componentes personalizados, así como la creación y manipulación de escenas interactivas. A través de las figuras geométricas, el uso de componentes personalizados para la creación de menús y el aprendizaje de técnicas avanzadas como la creación de entidades hijas y la implementación de retrasos en la interacción con las entidades, se ha consolidado el conocimiento práctico que es esencial para el desarrollo del proyecto final.

Además, con la incorporación de BabiaXR para el manejo de visualización de datos y el uso de diagramas circulares y de barras, se ha abierto el campo para futuras integraciones de datos dentro de las escenas en 3D, lo cual será clave para el proyecto final, donde se busca integrar elementos de visualización interactiva en un entorno virtual.

Si bien se ha avanzado significativamente en la creación de escenas interactivas con A-Frame, aún queda trabajo por hacer para perfeccionar las interacciones y explorar nuevas funcionalidades, como la integración de gráficos más complejos, que serán abordadas en los siguientes sprints. En general, este sprint ha sido crucial para afianzar las bases necesarias para el desarrollo posterior del proyecto.

## 3.3. Sprint 3

### 3.3.1. Objetivos del sprint

El objetivo principal de este sprint es la creación de una escena interactiva en A-Frame que integre menús interactuables mediante los cuales se pueda navegar a través de diferentes submenús hasta llegar a la opción de representar datos utilizando BabiaXR. Estos datos se representarán visualmente mediante diagramas de barras y diagramas circulares, permitiendo una navegación fluida y sencilla a través de las diferentes opciones que se presenten en los menús.

### 3.3.2. Tareas realizadas

#### ■ Diseño e Implementación de la Escena Inicial en A-Frame:

- Se creó una nueva escena en A-Frame en la que se diseñaron y ubicaron los menús principales y submenús. Para ello, se utilizaron entidades como cajas planos, y se implementó la funcionalidad básica para la interacción de los usuarios con los botones de estos menús.
- En esta escena se colocaron las bases para los botones interactivos, asignando interacciones mediante el uso de eventos como el clic, con el objetivo de abrir submenús sucesivos.

#### ■ Creación de Menús Interactivos:

- Se crearon varios menús en la escena, con botones que permiten a los usuarios navegar por diferentes opciones. Cada botón, al ser presionado, despliega un submenú con opciones adicionales.
- Implementación de un sistema de botones en A-Frame usando las entidades de la librería básica de A-Frame, asignando interacciones de clic para desplegar submenús o realizar otras acciones.

#### ■ Integración de BabiaXR para la Representación de Datos:

- Se integró la librería BabiaXR en la escena, asegurando que los diagramas de barras y circulares se generaran correctamente a partir de los datos proporcionados.
- Implementación de un sistema para navegar entre los menús y submenús hasta llegar a la visualización de los datos, asegurando que al hacer clic en los botones de los submenús, se carguen los diagramas correspondientes.
- Los diagramas de barras y circulares fueron configurados para que pudieran mostrar datos provenientes de un archivo .json, utilizando los filtros disponibles en BabiaXR.

#### ■ **Interactividad con los Datos:**

- Implementación de la funcionalidad para que, al seleccionar las opciones de los menús, los diagramas de barras o circulares representen los datos solicitados por el usuario.
- Se configuraron las interacciones de los botones para cambiar de un menú a otro sin errores y mantener la experiencia inmersiva del usuario.

#### ■ **Pruebas de Funcionamiento y Ajustes:**

- Se realizaron pruebas de interacción para verificar que la navegación por los menús fuera fluida y que los diagramas de datos se visualizaran correctamente.
- Se corrigieron errores menores de funcionalidad y se ajustaron los tiempos de transición entre menús para mejorar la experiencia de usuario.

### **3.3.3. Resultados obtenidos**

Como resultado del trabajo realizado en este sprint, se ha desarrollado una escena interactiva completamente funcional en A-Frame, donde el usuario puede navegar a través de un sistema de menús y submenús para visualizar datos utilizando diagramas generados con la librería BabiaXR.

La escena ha sido diseñada para interacción mediante ratón y está compuesta por los siguientes elementos funcionales:

- **Botón de Inicio:** Un botón rojo inicial situado en el centro de la escena permite al usuario desplegar el menú principal.

- Menú Principal: Este menú incluye dos opciones:
  - Barras: Para generar un diagrama de barras.
  - Circular: Para generar un diagrama circular.
- Submenú de Filtros 1: Al seleccionar cualquiera de las dos opciones anteriores, se despliega un submenú que permite al usuario filtrar los datos por distintas categorías:
  - Motor: Opciones de “eléctrico”, “gasolina” y “diésel”.
  - Color: Opciones de “blanco”, “negro”, “amarillo” y “rojo”.
  - Puertas: Opciones de “3 puertas” o “5 puertas”.
  - Completo: Visualización del conjunto de datos sin ningún tipo de filtro.
- Submenú de Filtros 2: Al seleccionar cualquiera de las opciones anteriores, se despliega un nuevo submenú que permite filtrar más específicamente los datos por lo siguiente:
  - Si se seleccionó Motor las opciones de este menú serán: “eléctrico”, “gasolina” y “diésel”.
  - Si se seleccionó Color las opciones de este menú serán: “blanco”, “negro”, “amarillo” y “rojo”.
  - Si se seleccionó Puertas las opciones de este menú serán: “3 puertas” o “5 puertas”.
  - Si se seleccionó Completo directamente nos redirige al submenú final.
- Submenú Final: Una vez seleccionados el tipo de diagrama y el filtro deseado, aparece un nuevo submenú con dos opciones:
  - Mostrar Diagrama: Genera el diagrama correspondiente con los parámetros elegidos.
  - Borrar Diagrama: Elimina el diagrama actualmente mostrado de la escena.

Navegación entre Menús: Todos los menús y submenús incluyen un botón adicional que permite al usuario volver atrás al menú anterior, asegurando una navegación fluida y controlada por toda la estructura de la interfaz.

Este sistema de navegación jerárquico permite al usuario seleccionar paso a paso cómo quiere visualizar los datos, lo cual representa un avance significativo en la usabilidad e interactividad de la aplicación. La correcta integración de BabiaXR ha sido clave para conseguir que los diagramas se representen de forma dinámica y visualmente clara, de acuerdo con las opciones seleccionadas.

Además, el sistema de borrado de diagramas implementado permite limpiar la escena sin tener que reiniciar el entorno, facilitando múltiples pruebas y visualizaciones dentro de la misma sesión.

Escena resultante:

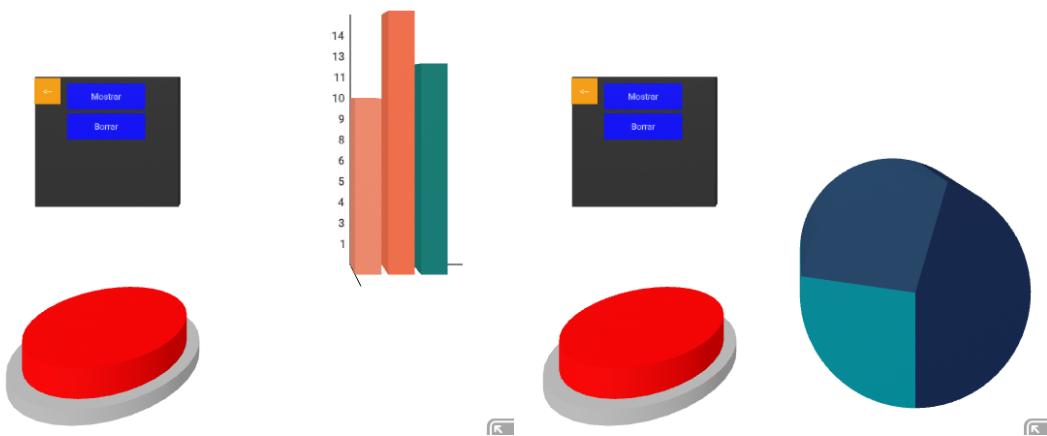


Figura 3.8: Escena con botón y menús interactuables. Diagrama de barras y circular.

### 3.3.4. Conclusiones del sprint

Este sprint ha supuesto un avance significativo en el desarrollo del proyecto, al materializarse una primera versión funcional de una escena interactiva con navegación por menús y visualización de datos mediante gráficos generados con BabiaXR. Se ha logrado estructurar un sistema jerárquico de menús que permite al usuario seleccionar, paso a paso, el tipo de representación gráfica deseada y aplicar distintos filtros sobre los datos.

Durante este proceso se ha consolidado el conocimiento adquirido en los sprints anteriores, especialmente en lo relacionado con la creación y gestión de interfaces interactivas en A-Frame, así como en el manejo de la lógica necesaria para mostrar u ocultar elementos en función de la navegación del usuario. También se ha profundizado en el funcionamiento interno de BabiaXR,

comprendiendo cómo cargar y filtrar los datos desde archivos .json y representarlos de forma dinámica y eficaz.

Asimismo, se han identificado buenas prácticas para el diseño modular de los menús, lo que facilitará futuras extensiones o mejoras en la experiencia de usuario. El sistema de navegación hacia atrás en los menús, y la opción de borrar el diagrama representado, mejoran la usabilidad y aportan flexibilidad a la escena.

En definitiva, el sprint ha cumplido sus objetivos de forma satisfactoria y ha sentado las bases de la estructura de interacción que se mantendrá y perfeccionará en los próximos sprints.

## 3.4. Sprint 4

### 3.4.1. Objetivos del sprint

El principal objetivo de este sprint fue comenzar a trabajar con las gafas de realidad virtual Meta Quest 3 y adaptar el proyecto para su correcto funcionamiento en un entorno inmersivo. En esta fase, se buscó comprender en profundidad el funcionamiento de las Meta Quest 3, tanto en lo relativo a su sistema operativo como en las posibilidades de interacción que ofrece al usuario.

Una vez adquirida esta base, el enfoque se centró en adaptar la escena desarrollada en el sprint anterior para que pudiera ser utilizada a través de los dispositivos de entrada de las gafas, especialmente los mandos y, de forma más relevante, las manos del usuario. El objetivo clave fue lograr una interacción natural con los menús mediante el uso de las manos, lo que implicó el aprendizaje e implementación de componentes que permitieran detectar gestos y colisiones táctiles dentro del entorno VR.

Además, otro objetivo fundamental fue implementar correctamente la funcionalidad de agarrar y mover los menús (grabbable), con el fin de proporcionar una experiencia más realista e inmersiva. Este aspecto supuso un desafío técnico relevante que fue superado con éxito, permitiendo que los elementos de la interfaz pudieran ser desplazados libremente en el espacio virtual utilizando las manos del usuario.

### 3.4.2. Tareas realizadas

Durante este sprint, se llevaron a cabo una serie de actividades destinadas a familiarizarse y dominar el uso de las gafas Meta Quest 3, así como adaptar las escenas del proyecto a un entorno interactivo basado en realidad virtual con control mediante mandos y manos.

- **Familiarización inicial con el dispositivo:**

Se comenzó con el uso lúdico de las gafas, experimentando con varios juegos y experiencias disponibles en la plataforma. Esta fase permitió conocer la interfaz de usuario, la navegación dentro del sistema operativo de Meta Quest 3 y las posibilidades de interacción con el entorno virtual.

- **Primeros ejercicios con los mandos de las gafas:**

A continuación, se desarrollaron pequeñas escenas interactivas con el objetivo de aprender a utilizar los mandos como dispositivos de entrada. Entre los ejercicios destacan:

- Una escena en la que, al pulsar un botón del menú, se añadía una caja en una posición aleatoria dentro del entorno.
- Una escena con una esfera que, al ser pulsada con los botones del mando, cambiaba tanto de color como de posición.

- **Primeros experimentos con interacción mediante manos:**

Tras comprender el funcionamiento de los mandos, se pasó a trabajar con el reconocimiento de manos. Se crearon escenas sencillas en las que las manos actuaban como elementos de interacción, comprobando su correcta detección, colisión con objetos y respuesta al contacto.

- **Implementación de una escena avanzada con manos:**

Como ejercicio de integración, se desarrolló una escena compleja que reproduce la lógica de menús del sprint anterior, pero esta vez con control total a través de las manos. Se implementó interacción táctil con botones mediante componentes pressable y clickable, así como la funcionalidad de agarrar y mover los menús dentro del entorno virtual (funcionalidad grabbable), lo cual supuso uno de los mayores retos técnicos del sprint.

### 3.4.3. Resultados obtenidos

- **Resultados de escenas con mandos como controladores:** Como parte del proceso de adaptación del proyecto a entornos de realidad virtual utilizando las Meta Quest 3, se desarrollaron diversas escenas interactivas orientadas a comprender y controlar la interacción mediante los mandos del dispositivo. A continuación, se detallan los resultados obtenidos en dos ejercicios específicos:
  - Cambio de color mediante interacción con láser y gatillo: En este primer ejercicio, se implementó una escena sencilla con una esfera colocada sobre un plano. Los mandos de las Meta Quest 3 emiten un rayo láser visible en el entorno, que permite apuntar y seleccionar elementos interactivos. Al enfocar la esfera con dicho láser y presionar uno de los gatillos del mando, se activaba un cambio de color en la esfera. Este ejercicio permitió comprender el funcionamiento del sistema de puntero láser, así como el uso de eventos asociados a los botones del mando, elementos clave para la navegación e interacción en escenas VR.

Resultado:

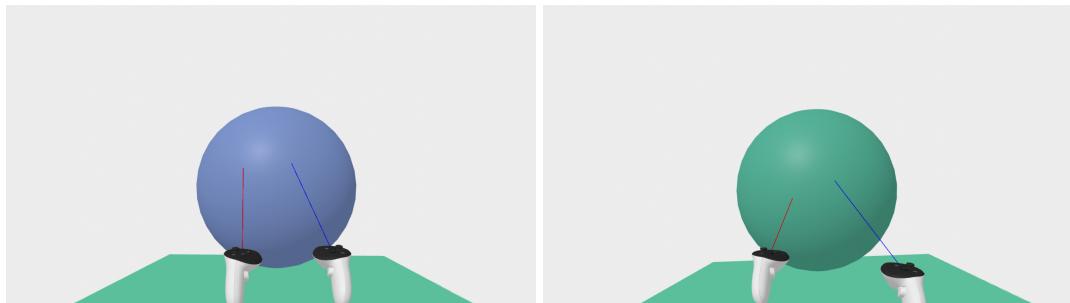


Figura 3.9: Escena con mandos como controladores. Esfera con cambio de color.

- Creación de objetos mediante botón del mando: En un segundo ejercicio se diseñó un pequeño menú que incluía un botón virtual etiquetado. Utilizando también el sistema de láser de los mandos, si el usuario enfocaba dicho botón y presionaba la tecla X, se generaba dinámicamente una caja en una posición aleatoria dentro de la escena. Este ejercicio demostró la posibilidad de crear elementos interactivos que respondan a acciones específicas de los mandos, así como la integración de lógi-

ca condicional para posicionamiento aleatorio, reforzando los conocimientos sobre manipulación del DOM de A-Frame en entornos de VR.

Resultado:



Figura 3.10: Escena con mandos como controladores. Menu con creación de cajas.

#### ■ Resultados de escenas con manos como controladores:

Una vez superada la fase de interacción mediante mandos, el enfoque principal del sprint se centró en la integración y experimentación con el seguimiento de manos en las Meta Quest 3, utilizando WebXR y las capacidades nativas del visor. Los ejercicios realizados han permitido comprobar el potencial de las manos como método de interacción natural, sin necesidad de dispositivos físicos. A continuación, se detallan los resultados más relevantes:

- Mini menú con selección de figuras y funcionalidades extra: En este primer ejercicio se diseñó un menú interactivo totalmente controlado mediante las manos. El menú contaba con tres botones visuales representados por una caja, una esfera y un toroide. Al "presionar" uno de estos botones con los dedos, se generaba en la escena la figura geométrica correspondiente. Además, el menú incorporaba un slider para ajustar dinámicamente el tamaño de los objetos generados y un botón de "modo oscuro", que alternaba la iluminación de la escena entre una ambientación diurna y otra nocturna. Este ejercicio permitió explorar múltiples componentes de interacción: detección de eventos táctiles con la mano, control de parámetros mediante deslizadores y modificación de propiedades globales de la escena en tiempo real.

Resultado:

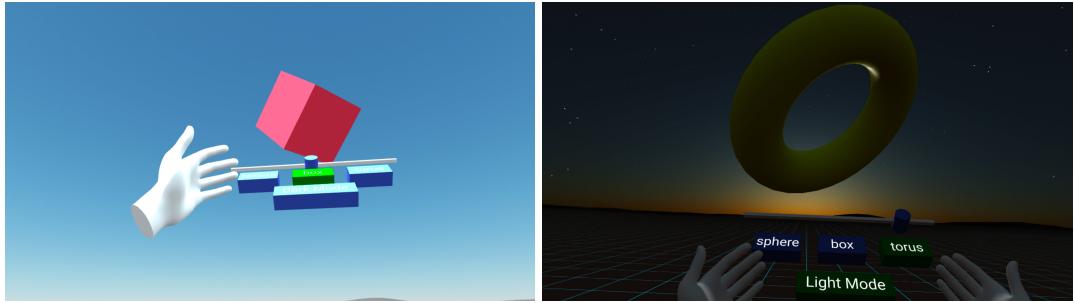


Figura 3.11: Escena con manos como controladores. Mini menu.

- Panel agarrible y desplazable: En este segundo ejercicio se construyó un panel rectangular interactivo que podía ser agarrado y desplazado por el entorno utilizando gestos naturales de las manos. Una vez detectado el gesto de agarre, el panel se adhiere a la mano y puede moverse libremente. Además, se implementó una respuesta visual: los bordes del panel cambiaban de color cuando estaba siendo agarrado, proporcionando retroalimentación al usuario. Esta escena sirvió para profundizar en el uso del componente `grabbable`, así como en la detección de colisiones y eventos de interacción táctil avanzados.

Resultado:

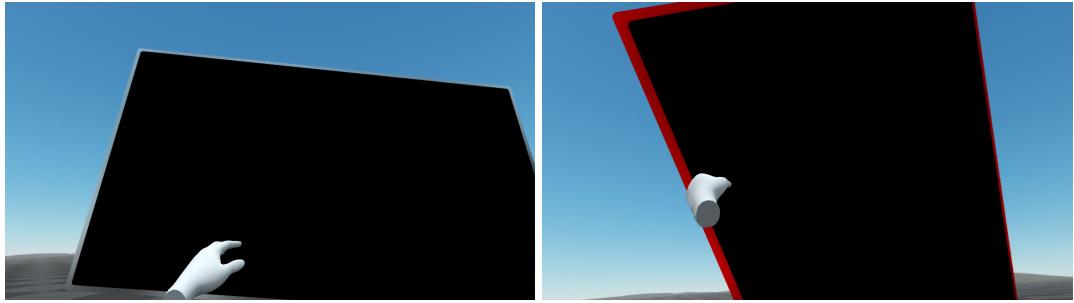


Figura 3.12: Escena con mandos como controladores. Panel agarrible.

- Primer prototipo del menú interactivo con manos: Como parte del desarrollo progresivo de la escena principal basada en la navegación por menús (iniciada en el Sprint 3), se creó un primer prototipo adaptado al control por manos. En esta versión preliminar, algunas funcionalidades del menú original ya eran accesibles mediante gestos manuales, si bien no estaban completamente finalizadas. Esta escena sentó las bases de la implementación definitiva desarrollada en el siguiente sprint.

Resultado:

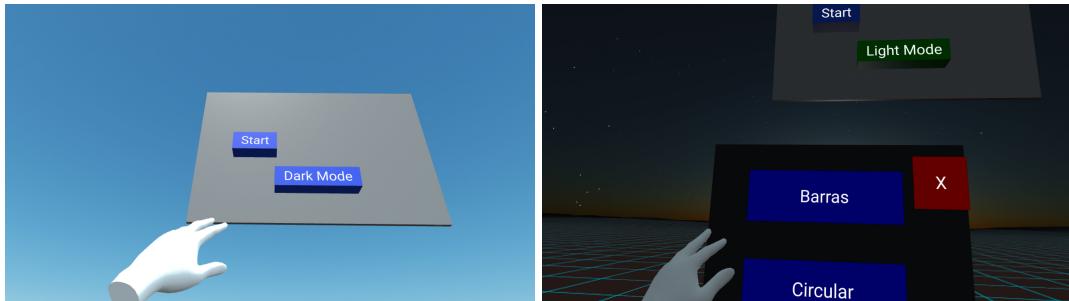


Figura 3.13: Escena con mandos como controladores. Prototipo de menú final.

#### 3.4.4. Conclusiones del sprint

Durante este sprint se ha dado un paso fundamental en la transición del entorno de desarrollo de escritorio hacia un entorno inmersivo en realidad virtual, empleando las gafas Meta Quest 3 como plataforma de interacción principal. A lo largo de esta fase, se ha logrado una familiarización progresiva con el dispositivo, primero mediante el uso de mandos y posteriormente con el sistema de seguimiento de manos, que ha sido el foco principal del trabajo.

Se han comprendido y aplicado con éxito las principales técnicas de interacción manual en A-Frame y WebXR, incluyendo la detección de gestos, la pulsación de botones (pressable), el agarre de elementos (grabbable), y la manipulación de componentes dinámicos dentro del entorno virtual. Estos avances han permitido llevar a cabo una serie de ejercicios prácticos que no solo han servido para validar el funcionamiento de estas tecnologías, sino también como base para el diseño e implementación de la escena principal del proyecto.

Además, se ha desarrollado un primer prototipo funcional del sistema de menús interactivos adaptado al uso de manos, que será completado y refinado en el siguiente sprint. Esta implementación ha supuesto un reto tanto técnico como conceptual, especialmente en lo relativo a la integración fluida de los distintos componentes y a la detección precisa de los gestos de usuario.

En conclusión, este sprint ha representado un punto de inflexión en el proyecto, permitiendo establecer las bases técnicas necesarias para la fase final de desarrollo. Se ha demostrado que es posible implementar una experiencia interactiva completa sin necesidad de controladores físicos, lo que acerca el proyecto a una experiencia de usuario más natural e inmersiva.

## 3.5. Sprint 5

### 3.5.1. Objetivos del sprint

El objetivo principal de este Sprint 5 ha sido el diseño y desarrollo de un modelo de menú interactivo modular en realidad virtual, el cual servirá como base reutilizable para distintas escenas que se implementarán en el próximo sprint (Sprint Final).

Este menú ha sido construido con un enfoque flexible, capaz de adaptarse a múltiples contextos y tipos de interacción en entornos inmersivos. Su arquitectura permite gestionar menús jerárquicos, botones interactivos y elementos de control (como sliders), todo ello mediante interacción natural con las manos, aprovechando tecnologías de hand tracking.

Un aspecto fundamental de este sistema es que toda la estructura de menús y botones se genera dinámicamente a partir de un archivo .json, lo que proporciona una clara separación entre los datos y la lógica de funcionamiento. Esto permite una alta modularidad, escalabilidad y facilidad de mantenimiento, ya que cualquier cambio en la interfaz puede realizarse desde el archivo de configuración sin modificar el código fuente.

Durante este sprint también se han implementado y consolidado funcionalidades esenciales como el minimizado y maximizado de menús, el control de visibilidad y navegación entre submenús, la gestión de estados mediante booleanos y la asignación automática de funciones a botones según su definición en el .json.

Este modelo está preparado para ser aplicado en diferentes experiencias VR, y será el núcleo sobre el que se construirán las distintas escenas del proyecto en el Sprint Final.

### 3.5.2. Tareas realizadas

Durante este sprint se llevaron a cabo las siguientes tareas clave:

- Diseño y estructuración del menú principal y sus submenús reutilizables en múltiples aplicaciones, siguiendo una lógica jerárquica que permite al usuario navegar mediante la interacción con las manos hasta llegar a una representación visual de los datos.
- Implementación de interacciones avanzadas mediante el uso de manos, utilizando las funcionalidades de WebXR y BabiaXR para detectar gestos y activar los botones de los menús (pressables y grabbables).

- Desarrollo de la funcionalidad para representar datos filtrados en diagramas de barras o circulares, a través del uso de la librería BabiaXR. Los filtros incluyen opciones como tipo de motor, color del vehículo, número de puertas, o representación completa sin filtros.
- Implementación del sistema de generación dinámica de menús a partir de un archivo .json, que contiene toda la información de los elementos a representar en la escena (textos, posiciones, funciones, eventos asociados, etc.). Se desarrollaron funciones específicas para leer el archivo y construir las entidades de forma automática.
- Pruebas intensivas de funcionalidad en Meta Quest 3, verificando que todas las interacciones se comportaran correctamente en el entorno de realidad virtual.
- Ajustes de diseño y experiencia de usuario, asegurando una navegación fluida y clara entre menús, así como una representación visual efectiva de los datos.
- Mejora de la estética de la escena, optimizando el aspecto visual de los menús, botones y representaciones gráficas de los diagramas. Esto incluye la mejora de texturas, colores, iluminación y detalles visuales que favorecen la interacción y mejoran la experiencia general del usuario en realidad virtual.

### 3.5.3. Resultados obtenidos

Durante este Sprint se ha desarrollado un modelo de menú interactivo reutilizable diseñado para entornos de realidad virtual, el cual será utilizado en diferentes aplicaciones en el siguiente Sprint. Este sistema de menús representa una base sólida, modular y flexible, que permite construir experiencias VR con interacción natural mediante el uso de las manos.

Entre las funcionalidades principales implementadas destacan:

- Navegación entre menús y submenús, con la posibilidad de avanzar y retroceder dentro de una estructura jerárquica.
- Minimizado y maximizado de menús, permitiendo una gestión dinámica del espacio visual.
- Cambio entre modos de iluminación, conmutando entre escenas de día y de noche.

- Interacción mediante hand tracking, compatible con dispositivos como Meta Quest 3.
- Creación automática de la interfaz a partir de un archivo .json, lo que permite definir menús, botones y funciones sin necesidad de modificar el código fuente.
- Capacidad de agarrar (grabbable) y mover los menús dentro de la escena usando las manos.
- Flexibilidad para incorporar nuevas funcionalidades a través de funciones definidas en el propio menu.js, las cuales se pueden asociar fácilmente a cualquier botón desde el archivo de configuración.

Este sistema se ha diseñado con una arquitectura escalable y mantenible, enfocada en la reutilización en múltiples escenas. En el siguiente sprint se aplicará este modelo para construir distintas experiencias interactivas que demostrarán su versatilidad.

El resultado visual obtenido en este Sprint puede observarse en la siguiente imagen:



Figura 3.14: Aspecto final del menú interactivo

### 3.5.4. Conclusiones del sprint

El Sprint 5 ha supuesto un punto de inflexión en el desarrollo del proyecto, al centrar todos los esfuerzos en la creación de un modelo de menú interactivo robusto, flexible y adaptable a múltiples contextos de uso. A diferencia de fases anteriores, en este sprint el enfoque ha estado más orientado a la construcción de una herramienta base reutilizable, capaz de integrarse en

diferentes escenas con mínimas modificaciones y conservando una lógica común de funcionamiento.

Durante esta etapa se ha trabajado intensamente en consolidar un sistema que permita separar completamente los datos de la lógica visual e interactiva. Esta decisión ha aportado una mayor modularidad al proyecto, facilitando tanto su mantenimiento como su escalabilidad a largo plazo. El menú desarrollado no es una escena cerrada, sino una infraestructura sobre la que se podrán construir múltiples experiencias interactivas en los siguientes sprints.

Además, se ha logrado una integración coherente de los distintos scripts que conforman la arquitectura del sistema, asegurando que el conjunto funcione como una unidad fluida y eficaz en entornos de realidad virtual. Esto ha requerido una planificación cuidadosa del flujo de datos, de las referencias a objetos, y del comportamiento dinámico de los elementos en la escena.

Este sprint no solo deja como resultado un sistema técnico consolidado, sino también una base metodológica clara para abordar los desarrollos futuros: construir escenas sobre una única arquitectura común, reutilizable, coherente y fácilmente configurable. Así, el proyecto entra en su fase final con una herramienta funcional completa y validada, lista para ser aplicada en diversas demostraciones prácticas.

## 3.6. Sprint Final

### 3.6.1. Objetivos del sprint

El objetivo de este sprint final es aplicar el sistema de menús interactivos desarrollado previamente en el Sprint 5 a diferentes contextos funcionales, demostrando así su versatilidad, modularidad y potencial de reutilización en distintos entornos inmersivos. Tras la creación y validación de un prototipo funcional de menú basado en A-Frame y BabiaXR, este sprint se centra en trasladar dicho sistema a tres casos de uso diferenciados que simulan aplicaciones prácticas del menú en experiencias tanto de realidad virtual como de realidad aumentada.

Cada caso de uso representa un tipo distinto de interacción o necesidad que podría resolverse mediante este sistema de menús, abarcando desde la exploración y visualización de datos hasta la personalización de entornos virtuales o la reproducción de contenidos multimedia. Los tres escenarios diseñados en este sprint permiten validar el funcionamiento del menú en distintas

situaciones, con distintos niveles de complejidad, al mismo tiempo que se exploran sus límites, se identifican mejoras potenciales y se ponen a prueba los conocimientos adquiridos durante el desarrollo del TFG.

Los tres casos de uso desarrollados son:

- **Caso de uso 1: Habitación de los diagramas**

En este entorno, el usuario puede navegar por un conjunto de menús que permiten filtrar información hasta llegar a representaciones visuales a través de distintos tipos de diagramas. El menú actúa como herramienta de consulta y visualización dinámica, facilitando el acceso a distintos niveles de datos según los criterios seleccionados.

- **Caso de uso 2: Habitación a diseñar (VR y AR)**

Este caso de uso permite amueblar y decorar una habitación virtual seleccionando objetos a través del menú, que ofrece distintas categorías de mobiliario y decoración. Además de la versión para realidad virtual, se ha desarrollado también una versión en realidad aumentada, lo que supone un paso más en términos de accesibilidad y contexto de uso, adaptando la experiencia a distintos dispositivos y entornos físicos reales.

- **Caso de uso 3: Habitación de los sonidos**

En esta escena, el menú permite reproducir sonidos relajantes y bandas sonoras de películas. Cada sonido está vinculado a un objeto representativo que refuerza visualmente la experiencia sonora. Este caso demuestra la capacidad del sistema para gestionar contenidos multimedia, así como la posibilidad de ofrecer experiencias inmersivas centradas en el audio y la ambientación.

En conjunto, este sprint busca consolidar el sistema de menús como un componente clave reutilizable en múltiples contextos interactivos, sirviendo como prueba de concepto para aplicaciones más amplias que podrían desarrollarse a partir de esta base.

### **3.6.2. Tareas realizadas**

Durante el desarrollo del Sprint Final, se llevaron a cabo una serie de tareas orientadas a aplicar el prototipo de menú interactivo, desarrollado en el Sprint 5, a tres escenarios diferenciados. Estas tareas permitieron validar su funcionalidad en contextos diversos, así como extender su

uso para cubrir necesidades concretas en entornos inmersivos. Las tareas se agrupan por cada uno de los tres casos de uso desarrollados:

- **Caso de uso 1: Habitación de los diagramas**

- Adaptación del menú para la navegación por distintas categorías de datos.
- Implementación de filtros que permiten seleccionar distintos conjuntos de información a visualizar.
- Diseño y creación de varios tipos de diagramas (de barras, de líneas, circulares...) que se generan dinámicamente en función de la información seleccionada por el usuario.
- Integración visual de los diagramas en la escena para mantener la coherencia espacial y estética.
- Pruebas funcionales para asegurar que la interacción con los menús y los filtros produce el resultado visual esperado.

- **Caso de uso 2: Habitación a diseñar (VR y AR)**

- Creación de un sistema de categorías de mobiliario (mesas, sillas, estanterías...) accesible desde el menú.
- Implementación de funcionalidades que permiten instanciar y posicionar objetos dentro de la escena mediante selección desde el menú.
- Inclusión de elementos decorativos adicionales (cuadros, plantas, alfombras...) que enriquecen la experiencia visual.
- Desarrollo de una versión específica en realidad aumentada (AR) mediante WebXR, adaptando el sistema a dispositivos móviles y garantizando una colocación realista de objetos sobre superficies físicas.
- Ajustes específicos para la versión en realidad virtual (VR), optimizando la experiencia con controladores y movimientos de cabeza.
- Pruebas cruzadas entre versiones VR y AR para asegurar la coherencia funcional y visual en ambas plataformas.

#### ■ Caso de uso 3: Habitación de los sonidos

- Creación de un sistema de menú para la selección de sonidos divididos en dos categorías principales: sonidos relajantes y bandas sonoras de películas.
- Carga e integración de distintos archivos de audio en la escena.
- Implementación de lógica de reproducción, pausa y cambio de pista mediante selección desde el menú.
- Asociación de cada sonido a un objeto representativo dentro de la escena, lo que permite una relación visual-auditiva que mejora la experiencia inmersiva.
- Ajuste de volumen, bucle y sincronización para garantizar una experiencia auditiva fluida.
- Testeo en dispositivos de VR para asegurar el correcto funcionamiento del audio espacial.

#### ■ Tareas generales del sprint

- Revisión y limpieza del código del menú para facilitar su reutilización en los distintos entornos.
- Refactorización de componentes comunes entre escenas para mantener una arquitectura modular y escalable.
- Verificación de compatibilidad entre BabiaXR y los nuevos elementos añadidos.
- Pruebas de usabilidad básicas en cada escena para asegurar una experiencia intuitiva.
- Documentación de cada uno de los casos de uso para su correcta descripción en la memoria del TFG.

### 3.6.3. Resultados obtenidos

El Sprint Final ha permitido validar y consolidar el sistema de menús desarrollado previamente, aplicándolo con éxito en tres escenarios diferenciados que han servido como demostraciones prácticas de su versatilidad y escalabilidad. Los resultados más relevantes obtenidos son los siguientes:

- Reutilización efectiva del sistema de menús: Se ha demostrado que el menú diseñado en el Sprint 5 es modular y adaptable a distintos contextos, lo que permite su integración sin necesidad de grandes modificaciones estructurales.
- Versatilidad del menú en tareas diversas: Las tres demos han evidenciado que el sistema es capaz de manejar tareas muy distintas entre sí, como la selección de datos para visualización, la colocación de objetos tridimensionales y la gestión de contenido sonoro.
- Generación dinámica de visualizaciones: En la habitación de los diagramas, el usuario puede filtrar y representar visualmente distintos conjuntos de datos en tiempo real, seleccionando el tipo de gráfico que mejor se adapta a sus necesidades.
- Aplicación multiplataforma (VR y AR): En la habitación a diseñar, se ha logrado implementar con éxito el sistema tanto en realidad virtual como en realidad aumentada. En VR se ha optimizado la interacción con controladores, mientras que en AR se ha validado la colocación realista de objetos sobre superficies del entorno físico mediante dispositivos móviles.
- Mejora de la experiencia inmersiva mediante audio: En la habitación de los sonidos, se ha integrado un sistema auditivo interactivo que permite al usuario elegir entre diferentes tipos de sonido, cada uno acompañado por un objeto representativo, enriqueciendo así la experiencia sensorial y explorando nuevas formas de interacción.
- Robustez y consistencia en la navegación: En todas las demos se ha confirmado que el sistema de menús mantiene una experiencia de usuario coherente, intuitiva y estable, independientemente del contexto en el que se aplique.
- Demostración del potencial de BabiaXR: La integración fluida de estas funcionalidades con BabiaXR refuerza la utilidad de esta librería para el desarrollo rápido de prototipos interactivos en realidad virtual y aumentada usando tecnologías web.
- Base sólida para desarrollos futuros: Los tres casos de uso no solo validan la solución técnica, sino que también abren la puerta a futuras extensiones del sistema hacia nuevos entornos, funcionalidades o tipos de contenido.

### 3.6.4. Conclusiones del sprint

Este Sprint Final ha supuesto un cierre práctico y funcional del proyecto, al aplicar de forma efectiva el sistema de menús desarrollado en el Sprint anterior en tres contextos distintos. Gracias a esta fase, se ha podido confirmar que el menú no solo cumple con los requisitos iniciales de navegación e interacción, sino que también posee una alta capacidad de adaptación a diferentes tipos de escenas y casos de uso.

A nivel técnico, se ha constatado la solidez de la arquitectura del menú, permitiendo su reutilización sin tener que rediseñar su estructura interna para cada demo. Esto refuerza la idea de que una buena planificación modular y orientada a la reutilización puede facilitar en gran medida la escalabilidad de proyectos interactivos.

El desarrollo de las tres habitaciones —diagramas, diseño y sonidos— ha puesto a prueba el sistema en tareas de distinta naturaleza (visualización, interacción con objetos, reproducción de sonido), lo que ha servido para validar su versatilidad. Especial mención merece la implementación en realidad aumentada, ya que ha permitido comprobar que la solución no solo es válida en entornos de realidad virtual, sino también en entornos físicos con elementos digitales superpuestos, ampliando así su rango de aplicación.

Además, este sprint ha servido como una pequeña "prueba de concepto" de cómo este tipo de menús podría ser utilizado en experiencias educativas, de entretenimiento o de diseño de espacios, entre otras. Las lecciones aprendidas en esta fase serán clave si el proyecto se quiere ampliar o adaptar a nuevas funcionalidades en el futuro.

En definitiva, el Sprint Final ha permitido cerrar el proyecto con una demostración clara del potencial del sistema, abriendo al mismo tiempo una puerta a su evolución futura en nuevos escenarios y con nuevas capacidades interactivas.

# **Capítulo 4**

## **Resultados**

### **4.1. Descripción funcional**

El proyecto desarrollado consiste en una aplicación inmersiva de visualización de datos en realidad virtual utilizando tecnologías web como A-Frame, WebXR y BabiaXR. Las escenas están diseñadas para ser accesible desde distintos dispositivos, aunque su funcionalidad completa se experimenta mediante el uso de gafas de realidad virtual Meta Quest 3, aprovechando especialmente la interacción con las manos como principal método de navegación y control.

El sistema permite al usuario acceder a un menú principal en el que, mediante botones interactivos, puede navegar por distintos submenús hasta llegar a la selección y representación de los datos. Esta visualización se realiza mediante diagramas circulares o de barras, generar formas como camas, sillas o mesas a partir de archivos .glb, todo ello generado a partir de un archivo JSON que contiene los datos estructurados.

Todos los elementos de la interfaz como los botones, menús, sliders, etc han sido diseñados para responder al tacto utilizando las manos del usuario gracias al soporte de WebXR Hand Tracking. Además, se ha añadido la funcionalidad de agarrar y mover los paneles (grabbable), permitiendo así una experiencia de usuario más personalizada y flexible dentro del entorno virtual.

La interacción principal del proyecto se basa en un enfoque intuitivo y visual, en el que el usuario puede explorar y analizar información de forma natural, manipulando elementos virtuales con sus propias manos. Esto convierte la aplicación en una herramienta tanto educativa como interactiva, pensada para demostrar el potencial de la visualización de datos en entornos

de realidad virtual accesibles mediante tecnologías web abiertas.

### Estructura general y funcionamiento

El punto de entrada principal del proyecto es el archivo index.html, que se encarga de importar todas las librerías necesarias para el funcionamiento de la escena. Entre las más importantes, destacan:

- aframe.min.js: biblioteca base para crear escenas WebVR/WebXR con A-Frame.
- aframe-hand-tracking-controls-extras.js y hand-tracking-controls-extras-components.js: utilizadas para habilitar y mejorar la detección y el seguimiento de manos en dispositivos VR compatibles.
- aframe-environment-component.js: permite crear un entorno envolvente de forma rápida, como una habitación.
- Otras librerías visuales como aframe-rounded.js, necesarias para elementos con estética personalizada.

Además, en este archivo se importan varios scripts clave que otorgan funcionalidad a la escena:

- menu.js (el script principal)
- initMenu.js
- button.js
- pressable.js
- pinchable.js
- size-change.js
- slider.js

Dentro de la etiqueta a-scene, se instancia la componente personalizada menu, que está definida en menu.js, y se configuran los controladores de entrada (ratón, manos, mandos VR), así como el entorno visual de la escena (habitáculo con paredes, techo, suelo, ventanales y fondo fotográfico).

### Componentes clave y su funcionalidad

■ **menu.js** Este es el componente más importante del proyecto. Se encarga de:

- Leer el archivo scene.json donde se define toda la estructura de la escena (menús, submenús, botones, funciones, posiciones, etc.).
- Crear automáticamente los menús y botones especificados.
- Asociar las funciones correctas a cada botón según su configuración en el JSON.
- Controlar funcionalidades complejas como:
  - Minimizar y maximizar menús.
  - Cambiar de un menú a otro.
  - Cambiar entre modos (día/noche).
  - Posicionar y rotar los menús en función del último estado.
  - Activar/desactivar interactividad de los menús (grabbable).
- Utiliza booleanos internos para gestionar el estado de visibilidad y navegación entre los distintos elementos de la escena.
- Permite extender la lógica funcional con funciones específicas para cada demo o escenario.

■ **scene.json** Este archivo es el corazón de la configuración dinámica. En él se define:

- Cada menú de la escena con propiedades como ID, visibilidad, posición y si es agarrable.
- Cada botón con:
  - Label o texto visible (o "noLabel").
  - Posición en el menú.
  - Posición del texto (label).
  - Tamaño y color.
  - Funciones asociadas y sus parámetros (por ejemplo: function1, parameter1F1, etc.).

Este archivo es leído automáticamente por menu.js para crear la interfaz interactiva sin tener que programar cada botón manualmente.

- **button.js** Define la apariencia y comportamiento de los botones:
  - Tamaño, color y forma.
  - Label (etiqueta de texto), posición y tamaño del mismo.
  - Posición y rotación del botón.
  - Capacidad de interacción mediante ratón o manos (detección por VR).
- **initMenu.js** Se encarga de definir la apariencia base de cada menú. Los menús se representan como paneles con esquinas redondeadas, color gris metálico y tamaño configurable (ancho, alto, radio de borde). Esta componente es la que otorga la estructura visual a cada menú.
- **pressable.js** Añade la capacidad de detectar presiones físicas sobre botones, simulando una experiencia realista de "tocar un botón". Es esencial para la interacción tanto con las manos como con el ratón.
- **pinchable.js** Habilita la capacidad de realizar gestos de "pinch" (pellizco) sobre elementos, especialmente diseñada para el uso con sliders. Soporta también la interacción con ratón. Es indispensable para cambiar valores deslizando.
- **size-change.js** Permite cambiar el tamaño de cualquier elemento de la escena cuando se usa un slider vinculado. Interpreta el valor actual del slider y modifica la escala de la entidad correspondiente.
- **slider.js** Es la definición funcional y visual de un slider interactivo. Combina pinchable.js y size-change.js para permitir que el usuario pueda ajustar tamaños de objetos de forma intuitiva. En este proyecto, su función principal es controlar el tamaño de los diagramas generados.

### Guía de uso del sistema (desde cero)

Para utilizar este sistema desde cero, un desarrollador debe seguir los siguientes pasos:

- 1. Crear un index.html donde se importen las librerías básicas (A-Frame, environment, hand-tracking, etc.) y los scripts necesarios (menu.js, initMenu.js, button.js, etc.).

- **2.** En la etiqueta `<a-scene>`, añadir la componente `menu`, y definir los controladores de entrada que se van a usar (ratón, manos, mandos VR).
- **3.** Diseñar el archivo `scene.json` donde se describa:
  - Qué menús existirán.
  - Qué botones tendrá cada menú.
  - Qué funciones ejecuta cada botón.
  - Posiciones, visibilidades, labels, colores, etc.
- **4.** Desarrollar funciones adicionales dentro de `menu.js` si se desea ampliar la funcionalidad base. Por ejemplo, para activar música, crear gráficos, cambiar colores, etc.
- **5.** Incluir los scripts `pressable.js`, `pinchable.js`, `slider.js` y `size-change.js` si se desea dotar de interacción avanzada a los botones y sliders.

## 4.2. Arquitectura general

La arquitectura general del proyecto se basa en una estructura modular y escalable que permite la creación dinámica de interfaces de usuario interactivas en entornos de realidad virtual, utilizando A-Frame como framework principal y tecnologías complementarias como BabiaXR y WebXR.

En esta sección se describen los scripts fundamentales que componen la base del sistema: `index.html`, `menu.js` y `scene.json`. Estos tres archivos constituyen el núcleo estructural del proyecto. Juntos, definen la lógica de carga de la escena, la generación dinámica de los elementos interactivos y el modelo de datos que describe el contenido y comportamiento de los menús en la experiencia virtual.

Además de estos archivos principales, el sistema se apoya en otros componentes personalizados como `initMenu.js`, `button.js`, `pinchable.js` y `pressable.js`, que enriquecen la interacción del usuario con funcionalidades específicas como detección de gestos, pulsaciones, arrastre o agarre de elementos. Esta arquitectura permite un alto grado de reutilización y personalización, lo que facilita la adaptación del sistema a una gran variedad de aplicaciones y contextos.

Gracias a esta base, he podido desarrollar tres demos distintas, cada una con funcionalidades específicas, las cuales serán descritas más adelante. No obstante, el modelo de menú interactivo implementado puede extenderse y reutilizarse fácilmente en múltiples escenarios y casos de uso, lo que lo convierte en una herramienta versátil para la creación de experiencias inmersivas.

A continuación, se muestra un diagrama representativo de la arquitectura general del proyecto, donde se visualizan los componentes principales y su relación entre ellos.

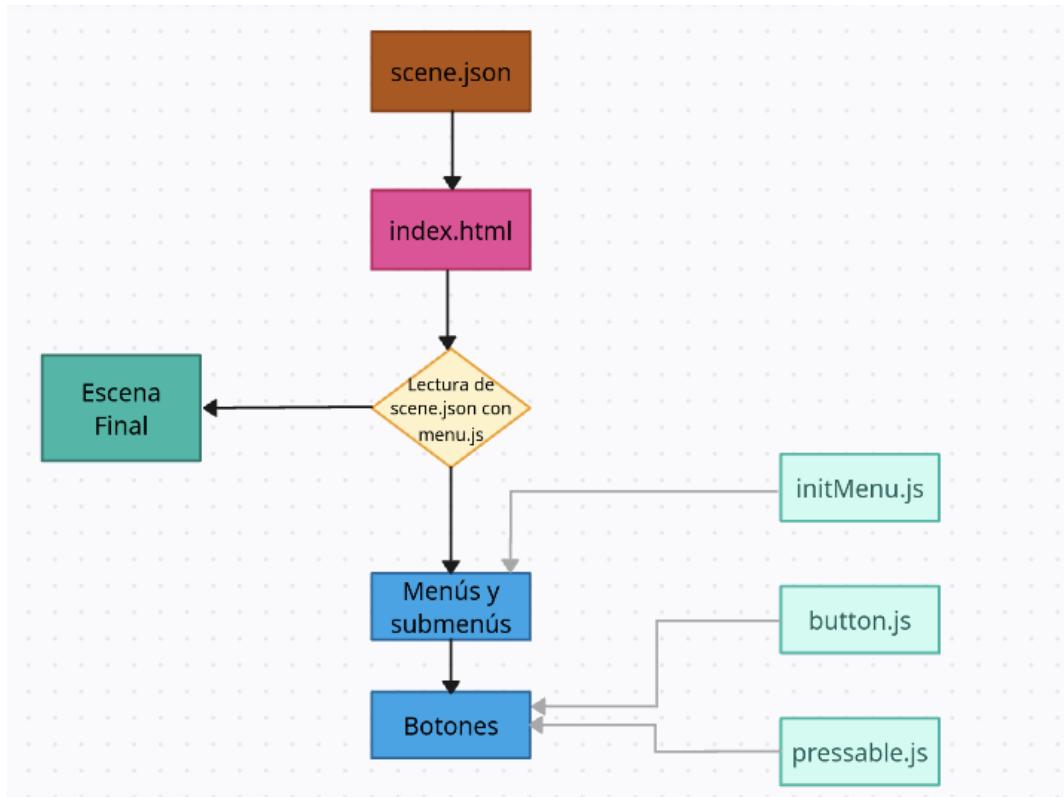


Figura 4.1: Arquitectura General.

#### 4.2.1. index.html

El archivo `index.html` actúa como punto de entrada principal para ambas demos desarrolladas. En este archivo se definen las configuraciones iniciales y se realiza la carga de los distintos componentes, scripts y librerías necesarias para el funcionamiento de la escena en realidad virtual. A continuación, se detallan sus elementos principales:

- Carga de librerías externas En la cabecera del documento se importan varias librerías esenciales para la construcción de la escena:

- A-Frame: Framework principal utilizado para construir la escena en realidad virtual usando HTML y componentes personalizados.
- BabiaXR: Librería que permite la representación de datos en la escena mediante gráficos interactivos como diagramas circulares o de barras.
- A-Frame Environment Component: Proporciona un entorno 3D prediseñado para la escena (por ejemplo, cielo, iluminación, suelo).
- Librerías de seguimiento de manos de WebXR: Permiten detectar y utilizar las manos del usuario como método de interacción dentro de la escena VR.

Estas librerías son imprescindibles para poder integrar todas las funcionalidades avanzadas de interacción y visualización que requiere el proyecto. En un subapartado posterior, se explicará con más detalle cada una de estas librerías y su función específica.

- Importación de scripts personalizados El index.html también enlaza varios archivos JavaScript desarrollados específicamente para el proyecto:
  - menu.js: Este script es uno de los elementos centrales del sistema. Se encarga de crear dinámicamente todos los menús y submenús definidos en un archivo externo llamado scene.json, que contiene la estructura lógica y visual de cada escena.
  - initmenu.js: Gestiona la inicialización de los menús y otros elementos interactivos en el momento de cargar la escena.
  - button.js: Define la funcionalidad básica de los botones utilizados en los menús.
  - pressable.js: Añade la funcionalidad necesaria para que los botones sean detectados como pulsables mediante interacción con manos o controladores.

Cada uno de estos scripts será detallado en secciones posteriores, donde se explicará su lógica interna y su papel en el sistema.

- Definición de controladores y entrada Dentro de la escena de A-Frame, se definen explícitamente los elementos que permiten la interacción del usuario con el entorno:
  - Mano izquierda (hand-left) y mano derecha (hand-right): Entidades que representan las manos del usuario dentro del entorno VR, permitiendo la interacción mediante

gestos, pulsaciones o agarres. Estas entidades utilizan el sistema de detección de manos proporcionado por WebXR.

- Cursor de ratón: Aunque el enfoque principal está orientado al uso de manos, también se define un cursor tradicional controlado con el ratón para poder acceder a la escena desde dispositivos que no cuenten con gafas de realidad virtual o seguimiento de manos.

La inclusión de múltiples métodos de entrada permite garantizar la accesibilidad y pruebas del sistema desde diferentes plataformas durante el proceso de desarrollo.

## Librerías importadas

Para el correcto funcionamiento e interacción de la escena desarrollada, se han utilizado varias librerías externas que amplían las funcionalidades básicas del framework A-Frame. A continuación, se detalla el propósito y utilidad de cada una de ellas:

### ■ A-Frame:

**Forma:** [aframe.io/releases/1.6.0/aframe.min.js](https://aframe.io/releases/1.6.0/aframe.min.js)

Esta es la librería principal sobre la que se construye toda la escena. A-Frame es un framework basado en HTML y WebGL que permite crear experiencias inmersivas de realidad virtual directamente desde el navegador. Permite definir entidades 3D, aplicar materiales, animaciones, y añadir lógica mediante componentes personalizados.

### ■ A-Frame Environment Component:

**Forma:** [unpkg.com/aframe-environment-component@1.5.x/dist/aframe-environment-component.min.js](https://unpkg.com/aframe-environment-component@1.5.x/dist/aframe-environment-component.min.js)

Este componente ofrece un entorno predefinido para la escena (por ejemplo, cielo, iluminación ambiental, terreno, etc.). Facilita crear escenas visualmente más atractivas y realistas sin necesidad de construir manualmente todo el entorno desde cero.

### ■ A-Frame Babia Components:

**Forma:** [unpkg.com/aframe-babia-components/dist/aframe-babia-components.min.js](https://unpkg.com/aframe-babia-components/dist/aframe-babia-components.min.js)

BabiaXR es una librería que permite visualizar datos en realidad virtual mediante la creación de gráficos interactivos como diagramas de barras o circulares. Estos componentes

están preparados para recibir datos (por ejemplo, desde un archivo .json) y representarlos de forma intuitiva en la escena, facilitando la exploración e interpretación de información.

- **Hand Tracking Controls Extras:**

**Forma:** [gftruj.github.io/hand.tracking.controls.extras/dist/aframe-hand-tracking-controls-extras.js](https://gftruj.github.io/hand.tracking.controls.extras/dist/aframe-hand-tracking-controls-extras.js)

Esta librería añade funcionalidades adicionales al sistema de seguimiento de manos de A-Frame/WebXR. Permite detectar gestos, realizar interacciones más precisas y enriquecer el control con las manos, mejorando significativamente la experiencia en dispositivos como las Meta Quest.

- **Hand Tracking Controls Extras - Components:**

**Forma:** [gftruj.github.io/hand.tracking.controls.extras/components/dist/hand-tracking-controls-extras-components.js](https://gftruj.github.io/hand.tracking.controls.extras/components/dist/hand-tracking-controls-extras-components.js)

Completa la librería anterior, ofreciendo componentes listos para usar como botones pulsables con las manos (pressable), elementos que pueden ser agarrados (grabbable), sliders y otros controles especialmente diseñados para interacción con seguimiento de manos.

- **A-Frame Rounded:**

**Forma:** [unpkg.com/aframe-rounded@1.0.0/aframe-rounded.min.js](https://unpkg.com/aframe-rounded@1.0.0/aframe-rounded.min.js)

Esta pequeña librería permite crear primitivas con bordes redondeados (como a-rounded), muy útiles para diseñar interfaces más suaves y modernas. Se ha utilizado para estilizar menús y botones de forma más estética y amigable para el usuario en la escena.

## 4.2.2. menu.js

El archivo menu.js es uno de los pilares del proyecto, ya que define una componente personalizada de A-Frame cuya responsabilidad principal es leer y procesar la estructura definida en el archivo scene.json y, a partir de ella, construir dinámicamente todos los elementos de la interfaz de usuario: menús, submenús, botones, sliders, etc.

Esta componente no solo crea visualmente los elementos, sino que también define toda la lógica de comportamiento de la escena en función de las interacciones del usuario.

Funcionalidades principales de menu.js:

- **Lectura del archivo `scene.json`:** Interpreta el contenido del JSON, donde están definidos todos los menús, botones, etiquetas, acciones asociadas, jerarquías de submenús y comportamientos interactivos. Este enfoque permite mantener una arquitectura modular, flexible y fácilmente extensible.
- **Creación de la interfaz interactiva:** A partir del JSON, se generan todos los elementos visuales de los menús. Esto incluye:
  - Botones pulsables (para navegar o ejecutar acciones).
  - Menús y submenús organizados jerárquicamente.
  - Opciones dinámicas como sliders o botones de selección.
- **Navegación entre menús y submenús:** Permite moverse hacia adelante y hacia atrás entre menús, con botones de navegación generados automáticamente según la estructura definida en el archivo `scene.json`.
- **Cambio entre modo día y noche:** Incluye una función que altera la iluminación global y el color del entorno, simulando la transición entre ambientes diurnos y nocturnos en la escena virtual.
- **Visualización de datos y modelos 3D:** Incorpora lógica para representar visualmente tanto datos filtrados (mediante diagramas de barras y circulares con la librería BabiaXR), como modelos 3D en formato `.glb`, permitiendo al usuario cargar e inspeccionar objetos tridimensionales dentro del entorno inmersivo.
- **Reproducción de música y sonidos:** Permite asociar funciones de reproducción, pausa o detención de pistas de audio a botones del menú, facilitando así la integración de experiencias sonoras interactivas.
- **Minimizar y maximizar menús:** El menú puede minimizarse mediante un botón, quedando oculto y reemplazado por un ícono anclado a la mano del usuario (modo reloj), desde donde puede volver a expandirse.

- **Control de posición y movimiento de menús:** Los menús pueden mostrarse en ubicaciones dinámicas, moverse en tiempo real y ser posicionados libremente dentro de la escena, lo que contribuye a una experiencia más fluida y adaptativa.
- **Funcionalidad agarrible (grabbable):** Gracias al seguimiento de manos, los usuarios pueden coger físicamente los menús y reposicionarlos dentro del entorno, aumentando así el nivel de personalización e interacción directa con la interfaz.
- **Extensibilidad funcional:** A través de `scene.json`, pueden definirse nuevas funciones o comportamientos a ser ejecutados por los botones, lo que permite escalar fácilmente la aplicación a diferentes contextos o necesidades.

### Funciones utilizadas

A continuación se describen las funciones principales que conforman el componente `menu.js`. Estas funciones permiten construir y controlar dinámicamente todos los elementos interactivos de las escenas a partir de los datos definidos en el archivo `scene.json`.

- **signalCreateMenus()**

Función encargada de iniciar el proceso de construcción de la escena. Llama a `createMenu()` y `createChildMenus()` para generar tanto los menús principales como sus submenús, según la estructura jerárquica definida en el JSON.

- **createMenu(menuData)**

Función que crea un menú principal o un submenú individual. Define su ID, posición, visibilidad y si es o no agarrible. Además, se encarga de llamar a `createButtons()` para generar los botones asociados a ese menú.

- **createChildMenus(buttonsData)**

Función que genera los submenús contenidos dentro de los botones de un menú superior. Utiliza `createMenu()` para construir cada uno de ellos de forma modular.

- **createButtons(menuId, buttonsData)**

Función responsable de crear todos los botones de un menú, asignándoles atributos como ID, posición, visibilidad, tamaño, geometría, etiquetas, etc. Los botones se implementan usando la componente `button.js`.

**■ `getMenuPosition(menuId)`**

Guarda la última posición y rotación de un menú antes de que este sea reemplazado o minimizado. Sirve para mantener la continuidad visual cuando el menú se vuelve a mostrar más adelante.

**■ `applyMenuPosition(menuId)`**

Aplica la posición guardada previamente a un menú que se vuelve a hacer visible (ya sea por un cambio de menú o tras una minimización).

**■ `changeGrabbable(toMenuId, fromMenuId)`**

Desactiva el atributo `grabbable` (`agarrable`) del menú anterior y lo activa en el nuevo menú, permitiendo que solo uno a la vez pueda ser agarrado y movido.

**■ `nextMenu(fromMenuId, toMenuId)`**

Oculta un menú y muestra el siguiente. Es la lógica principal de navegación entre menús y submenús.

**■ `changeBoolean(booleanName)`**

Cambia el estado de un booleano usado para controlar filtros, navegación o visibilidad. Si está en `true`, lo pone en `false`, y viceversa.

**■ `minimizeMenu(menuId)`**

Oculta el menú actual y muestra un botón de maximizar (representado como un reloj) anclado a la mano izquierda del usuario.

**■ `maximizeButton(menuId)`**

Función que vuelve a mostrar el menú que había sido previamente minimizado.

**■ `deleteMenu(menuId)`**

Hace invisible un menú, eliminándolo de la vista, pero no necesariamente del DOM.

**■ `darkMode() / lightMode()`**

Cambian la apariencia general de la escena, simulando un modo noche/día mediante el ajuste de iluminación, color de fondo, y entorno.

- **initializeBoolean(booleanName)**

Pone un booleano específico a false. Se utiliza normalmente al iniciar la escena o al resetear estados.

- **multipleBack()**

Permite volver al menú anterior en contextos donde hay múltiples caminos posibles (por ejemplo, cuando un menú principal tiene varias categorías como "motorz color"). Usa los booleanos de control para decidir a cuál volver exactamente.

- **setupMenuReferences()**

Función clave para inicializar automáticamente referencias a todos los menús definidos en el JSON. Evita tener que hacer manualmente document.querySelector() para cada menú, generando variables con nombres como this.menuIdEl.

- **setupButtonReferences()**

Igual que la anterior, pero aplicada a todos los botones. Crea referencias automáticas como this.buttonIdEl.

- **setupAutomaticButtonEvents()**

Asigna automáticamente los eventos a los botones en función de las acciones especificadas en el JSON, sin necesidad de definir manualmente el comportamiento botón por botón.

**\*Nota:** Aunque en este apartado se han descrito las funciones generales del proyecto, existen otras funciones más específicas relacionadas con cada demo. Estas serán explicadas en su sección correspondiente del documento.

### 4.2.3. **scene.json**

El archivo scene.json constituye una pieza clave dentro de la arquitectura del proyecto, ya que actúa como la fuente de datos estructurados a partir de la cual se genera dinámicamente toda la interfaz de usuario y lógica de interacción en la escena virtual. Este archivo es leído por la componente menu.js, la cual interpreta su contenido y construye a partir de él los menús, botones, submenús y demás elementos interactivos que configuran la experiencia inmersiva.

## Estructura general

El archivo sigue un formato jerárquico en el que se definen uno o varios menús principales (denominados menuP) y, a partir de ellos, se pueden encadenar uno o varios menús hijos (menuCX, donde X es un número identificador). Esta estructura en árbol permite la navegación fluida entre distintas capas de interacción.

A continuación se detallan los elementos que componen la definición de un menú:

- **Menú principal (menuP)**

Un menú principal tiene los siguientes atributos:

- id: Identificador único del menú.
- position: Coordenadas espaciales en las que se mostrará dentro de la escena (x y z).
- visible: Determina si el menú es visible inicialmente (true o false).
- grabbable: Define si el menú puede ser agarrado y movido por el usuario (true o false).
- buttons: Lista de botones que componen el menú. Cada botón puede tener funcionalidades independientes o servir como acceso a un submenú.

- **Botones**

Cada botón definido dentro de un menú contiene múltiples propiedades, entre ellas:

- label: Texto que se mostrará junto al botón. Si no se desea incluir texto, puede utilizarse el valor "noLabel".
- id: Identificador del botón.
- position: Posición del botón dentro del menú.
- posetx, posety, posetx: Posición de la etiqueta en el eje x , y o z, respectivamente.
- width: Ancho de la etiqueta del botón.
- color: Color del botón.
- visible: Visibilidad del botón.
- functionX: Nombre de la función que debe ejecutarse al pulsar el botón (por ejemplo: nextMenu, spawnGLB, playAudio, etc.).

- parameter1fX, parameter2fX, etc.: Parámetros que se pasan a la función correspondiente.

Si el botón permite el acceso a un nuevo menú hijo, este submenú se define directamente dentro de la estructura del botón, bajo la clave menuCX, con una sintaxis anidada que replica la estructura del menuP.

- **Menú hijo (menuC1, menuC2, ...)**

Los menús hijos tienen una estructura similar al menú padre:

- id: Identificador del menú hijo.
- position: Posición espacial.
- visible: Visibilidad inicial (normalmente false hasta que se accede desde el menú anterior).
- grabbable: Si el menú puede ser movido.
- buttons: Lista de botones, con sus respectivas propiedades y funcionalidades.

Esta forma de definición permite construir interfaces altamente dinámicas y escalables, en las que cada botón puede ejecutar múltiples funciones, navegar a otros menús o generar nuevos elementos en la escena, como visualizaciones (.glb), música, sliders, diagramas, etc.

### **Flexibilidad y adaptabilidad**

Una de las ventajas fundamentales de este sistema es que el archivo scene.json no sigue un esquema rígido y puede diseñarse a medida según las necesidades de cada escena. De esta forma, el desarrollador puede definir nuevos botones, añadir funciones personalizadas, incorporar modelos .glb, controlar la iluminación, añadir opciones de audio, o cualquier otra lógica que pueda implementarse desde menu.js.

En definitiva, scene.json actúa como el esqueleto declarativo de la interfaz y su comportamiento, separando completamente los datos de la lógica, y facilitando la extensibilidad del sistema a múltiples entornos y contextos de aplicación.

### 4.3. Guía de usuario

Esta guía detalla el proceso completo para implementar el sistema de menús interactivos en A-Frame desarrollado en este proyecto. Incluye desde la configuración inicial del entorno hasta la personalización de escenas y componentes, cubriendo tanto aspectos técnicos como de integración.

- **1. Descarga de Scripts Necesarios** Antes de comenzar, asegúrate de tener en tu proyecto los siguientes scripts JavaScript. Cada uno de ellos cumple una función específica en la creación, visualización e interacción con los menús.

#### Scripts principales:

- button.js: Define los botones interactivos. Permite configurar:
  - Tamaño del botón
  - Etiqueta (texto mostrado)
  - Estilo de la etiqueta (tamaño, fuente, color, etc.)
  - Componente pressable para que pueda ser pulsado con la mano o el ratón.
- initMenu.js: Se encarga del formato general del menú: su estructura visual, orientación de los botones, distribución en la escena, etc.
- pressable.js: Componente que gestiona la interacción física con los botones, ya sea mediante manos (VR/AR) o con el ratón (en escritorio).

#### Scripts adicionales para sliders:

- pinchable.js: Componente que permite "pinchar" arrastrar el slider con los dedos o punteros.
- slider.js: Componente que define la estructura del slider: barra, punto deslizante y valores.
- size-change.js: Lógica que permite que al mover el slider, un objeto asociado cambie de tamaño dinámicamente.

Asegúrate de incluir todos estos scripts en tu carpeta del proyecto y enlazarlos correctamente en tu index.html.

- **2. Creación de index.html** Este archivo es el punto de entrada a tu escena. Aquí se definen los elementos HTML necesarios y se cargan las librerías y scripts.

#### **Librerías necesarias (deben estar en el head):**

```
<!-- A-Frame principal -->
<script src="https://aframe.io/releases/1.4.2/aframe.min.js"></script>

<!-- Super Hands (para interacción con manos en VR/AR) -->
<script src="https://cdn.jsdelivr.net/npm/aframe-super-hands-component@4.0.5/dist/aframe-super-hands-component.min.js"></script>

<!-- Physics (si usas interacción física) -->
<script src="https://cdn.jsdelivr.net/npm/aframe-physics-system@4.0.1/dist/aframe-physics-system.min.js"></script>
```

#### **Scripts del proyecto (añadir antes del cierre del head):**

```
<script src="js/initMenu.js"></script>
<script src="js/button.js"></script>
<script src="js/pressable.js"></script>
<script src="js/menu.js"></script>
<script src="js/slider.js"></script>
<script src="js/pinchable.js"></script>
<script src="js/size-change.js"></script>
```

#### **Controladores de entrada (manos y ratón):**

```
<a-entity id="leftHand" hand-tracking-controls="hand: left" hand-tracking-grab-controls="hand: left">
<a-entity id="rightHand" hand-tracking-controls="hand: right" hand-tracking-grab-controls="hand: right"></a-entity>
<a-entity cursor="fuse: false; rayOrigin: mouse;" raycaster="objects: .clickable"></a-entity>
```

### Entorno visual (opcional):

- Puedes añadir un entorno personalizado con:
  - Fondos en 360° (imágenes HDR o panorámicas)
  - Formas básicas: `ja-box`, `ja-sphere`, `ja-plane`, `ja-cylinder`, etc.
  - Materiales, luces y decoraciones para simular habitaciones u otras escenas.
- **3. Archivo scene.json: Creación y configuración de menús** Este archivo es el núcleo de la personalización de tu experiencia. Aquí defines qué menús se muestran, qué botones contiene cada uno y qué funcionalidades tienen.

### Estructura básica de scene.json:

```
{
  "menus": [
    {
      "id": "menu-principal",
      "position": [0, 1.5, -2],
      "grabbable": true,
      "buttons": [
        {
          "label": "Mostrar Diagrama",
          "action": "mostrarDiagrama"
        },
        {
          "label": "Cambiar Color",
          "action": "cambiarColorObjeto"
        }
      ]
    },
    {
      "id": "menu-slider",
      "position": [1, 1.5, -2],
      "slider": {
        "min": 0.5,
        "max": 1.0
      }
    }
  ]
}
```

```
        "max": 3,  
        "step": 0.1,  
        "action": "cambiarTamao"  
    }  
}  
]  
}
```

### ¿Qué puedes configurar?

- id: Identificador único del menú.
- position: Posición 3D en el espacio (x, y, z).
- grabbable: Si es true, el menú puede moverse con la mano.
- buttons: Array de botones con etiquetas y acciones asociadas.
- slider: Parámetros del slider (rango, paso, acción que realiza).

Este sistema es flexible y permite múltiples menús en pantalla. Cada menú puede tener botones, sliders o ambos.

- **4. Script menu.js: Controlador de lógica** Este archivo contiene la lógica funcional asociada a los botones y sliders definidos en scene.json.

### ¿Qué define menu.js?

- La carga y lectura del archivo scene.json.
- La creación dinámica de menús según esa configuración.
- La asociación de cada botón a su función específica (ej.: cambiar color, activar sonido, cambiar escena).
- La lógica del comportamiento grabbable para permitir mover los menús si se desea.
- Acciones personalizadas, como:
  - Cambiar el tipo de gráfico mostrado
  - Amueblar una habitación

- Iniciar un sonido relajante o una banda sonora

### Ejemplo de función dentro de menu.js:

```
nextMenu: function (prevM, nextM) {
    this.lastMenuPosition = this.getMenuPosition(prevM);
    this.changeGrabbable(nextM, prevM);
    prevM.setAttribute('visible', false);
    const buttons2 = prevM.querySelectorAll('[id]');
    buttons2.forEach(button => {
        button.setAttribute('visible', false);
    });
    setTimeout(() => {
        this.applyMenuPosition(nextM, this.lastMenuPosition);
        nextM.setAttribute('visible', true);
        const buttons = nextM.querySelectorAll('[id]');
        buttons.forEach(button => {
            button.setAttribute('visible', true);
        });
        if(nextM == this.menuinicio){
            if(this.isDarkMode){
                this.darkButtonEl.setAttribute('visible', false);
            }else{
                this.lightButtonEl.setAttribute('visible', false);
            }
        }
    }, 500);
}
```

## ■ 5. Recomendaciones Finales

- Organización: Mantén una estructura clara de carpetas (js/, assets/, css/, etc.) para que el proyecto sea escalable.
- Modularidad: Puedes crear múltiples scene.json para distintos casos de uso (habitaciones diferentes, escenas temáticas, etc.).

- Compatibilidad: Prueba tu escena en VR, AR (si usas WebXR con dispositivos móviles) y escritorio para asegurar la accesibilidad desde múltiples plataformas.
- Pruebas: Realiza pruebas frecuentes con distintos tamaños de pantalla, resoluciones y controladores (ratón, manos, mandos) para garantizar buena UX.

**Estructura recomendada del proyecto:**

```
index.html  
scene.json  
js/  
    button.js  
    initMenu.js  
    pressable.js  
    menu.js  
    slider.js  
    pinchable.js  
    size-change.js  
assets/ (opcional para texturas, sonidos, fondos, etc.)
```

## 4.4. Caso de uso La habitación de los diagramas

Esta demo tiene como objetivo principal ofrecer una experiencia interactiva de representación visual de datos en un entorno de realidad virtual. El usuario se encuentra en una habitación que sirve como espacio expositivo, donde puede generar distintos tipos de diagramas barras o circulares a partir de un conjunto de datos sobre automóviles almacenado en un archivo JSON.

A través del menú jerárquico, el usuario puede aplicar filtros progresivos sobre la información: seleccionar el tipo de motor (diésel, gasolina o eléctrico), el número de puertas (3 o 5), o el color del coche (blanco, negro, rojo o amarillo). Una vez realizados los filtros, se puede generar un diagrama que refleje únicamente los datos seleccionados. Estos diagramas se representan en 3D y se colocan sobre plataformas dentro de la habitación, junto con una etiqueta que indica su tipo.

Este escenario permite no solo explorar formas innovadoras de presentar información filtrada, sino también experimentar con la interacción natural basada en gestos para navegar y operar visualmente sobre grandes volúmenes de datos.

La primera de las tres escenas desarrolladas en este proyecto ha sido denominada La habitación de los diagramas. Al iniciar esta demo, el usuario se sitúa dentro de una habitación virtual, donde justo frente a él aparece el menú principal interactivo. Este menú es parte del modelo modular desarrollado previamente, por lo que cuenta con todas sus funcionalidades básicas: interacción mediante manos, navegación entre submenús, cambio entre modo día y noche, minimización/maximización, y control de posición mediante agarre.

Alrededor de la habitación se encuentran distribuidos distintos tipos de diagramas 3D que representan datos cargados desde archivos .json. Entre ellos se pueden observar diagramas de barras en 2D y 3D, un diagrama donut, y un diagrama de burbujas, todos generados dinámicamente y posicionados sobre plataformas, con etiquetas que identifican claramente su tipo.

En cuanto al funcionamiento del menú en esta escena, se estructura de la siguiente manera:

- En la parte superior derecha de cada menú o submenú, se encuentra el botón para minimizar. Al pulsarlo, el menú se oculta visualmente y es sustituido por un icono de reloj ubicado en la muñeca de la mano izquierda del usuario, desde el cual se puede maximizar de nuevo en cualquier momento.
- El botón Start del menú principal da paso al primer submenú. Como todos los submenús, este incluye:
  - Botón naranja en la esquina superior izquierda para volver al menú anterior.
  - Botón azul para minimizar el submenú actual.
- Este primer submenú permite al usuario escoger entre visualizar un diagrama de barras o un diagrama circular, mediante los botones correspondientes “Bar” y “Circular”.
- Una vez seleccionada la forma del diagrama, se accede al siguiente submenú, donde se presentan los criterios de filtrado: motor, doors, y color. Al seleccionar uno de estos filtros, se abre otro submenú que presenta las opciones correspondientes:
  - Para motor: Diesel, Electric, Gasoline.

- Para doors: 3 doors, 5 doors.
- Para color: White, Black, Red, Yellow.
- Tras elegir una de estas opciones de filtrado, se accede al último submenú, el cual ofrece:
  - Un botón Delete, que elimina el diagrama mostrado si ya había uno presente.
  - Un botón Show, que genera el nuevo diagrama filtrado con la información elegida.
  - Un slider que permite modificar dinámicamente el tamaño del diagrama mostrado.

De esta forma, el usuario puede construir visualizaciones personalizadas a partir de los datos disponibles, de manera completamente inmersiva, utilizando gestos naturales con las manos para controlar todo el proceso.

A continuación, se muestran algunas capturas representativas del entorno:



Figura 4.2: La habitación de los diagramas con visualizaciones dinámicas activadas.

## 4.5. Caso de uso La habitación a diseñar VR/AR

Esta demo llamada la habitación a diseñar, propone una experiencia creativa en la que el usuario puede amueblar y decorar un entorno de forma personalizada. Al iniciar, se ofrece al usuario la posibilidad de elegir entre dos modalidades distintas: modo VR, en el que se accede a una habitación virtual ya construida que se puede diseñar y organizar a gusto; y modo AR, en el que los objetos virtuales se colocan directamente sobre el espacio físico real visible a través del visor, permitiendo así al usuario decorar su entorno real con mobiliario virtual.

En ambas modalidades, el sistema parte del menú principal, desde el cual se puede alternar entre el modo día y noche, minimizar la interfaz o incluso eliminar el menú por completo. Pul-

sando el botón Start, se accede a un submenú donde el usuario puede elegir entre dos categorías principales: Furniture (mobiliario) y Decoration (decoración).

En la sección de Furniture, se encuentran opciones para colocar objetos como camas (bed), mesas (table), sillas (chair) y mesitas de noche (nightstand). En la sección de Decoration, el usuario puede añadir elementos como lámparas (lamp), fruteros (bowl), cuadros (pictures) y macetas (plant). En ambos casos, tras seleccionar un tipo de objeto, se accede a un submenú final desde el cual es posible crear o eliminar dicho elemento.

Todos los objetos y menús colocados pueden ser agarrados, movidos y posicionados libremente dentro de la escena ya sea la habitación virtual o el entorno físico en AR, brindando una experiencia altamente personalizada e inmersiva. Esta demo explora, por tanto, no solo la generación de contenido 3D interactivo, sino también su integración contextual según el entorno del usuario.

A continuación, se presentan algunas capturas representativas de la Demo 2, en las que se puede observar tanto la disposición del menú como la colocación e interacción con distintos objetos dentro del entorno, ya sea en realidad virtual (VR) o realidad aumentada (AR).



Figura 4.3: La habitación a diseñar en modo AR y VR.

## 4.6. Caso de uso La habitación de los sonidos

La tercera demo, titulada La habitación de los sonidos, ofrece una experiencia centrada en la ambientación sonora dentro de un entorno inmersivo. El objetivo principal de esta escena es permitir al usuario activar distintos tipos de sonidos —ya sean relajantes o bandas sonoras— y acompañarlos con elementos visuales representativos, integrando audio y objetos 3D en una

habitación virtual.

Al iniciar la demo, el usuario se encuentra en una habitación cerrada con el menú principal frente a sí. Este menú, al igual que en las otras escenas, es completamente interactivo: puede agarrarse con las manos, moverse libremente por la escena, minimizarse o cambiar el entorno de día a noche. Al pulsar el botón Start, se accede al primer submenú con dos opciones principales: Sounds y Soundtrack.

- Opción 1: Sounds Esta rama está enfocada en sonidos ambientales y relajantes. Al seleccionarla, se accede a un nuevo submenú con las siguientes opciones:
  - Birds: reproduce sonidos de pájaros y muestra dos aves animadas en la ventana (modelos .glb).
  - Fire: emite un sonido de fuego acompañado por una hoguera virtual (objeto 3D).
  - Water: genera el sonido de agua fluyendo y muestra una fuente.
  - Rain: activa una ambientación de lluvia junto a la aparición de un paraguas virtual.

En cada uno de estos casos, se accede a un submenú final que permite controlar el sonido con los botones Play, Pause y Stop. Además, los elementos 3D aparecen al iniciar la reproducción y desaparecen si se detiene o se retrocede con el botón de volver.

- Opción 2: Soundtrack Esta sección permite al usuario seleccionar y reproducir bandas sonoras de películas icónicas. Al acceder, se presentan botones para las siguientes películas:
  - Gladiator → Aparece un gladiador.
  - El bueno, el feo y el malo → Aparece un revólver.
  - Érase una vez en América → Aparece un maletín.
  - Malditos bastardos → Aparece un bate.

En cada caso, el usuario accede a un submenú con los controles Play, Pause y Stop. Al pulsar Play, se reproduce la banda sonora correspondiente y aparece el objeto 3D representativo. Al retroceder o detener el audio, el objeto también desaparece, asegurando una coherencia visual-sonora dentro de la experiencia.

Esta demo demuestra cómo el modelo de menú creado puede extenderse a experiencias sensoriales más completas, combinando sonido, interacción natural y elementos visuales que refuerzan la ambientación deseada.

A continuación, se presentan algunas capturas que muestran el funcionamiento y la ambientación de la Demo 3:



Figura 4.4: Demo 3 - La habitación de los sonidos.

# Capítulo 5

## Conclusiones

### 5.1. Consecución de objetivos

En esta sección se analiza hasta qué punto se han alcanzado los objetivos generales, específicos y complementarios planteados al inicio del proyecto.

El objetivo general de diseñar y desarrollar una aplicación interactiva en realidad virtual y aumentada, que permita al usuario explorar contenido y generar visualizaciones de datos mediante menús manipulables con las manos, ha sido plenamente alcanzado. El resultado final permite, a través de la biblioteca BabiaXR y el framework A-Frame, crear experiencias inmersivas tanto en VR como en AR, sin necesidad de mandos físicos y con total compatibilidad con las Meta Quest 3.

A nivel de objetivos específicos, se ha conseguido:

- **Investigar tecnologías adecuadas** para experiencias inmersivas en WebXR, utilizando A-Frame, BabiaXR y seguimiento de manos con éxito. Se ha seleccionado y utilizado un stack tecnológico apropiado que ha permitido cubrir tanto los aspectos de interacción como los gráficos.
- **Desarrollar escenas inmersivas** con distintos niveles de complejidad. Las tres demos implementadas permiten desde interacciones sencillas como reproducir sonidos o crear muebles, hasta generar y filtrar visualizaciones de datos mediante diagramas interactivos.
- **Diseñar e implementar menús interactivos tridimensionales** manipulables con las manos. Estos menús permiten navegar intuitivamente entre distintas opciones, activar fun-

cionalidades y controlar elementos de la escena mediante gestos naturales.

- **Garantizar la compatibilidad con Meta Quest 3 y navegadores compatibles con WebXR.**  
La aplicación ha sido probada y ajustada específicamente para estos dispositivos, asegurando un rendimiento fluido en modos tanto de realidad virtual como aumentada.
- **Integrar BabiaXR para la visualización de datos en tiempo real,** permitiendo filtrar datos, representarlos en distintos formatos y manipular sus propiedades (tamaño, posición, color) desde la interfaz de usuario.

Además, se han cumplido los objetivos complementarios, como:

- La creación e integración de componentes personalizados en A-Frame para modularizar el proyecto, reutilizar funcionalidad y mantener una estructura de código clara y escalable.
- La familiarización profunda con BabiaXR, tanto a nivel de uso como de extensión para adaptarlo a los requerimientos específicos del proyecto.
- El mejor diseño visual y de usabilidad, cuidando aspectos estéticos de los menús, botones y la disposición espacial de los elementos en escena.
- El uso fluido de tecnologías web (HTML, JavaScript y CSS) dentro de un entorno 3D, combinando lógica de interacción con diseño visual.
- La experimentación práctica con Meta Quest 3, comprendiendo sus capacidades de seguimiento de manos, diferencias entre VR y AR, y peculiaridades del desarrollo en este entorno.
- Y por último, se ha logrado consolidar los conocimientos adquiridos durante el grado, integrando habilidades técnicas, diseño de interfaces, lógica de interacción y nuevas tecnologías inmersivas en un proyecto cohesivo y funcional.

## 5.2. Aplicación de lo aprendido

Durante la realización de este Trabajo Fin de Grado he puesto en práctica una gran parte de los conocimientos adquiridos a lo largo del grado, especialmente aquellos relacionados con el desarrollo web y la programación.

La asignatura más directamente relacionada con el proyecto ha sido Aplicaciones Telemáticas, ya que en ella se trabajaron en profundidad tecnologías como HTML, CSS y JavaScript, fundamentales para el uso de A-Frame y la construcción de escenas interactivas en entornos tridimensionales. Gracias a esta asignatura, adquirí una base sólida en desarrollo frontend que ha sido clave para estructurar correctamente las interfaces inmersivas, los menús y la interacción por gestos que forman parte del proyecto.

También fue útil la asignatura Servicios Telemáticos, que me aportó una visión más global del desarrollo web, al integrar tanto la parte de cliente como de servidor. Esta perspectiva me ayudó a comprender cómo organizar y comunicar los distintos elementos del sistema, especialmente a nivel de arquitectura y flujo de datos.

Además, en asignaturas como Sistemas Operativos (con programación en C), Ampliación de Sistemas Telemáticos (Java) y Sistemas Distribuidos (Go), adquirí experiencia en distintos lenguajes y paradigmas de programación que me han permitido mejorar mis habilidades de estructuración, lógica y resolución de problemas, aplicadas también en este proyecto para optimizar el rendimiento, modularizar el código y garantizar la escalabilidad de las escenas desarrolladas.

En resumen, este TFG ha sido una oportunidad excelente para integrar todos estos conocimientos en un proyecto práctico, innovador y alineado con las tecnologías emergentes del desarrollo inmersivo en realidad virtual y aumentada.

### 5.3. Lecciones aprendidas

El desarrollo de este Trabajo Fin de Grado me ha permitido adquirir numerosos conocimientos técnicos y metodológicos que no solo complementan mi formación académica, sino que también amplían mis competencias en áreas emergentes como la realidad virtual y aumentada.

En primer lugar, he aprendido a trabajar en profundidad con A-Frame, entendiendo su estructura, sus componentes y su potencial para construir experiencias inmersivas en el navegador. Esto me ha llevado a profundizar también en tecnologías asociadas como WebXR, Three.js y especialmente BabiaXR, que ha sido una parte central del proyecto. El uso de BabiaXR me ha obligado a comprender cómo se generan, manipulan e integran visualizaciones de datos dentro

de entornos tridimensionales, lo cual ha supuesto un gran reto a nivel técnico.

Uno de los aprendizajes más valiosos ha sido el manejo del seguimiento de manos como método de interacción. Implementar menús completamente manipulables mediante gestos ha requerido un gran trabajo de prueba, ajuste y comprensión de cómo funcionan los eventos en WebXR y cómo se detectan y procesan las colisiones y gestos de la mano en entornos inmersivos.

Además, el proyecto me ha permitido afianzar mis conocimientos en JavaScript, HTML y CSS, elevando mi nivel en el desarrollo frontend, especialmente aplicado a entornos 3D. También he aprendido a estructurar y modularizar código reutilizable en A-Frame, lo que ha mejorado mi capacidad para mantener y escalar escenas complejas.

Desde el punto de vista práctico, he aprendido a desarrollar, probar y optimizar una aplicación pensada para dispositivos como las Meta Quest 3, teniendo en cuenta tanto las limitaciones técnicas como la experiencia de usuario. Esto ha incluido desde configurar el entorno de desarrollo adecuado hasta hacer pruebas en condiciones reales de uso.

Por último, también destaco la experiencia adquirida en la elaboración de documentación técnica con LaTeX, así como en la organización del trabajo mediante una planificación flexible y progresiva. Todo ello me ha ayudado a afrontar de forma autónoma un proyecto multidisciplinar, integrando conocimientos adquiridos durante el grado en un contexto práctico y aplicado.

## 5.4. Trabajos futuros

Aunque el desarrollo de esta aplicación ha alcanzado los objetivos planteados, existen múltiples vías de mejora y expansión que podrían abordarse en futuros Trabajos Fin de Grado o Máster.

Una posible línea de continuación sería ampliar la funcionalidad de los menús interactivos, incluyendo opciones más complejas como menús contextuales, herramientas de edición de escenas o integración con comandos de voz. Esto permitiría al usuario un mayor grado de personalización y control dentro del entorno inmersivo, aumentando la usabilidad de la aplicación.

También sería interesante mejorar la integración con fuentes de datos externas en tiempo real, como APIs o bases de datos dinámicas. Esto abriría la puerta a la creación de visualizaciones de datos actualizadas en vivo, ampliando el potencial de la herramienta para aplicaciones

educativas, científicas o empresariales.

En cuanto a la interacción, un avance importante sería refinar el sistema de reconocimiento de gestos y explorar su combinación con otras formas de entrada, como el seguimiento ocular o comandos por voz, para lograr una experiencia aún más natural e intuitiva.

Una funcionalidad muy útil que podría implementarse en futuras versiones es permitir el desplazamiento dentro de la escena mediante gestos con las manos, así como cambiar la vista o el punto de referencia del usuario sin necesidad de utilizar controladores físicos. Esto incrementaría la inmersión y daría más libertad al usuario para explorar entornos complejos.

Otra dirección prometedora sería adaptar la aplicación a un entorno colaborativo multiusuario, permitiendo que varios usuarios interactúen simultáneamente en la misma escena desde diferentes dispositivos. Esto implicaría integrar tecnologías de sincronización en red y avatares personalizados, abriendo el camino hacia experiencias compartidas en tiempo real.

Además, podría explorarse la mejora del rendimiento y la accesibilidad, optimizando la carga de escenas para dispositivos menos potentes o explorando mecanismos que permitan su uso desde el navegador móvil sin gafas VR, haciendo la experiencia más inclusiva.

Por último, sería beneficioso crear una interfaz de configuración para desarrolladores, que permita construir visualizaciones personalizadas desde una interfaz gráfica sin necesidad de modificar el código, acercando esta herramienta a usuarios sin conocimientos técnicos.



# Apéndice A

## **Enlaces de interés:**

A continuación, se presentan los enlaces más relevantes asociados al desarrollo de este Trabajo de Fin de Grado. Estos recursos permiten acceder al código fuente, a la versión desplegada en la web y a la publicación del componente en el gestor de paquetes npm.

- Repositorio del proyecto en GitHub:

<https://github.com/mcobom2019/TFG>

Aquí se encuentra disponible el código fuente completo del proyecto, incluyendo scripts, escenas, documentación y recursos utilizados durante el desarrollo.

- Página web del TFG (versión desplegada):

<https://mcobom2019.github.io/TFG/>

Este sitio web ofrece una versión funcional del proyecto que puede visualizarse desde un navegador compatible con WebXR. Está optimizada para su uso tanto en ordenadores como en dispositivos de realidad virtual y aumentada.

- Componente publicado en npm:

<https://www.npmjs.com/package/mcobom2019-menu>

Con el objetivo de fomentar la reutilización del trabajo realizado y facilitar que otros desarrolladores puedan integrarlo en sus propios proyectos, se ha subido el componente principal del sistema de menús interactivos a npm. Esta publicación permite la instalación mediante gestores de paquetes, e incluye instrucciones básicas de uso y ejemplos de integración.



# **Bibliografía**