



**Universidad  
Rey Juan Carlos**

**GRADO EN INGENIERÍA EN TELEMÁTICA**

Curso Académico 2021/2022

Trabajo Fin de Grado

**Aplicación de Web Progresiva (PWA) y optimización  
de la asistencia a seminarios de la URJC**

Autor : David González Endrinal

Tutor : Dr. Gregorio Robles Martínez



# **Trabajo Fin de Grado**

Aplicación de Web Progresiva (PWA) y optimización de la asistencia a  
seminarios de la URJC

**Autor :** David González Endrinal

**Tutor :** Dr. Gregorio Robles Martínez

La defensa del presente Trabajo Fin de Grado se realizó el día                de  
de 2021, siendo calificada por el siguiente tribunal:

**Presidente:**

**Secretario:**

**Vocal:**

y habiendo obtenido la siguiente calificación:

**Calificación:**

Fuenlabrada, a                de                de 2021



*Dedicado a  
mis padres, mis hermanas, mi abuelo y mis sobrinos*



# Agradecimientos

En primer lugar quiero dar gracias a mi familia. En especial a mis padres quienes han estado conmigo desde el primer minuto, con un apoyo incondicional en todo momento, dándome ánimos en momentos en los que yo no los encontraba y los que me han dado la oportunidad de formarme durante toda mi vida. A mis hermanas, quienes se han alegrado tanto como yo en los momentos en los que aprobaba, mostraban ilusión por mis logros y las que me han aconsejado siempre buscando lo mejor para mí. A mi abuelo, que siempre ha tenido palabras de ánimo en todo momento que ha durado esta andadura universitaria.

No hay palabras suficientes de agradecimiento por el apoyo que siempre he sentido y por demostrarme que podía con ello, este título es, en gran medida, gracias a vosotros.

También quiero dar las gracias a mi tutor y profesor, Gregorio Robles, por darme la oportunidad de realizar este proyecto, de su ayuda y el interés mostrado durante la realización del mismo. También a todos los demás profesores que durante todos estos años han contribuido en mi formación.

Por último, a todos los amigos que durante estos años universitarios han compartido conmigo sus días, los buenos y los no tan buenos, con los que he podido forjar una gran amistad.

¡Gracias!



# Resumen

Este proyecto consiste en el desarrollo de una Aplicación de Web Progresiva (PWA en inglés, *Progressive Web Applications*) que facilite los docentes el proceso de ofrecer a los alumnos los cursos/seminarios y jornadas tecnológicas que se imparten o tienen lugar en la Universidad durante el curso académico.

Esta aplicación y todo su desarrollo se enmarca dentro del modelo cliente-servidor. Se trata de un modelo de diseño de software en el que las tareas se reparten entre los proveedores (servidores) y los demandantes (clientes). En este caso, un cliente, nuestra PWA desarrollada en este proyecto, realiza peticiones al servidor que será el encargado de toda la administración y gestión de los contenidos que contendrá dicha aplicación. Según los tipos de arquitectura de cliente-servidor este proyecto se enmarcaría dentro de la arquitectura de dos capas, esta arquitectura se utiliza para describir los sistemas cliente servidor donde el cliente solicita recursos y el servidor responde directamente a la solicitud con sus propios recursos, esto significa que el servidor no requiere de una aplicación extra para proporcionar parte del servicio.

Este tipo de modelo presenta una serie de ventajas y desventajas, en cuanto a las ventajas:

- Administración centrada en el servidor. Los clientes tienen poca trascendencia en el esquema y sus necesidades de administración son menores.
- Centralización de los recursos. Los recursos comunes a todos los usuarios se administran en el servidor. Así se evitan situaciones como la redundancia o inconsistencia de información en las bases de datos.
- Mejora de la seguridad. Al disponer de un mecanismo central de autenticación, las posibilidades de acceso indebido se reducen considerablemente.
- Escalabilidad de la instalación. Se pueden añadir o suprimir clientes sin que el funciona-

miento de la red se vea afectado.

Y por otro lado los siguientes inconvenientes:

- Coste elevado. Tanto la instalación como el mantenimiento son más elevados debido al perfil muy técnico del lado servidor.
- Dependencia del servidor. Toda la red está construida alrededor del servidor y si éste deja de funcionar o lo hace con un rendimiento inadecuado, afectará a toda la infraestructura.

Para el desarrollo de la parte servidor se han utilizado diferentes entornos y lenguajes, destacando Python, Node.js y MongoDB para la gestión y almacenamiento en bases de datos.

Con la creación de esta aplicación se intenta, por el lado docente, buscar una eficiencia a la hora de repartir y adjudicar las distintas aulas de las que se dispone para la realización de los distintos cursos y seminarios, por otro lado, por el lado de los alumnos, se busca tener un entorno de fácil manejo y simpleza a la hora de informarse, apuntarse y obtener los créditos correspondientes de dichos cursos y seminarios.

# Summary

This project consists of the development of a Progressive Web Application (PWA in English, *emph* Progressive Web Applications) that facilitates teachers the process of offering students courses / seminars and technological conferences that are taught or take place in the University during the academic year.

This application and all its development is framed within the client-server model, it is a software design model in which the tasks are shared between the providers (servers) and the applicants (clients). In this case, a client, our PWA developed in this project, makes requests to the server that will be in charge of all the administration and management of the content that said application will contain. According to the types of client-server architecture, this project would be framed within a two-tier architecture, this architecture is used to describe client-server systems where the client requests resources and the server responds directly to the request with its own resources, this means that the server does not require an extra application to provide part of the service.

This type of model has a number of advantages and disadvantages, in terms of advantages:

- Server-centric administration. Clients are of little consequence in the scheme and their management needs are less.
- Centralization of resources. The resources common to all users are managed on the server. In this way, situations such as redundancy or inconsistency of information in the databases are avoided.
- Improved security. By having a central authentication mechanism, the chances of improper access are greatly reduced.
- Scalability of the installation. Clients can be added or deleted without affecting the ope-

ration of the network.

And on the other hand the following drawbacks:

- High cost. Both installation and maintenance are higher due to the highly technical profile on the server side.
- Server dependency. The entire network is built around the server and if it stops working or does so with inadequate performance, it will affect the entire infrastructure.

For the development of the server part, different environments and languages have been used, highlighting Python, Node and MongoDB for management and storage in databases.

With the creation of this application, an attempt is made, on the teaching side, to seek efficiency when it comes to distributing and allocating the different classrooms that are available for the different courses and seminars, on the other hand, on the side of The students seek to have an environment of easy handling and simplicity when it comes to information, signing up and obtaining the corresponding credits of said courses and seminars.

# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Contexto . . . . .	1
<b>2. Objetivos</b>	<b>3</b>
2.1. Objetivo general . . . . .	3
2.2. Objetivos específicos . . . . .	3
2.3. Planificación temporal . . . . .	4
<b>3. Estado del arte</b>	<b>7</b>
3.1. Angular . . . . .	7
3.1.1. @zxing/ngx-scanner v.3.0.0 . . . . .	8
3.1.2. @angular/pwa v.0.1101.1 . . . . .	8
3.2. Typescript . . . . .	9
3.3. HTML . . . . .	9
3.4. CSS . . . . .	10
3.5. Bootstrap . . . . .	10
3.6. Node.js . . . . .	10
3.7. MongoDB . . . . .	11
3.8. Python . . . . .	12
<b>4. Diseño e implementación</b>	<b>15</b>
4.1. Arquitectura general . . . . .	15
4.1.1. Cliente . . . . .	15
4.1.2. Servidor . . . . .	17
4.2. Modelo de datos . . . . .	19

4.2.1. MongoDB . . . . .	19
4.2.2. Registros de Alumnos y Eventos . . . . .	21
4.2.3. Plantilla documento de obtención créditos . . . . .	22
4.3. Diseño e implementación por funcionalidad . . . . .	24
4.3.1. Cliente . . . . .	24
4.3.2. Servidor . . . . .	34
<b>5. Pruebas y funcionamiento</b>	<b>39</b>
5.1. Server Workers . . . . .	39
5.2. Instalación en un dispositivo móvil . . . . .	39
5.3. Optimización . . . . .	40
5.4. Seguridad . . . . .	41
<b>6. Resultados</b>	<b>43</b>
<b>7. Conclusiones</b>	<b>45</b>
7.1. Aplicación de lo aprendido . . . . .	45
7.2. Lecciones aprendidas . . . . .	46
7.3. Trabajos futuros . . . . .	47
<b>A. Manual de usuario</b>	<b>49</b>
<b>Bibliografía</b>	<b>51</b>

# Índice de figuras

2.1. Diagrama de GANTT del desarrollo del TFG. . . . .	5
4.1. Arquitectura general. . . . .	16
4.2. Modelo Eventos. . . . .	19
4.3. Modelo Usuarios. . . . .	20
4.4. Modelo Salas. . . . .	20
4.5. Modelo Salas. . . . .	21
4.6. Modelo del archivo CSV. . . . .	21
4.7. Plantilla PDF créditos. . . . .	23
4.8. Página Login . . . . .	24
4.9. Página Login . . . . .	25
4.10. Menú principal de la aplicación . . . . .	26
4.11. Página de inicio. . . . .	26
4.12. Pestaña de validación de QR . . . . .	27
4.13. Pestaña de validación de QR con mensaje de error . . . . .	28
4.14. Página de calendario . . . . .	28
4.15. Página de descripción del evento y botón de apuntarse . . . . .	29
4.16. Página para obtener el certificado de créditos . . . . .	29
4.17. Pop-up con la vista del certificado . . . . .	30
4.18. Vista de administrador de la pestaña Lista de Eventos . . . . .	31
4.19. Contenido de Eventos.zip descargado al presionar <i>Descargar PDF Todos</i> . . . . .	32
4.20. Vista de administrador de la ventana informativa del evento de Telefónica . . . . .	32
4.21. Pestaña Calendario Eventos . . . . .	33
4.22. Pestaña Estadística . . . . .	34

4.23. Esquema funcionamiento createCSV.py . . . . .	36
5.1. Service Workers funcionando en la PWA . . . . .	40
5.2. PWA en la pantalla de inicio del dispositivo móvil . . . . .	40
5.3. Salida tras ejecutar Python optimizarSalas.py . . . . .	41

# **Capítulo 1**

## **Introducción**

En el presente trabajo consta la realización de una aplicación web progresiva (PWA) que ayudará y facilitara a los docentes el proceso de ofrecer a los alumnos los cursos/seminarios y jornadas tecnológicas que se impartirán en la Universidad durante el curso.

### **1.1. Contexto**

La herramienta propuesta en este TFG nace de la necesidad por parte de la Universidad Rey Juan Carlos, y más concretamente del administrador/profesor encargado de gestionar todo lo relacionado con las charlas y seminarios que en ella se ofrecen.

Está principalmente orientada hacia los alumnos, que son en definitiva las personas que más uso harán de la misma. En cuanto a la parte web, supondrá una herramienta de fácil manejo donde ver estos horarios, apuntarse a ellos y obtener los créditos correspondientes por acudir a cada uno de ellos.

Por parte de la Universidad, y más concretamente del administrador o encargado de las charlas, agilizará y le quitará gran carga de trabajo, facilitando la gestión y adjudicación de los certificados que se entregarán a los usuarios finales de las mismas.

La motivación para realizar este proyecto es introducir, más si cabe, las nuevas tecnologías y la agilidad que éstas nos proporcionan dentro del entorno universitario, en su parte backend, destaca la optimización de la gestión de las salas disponibles en la universidad en función de los alumnos apuntados a cada uno de estos seminarios.

Es un proyecto muy centrado en la automatización de procesos, guardando la seguridad que

se requiere en cuanto asuntos de gestión universitaria se refiere, facilitando y agilizando las gestiones tanto a alumnos como a la propia universidad.

# **Capítulo 2**

## **Objetivos**

### **2.1. Objetivo general**

La realización de este trabajo de fin de grado consiste en crear una Aplicación Web Progresiva o PWA que proporcionara a alumnos y a docentes una herramienta de fácil manejo con la que estar informado y gestionar los distintos cursos/seminarios y jornadas tecnológicas que se impartirán en la Universidad durante el curso.

### **2.2. Objetivos específicos**

- Automatizar el reparto de las aulas durante los días y las horas disponibles.
- Informar al docente o administrador de la aplicación del número de alumnos apuntados a los distintos cursos/seminarios y jornadas.
- Informar a los alumnos del horario de los distintos cursos/seminarios y jornadas tecnológicas.
- Gestionar (apuntando y borrando) a los alumnos interesados en los seminarios disponibles.

## 2.3. Planificación temporal

Antes incluso de saber la temática del TFG o sobre que iba a tratar, tenía claro las tecnologías que quería usar. En esos momentos estaba adentrándome en el mundo front con Angular y el desarrollo de aplicaciones Web.

La primera toma de contacto con Gregorio fue a través del correo de la URJC, en el que por medio del mismo comenté mi preferencia a hacer el TFG con él, allá por febrero del 2020. En la primera reunión Gregorio me propuso varios temas sobre los que podía enfocar mi TFG, todos sobre desarrollo Web pero con un requisito claro, debería funcionar y estar optimizados para utilizarlos en dispositivos móviles.

Después de esa reunión pase los siguientes días investigando sobre que tipo de aplicación debería tratar, y fue ahí cuando decidí hacer una aplicación de web progresiva (PWA), habría leído sobre ellas, las conocía un poco y vi que era una gran oportunidad de aprender y desarrollar algo nuevo para mí. En la siguiente reunión del mismo mes de febrero presenté esta tecnología a Gregorio y fue cuando empezó el desarrollo de la misma.

Entre todo el 2020 y alternando con mi trabajo a jornada completa fui desarrollando el proyecto, teniendo durante este tiempo tres reuniones vía Skype con Gregorio para mostrar los avances de la aplicación, preguntar algunas dudas y recibir consejos de por donde podría enfocar el desarrollo.

En enero de 2021 tuve la última reunión para hablar del desarrollo de la aplicación, ya estaba terminada con la integración de algunos puntos y funciones que me pedía Gregorio y fue en ese momento cuando recibí la validación por su parte y pasé a redactar la memoria.

Durante la redacción de esta memoria he recibido numerosos consejos de como debía redactarla, posibles modificaciones o puntos necesarios que no podrían faltar en la memoria para que la aplicación quedara correctamente descrita.

Por último con la memoria terminada, repasando y buscando nuevas optimizaciones de código fui añadiendo funcionalidades nuevas a la aplicación que entendía que podían ser útiles para el administrador de la misma.

Cuando estaban terminadas las dos partes principales de este TFG, la aplicación propiamente programada y la presente memoria, fue cuando decidí presentar este TFG para su defensa.

Éste TFG no habría sido posible sin la ayuda y la gran participación que ha tenido Gregorio

## 2.3. PLANIFICACIÓN TEMPORAL

5

a lo largo del desarrollo del mismo, a quién le estoy tremadamente agradecido.

A continuación, muestro de forma gráfica el diagrama temporal destacando los puntos que anteriormente he desarrollado a lo largo de los meses con respecto a la realización de este TFG.



Figura 2.1: Diagrama de GANTT del desarrollo del TFG.



# Capítulo 3

## Estado del arte

A continuación, se detallan las tecnologías utilizadas en el desarrollo del proyecto.

### 3.1. Angular

Angular [3] es un framework para aplicaciones web desarrollado en TypeScript (lenguaje del que hablaremos más adelante), de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.

Entre sus virtudes se destacan:

- La posibilidad de utilizar templates declarativos, aplicar inyecciones de dependencias y crear componentes reutilizables.
- Es modular, por lo cual se basa en un core y en módulos que permiten acceder a más características cuando es necesario. Permite crear componentes, razón por la cual se destaca en la reutilización de los elementos.
- Ejecuta la primera vista de tu aplicación en Node.js, .NET, PHP, y otros servidores para un renderizado de forma casi instantánea obteniendo solo HTML y CSS. También abre posibilidades para la optimización del SEO (Search Engine Optimization) del sitio, incluyendo configuración.

- Son aplicaciones que cargan rápidamente gracias al nuevo enrutador de componentes. Este ofrece una división automática de códigos para que los usuarios solo carguen el código necesario para procesar la vista que soliciten.
- Por último, dentro de las cosas que debemos tener en cuenta de Angular, es que nuestros proyectos deben ser compilados para traducirlos en una aplicación Web que podamos publicar. Este compilador ha sido mejorado a partir de Angular 5, permitiendo ahora compilación incremental y logrando como resultado que los re-build sean más rápidos.

A continuación destacamos dos paquetes npm [16] que son importantes y sobre los que gira gran parte de este proyecto de fin de grado:

### **3.1.1. @zxing/ngx-scanner v.3.0.0**

Paquete con el que mediante una única línea de código permite a nuestra aplicación angular (PWA) activar la cámara de nuestro dispositivo, pudiendo leer códigos QR o Códigos de barras simples obteniendo información de los mismos. En este proyecto daremos uso a este paquete a la hora de validar nuestra presencia en cualquiera de las charlas o seminarios en los que estemos inscritos y así poder certificar que hemos asistido a ellos para más tarde poder pedir el certificado con los créditos correspondientes. [13]

### **3.1.2. @angular/pwa v.0.1101.1**

Es el paquete principal que usaremos para convertir nuestra aplicación Angular en una Aplicación de Web Progresiva (PWA) [2], mediante el comando *ng add @angular/pwa* se añadirán una serie de archivos y se modificarán otros existentes para poder realizar este cambio, dichas modificaciones son:

1. Se agregará el paquete *@angular/service-worker* al proyecto.
2. Habilitará la compatibilidad con la compilación del service worker en CLI.
3. Importará y registrará al service worker en el módulo de la aplicación.
4. Actualizará el archivo *index.html* incluyendo un enlace para agregar el archivo *manifest.webmanifest*.

5. Instalará archivos de iconos para admitir la aplicación web progresiva (PWA) instalada.
6. Creará el archivo de configuración del service worker *llamado ngsw-config.json*, que especifica los comportamientos de almacenamiento en caché y otras configuraciones.

## 3.2. Typescript

TypeScript [19] es un superconjunto de JavaScript. Una tecnología es un superconjunto de un lenguaje de programación cuando puede ejecutar programas de la tecnología, TypeScript en este caso, y del lenguaje del que es el superconjunto, JavaScript en este mismo ejemplo.

Esto permite que se pueda integrar TypeScript en proyectos existentes de JavaScript sin tener que reimplementar todo el código del proyecto en TypeScript. De hecho, es común que existan proyectos que introduzcan tanto TypeScript como JavaScript.

La principal característica de TypeScript es el tipado estático. Decimos que un lenguaje es de tipado estático cuando no es necesario definir las variables antes de su uso. Esto implica que la tipificación estática tiene que ver con la declaración explícita (o inicialización) de las variables antes de que se empleen.

## 3.3. HTML

HTML [22] son las siglas de “HyperText Markup Language”. Se trata del lenguaje estándar utilizado para la creación de páginas web. Fue creado por Tim Berners-Lee en los años ochenta. Es un lenguaje de marcas, esto quiere decir que para indicar la estructura del documento se utilizan etiquetas. Al ser un estándar, todos los navegadores leen e interpretan el lenguaje HTML y lo presentan de un modo similar.

### 3.4. CSS

CSS [1] son las siglas de “Cascading Style Sheets”. Es un lenguaje que permite establecer la apariencia de un documento escrito mediante un lenguaje de marcas. CSS permite asociar reglas a los elementos que aparecen en una página web, las reglas indican como debe representarse el contenido de esos elementos.

### 3.5. Bootstrap

Bootstrap [6] es un framework front-end<sup>1</sup> que permite la construcción de páginas web adaptativas. Se trata de un software de código abierto creado en el año 2011 por Mark Otto y Jacob Thornton, empleados de Twitter.

Bootstrap ha evolucionado desde ser un proyecto enteramente basado en CSS hasta incluir múltiples plugins JavaScript e iconos junto con formularios y botones. Presenta una cuadrícula de doce columnas y 940px de ancho.

La creación de páginas web mediante Bootstrap se simplifica al disponer de elementos predefinidos fácilmente incluibles en cualquier página web, tales como menús desplegables, botones, iconos, ventanas emergentes, etc.

### 3.6. Node.js

Node.js [14] es un entorno JavaScript que nos permite ejecutar en el servidor, de manera asíncrona, con una arquitectura orientada a eventos y basado en el motor V8 de Google.

El motor V8 de Google compila JavaScript en código máquina nativo en vez de interpretarlo en el navegador, consiguiendo así una velocidad mucho más alta. Además de la alta velocidad de ejecución, Node.js dispone del Bucle de Eventos (Event Loop), que permitirá gestionar enormes cantidades de clientes de forma asíncrona. Tradicionalmente para trabajar de forma asíncrona, las aplicaciones se valían de la programación basada en hilos (programming threaded applica-

---

<sup>1</sup>Un framework front-end es una herramienta que se integra con nuestro proyecto web para conseguir que el desarrollo front-end (interfaz, animaciones, entre otros) sea más fácil, rápido y robusto; convirtiéndolo en una herramienta muy útil para desarrolladores principiantes y programadores con poco tiempo y poca experiencia en ámbitos de diseño

tions), pero esto supone la utilización (normalmente ineficaz) de un espacio de memoria que va escalando a medida que la cantidad de clientes conectados a nuestra aplicación aumenta.

Node.js resuelve este problema cambiando la manera de realizar las conexiones con el servidor. En vez de generar un nuevo hilo E/S para cada cliente, cada conexión dispara la ejecución de un evento dentro del proceso del motor de Node. De este modo, Node.js permite que un solo servidor que lo ejecute pueda soportar decenas de miles de conexiones. Por lo tanto, si necesitamos gestionar grandes cantidades de conexiones no tendremos que ampliar el número de servidores.

En conclusión: Es una plataforma que agiliza y facilita las conexiones cliente servidor por lo que es una tecnología que avanza muy rápidamente y cada vez está más presente en el mercado.

## 3.7. MongoDB

Es una de las bases de datos NoSQL y orientada a documentos que existen. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON. Una de las diferencias más importantes con respecto a las bases de datos relacionales es que no es necesario seguir un esquema; los documentos de una misma colección –concepto similar a una tabla de una base de datos relacional– pueden tener esquemas diferentes.

Aunque se suele decir que las bases de datos NoSQL tienen un ámbito de aplicación reducido, MongoDB [11] se puede utilizar en muchos de los proyectos que desarrollamos en la actualidad:

- No existen las transacciones: Aunque nuestra aplicación puede utilizar alguna técnica para simular las transacciones, MongoDB no tiene esta capacidad. Solo garantiza operaciones atómicas a nivel de documento.
- No existen los JOINS: MongoDB está destinado a proyectos en los que los datos no tengan que ser estructurados en tablas y que tampoco tengamos que hacer relaciones entre ellas.

### 3.8. Python

Python [18] es un lenguaje de scripting<sup>2</sup> desarrollado a principios de los noventa por Guido van Rossum, independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa: desde aplicaciones Windows a servidores de red o incluso, páginas web. Entre sus muchas características destacamos:

- Multiplataforma: Hay versiones disponibles de Python en muchos sistemas informáticos distintos. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para él.
- Interpretado: Quiere decir que no se debe compilar el código antes de su ejecución. En ciertos casos, cuando se ejecuta por primera vez un código, se producen unos bytecodes<sup>3</sup> que se guardan en el sistema y que sirven para acelerar la compilación implícita que realiza el intérprete cada vez que se ejecuta el mismo código.
- Interactivo: Python dispone de un intérprete por línea de comandos en el que se pueden introducir sentencias. Cada sentencia se ejecuta y produce un resultado visible, que puede ayudarnos a entender mejor el lenguaje y probar los resultados de la ejecución de porciones de código rápidamente.
- Orientado a Objetos: La programación orientada a objetos está soportada en Python y ofrece en muchos casos una manera sencilla de crear programas con componentes reutilizables.
- Gran cantidad de funciones y bibliotecas: Dispone de muchas funciones incorporadas en el propio lenguaje para el tratamiento de strings, números, archivos, etc. Además, existen muchas bibliotecas que podemos importar en los programas para tratar temas específicos

---

<sup>2</sup>Los lenguajes de scripting son un tipo específico de lenguajes informáticos que se pueden utilizar para dar instrucciones a otro software, como un navegador web, un servidor o una aplicación independiente. Muchos de los lenguajes de scripts más populares de hoy en día son lenguajes de programación, como JavaScript, PHP, Ruby, Python, y varios otros.

<sup>3</sup>El bytecode es un código intermedio más abstracto que el código máquina. Habitualmente es tratado como un archivo binario que contiene un programa ejecutable similar a un módulo objeto, que es un archivo binario producido por el compilador cuyo contenido es el código objeto o código máquina .

como la programación de ventanas o sistemas en red o cosas tan interesantes como crear archivos comprimidos en .zip.

- Sintaxis clara: Por último, cabe destacar que Python tiene una sintaxis muy visual, gracias a una notación indentada (con márgenes) de obligado cumplimiento. Para separar las porciones de código en Python se debe tabular hacia dentro, colocando un margen al código que iría dentro de una función o un bucle. Esto ayuda a que todos los programadores adopten unas mismas notaciones y que los programas de cualquier persona tengan un aspecto muy similar.



# Capítulo 4

## Diseño e implementación

A continuación se detalla el proyecto de Aplicación de Web Progresiva (PWA) del que trata este trabajo de fin de grado.

### 4.1. Arquitectura general

Este proyecto gira en torno a la arquitectura software del modelo cliente-servidor<sup>1</sup>, como se detalla en la figura 4.1. Destacamos la parte del cliente, principalmente manejada por Angular y más concretamente con una PWA. Por otro lado, la parte del servidor, controlada en su punto central por Node.js, dispone de una base de datos MongoDB y un par de programas desarrollados en Python que ayudarán a la automatización, optimización y organización de las salas.

#### 4.1.1. Cliente

En el lado del cliente tenemos toda la parte desarrollada con Angular, específicamente con el paquete `@angular/pwa v.0.1101.1`. Al tratarse de una PWA y tratar con el framework Bootstrap

---

<sup>1</sup>La arquitectura cliente-servidor es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

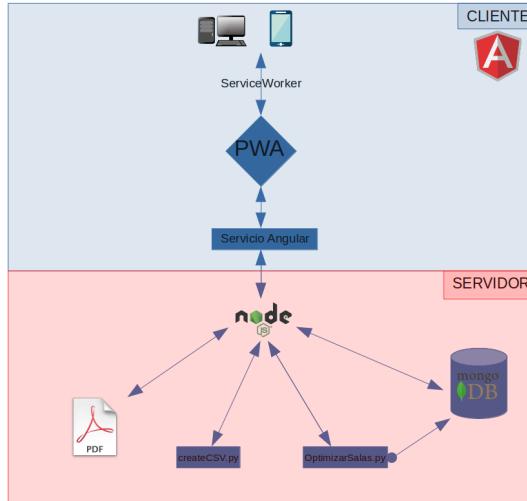


Figura 4.1: Arquitectura general.

permite que todas las pantallas de la aplicación sean `responsive`<sup>2</sup>, lo que quiere decir que estas pantallas se ajustarán automáticamente a la resolución del dispositivo en el que se muestran. Esto es, en la parte cliente nuestro diseño está centrado principalmente en dispositivos móviles, llegándose a asemejar a una app nativa.

Debido al uso de la cámara del dispositivo, los server workers (proporcionados por el paquete `@angular/pwa v.0.1101.1`) y un login inicial de usuario (usuario y claves), es de obligado cumplimiento que la página se muestre con el protocolo seguro de transferencia de hipertexto (en inglés, Hypertext Transfer Protocol Secure o HTTPS) para garantizar que la transferencia de datos no se verá comprometida en ningún momento y sea seguro. Este tipo de comunicación segura se consigue con un certificado de claves públicas para el servidor web.

En las vistas del cliente diferenciamos dos variantes principales una vez que se hace login:

- Los usuarios que serán generalmente alumnos.
- El administrador de la aplicación, este último tendrá unas vistas propias y unos privilegios por parte del servidor que no tendrá ningún otro usuario. Estos privilegios le permitirán modificar elementos de la base de datos, obtener información de todas las charlas, seminarios y de los alumnos apuntados a cada una de ellas.

<sup>2</sup>El diseño web adaptable (también diseño web adaptativo o responsive; este último calco del inglés responsive web design) es una filosofía de diseño y desarrollo cuyo objetivo es adaptar la apariencia de las páginas web al dispositivo que se esté utilizando para visitarlas

### 4.1.2. Servidor

En el lado del servidor se realiza el desarrollo back-end al completo, desde la recepción de peticiones HTTPS hasta el acceso a datos o la descarga de ellos en formato CSV (comma-separated values, del inglés valores separados por comas) o PDF [7]. La parte central de este servidor gira en torno a Node.js, específicamente en su framework Express<sup>3</sup> [3] que nos proporciona los mecanismos para poder manejar las peticiones HTTPS (GET, POST, PUT y DELETE) de los distintos usuarios. Una vez recibida una de estas peticiones, por medio del manejador proporcionado por Express realizaremos una u otra acción, entre las principales de login, registro, getEvento entre otros. Destacaremos dos:

- **/getCreditoEventos:** Esta petición, en concreto, se encarga de recibir como parámetros strings: el nombre, apellidos y el DNI del usuario. Por medio de la pestaña de *certificado* y más concretamente apretando el botón “*Recibir Créditos*” rellena una plantilla PDF suministrada por la URJC donde se adjudican al alumno los créditos correspondientes por asistir a una de estas charlas o seminarios que previamente se apuntó.
- **/zip:** Esta petición solo es posible recibirla cuando es el administrador el que la requiere. Se recibe como parámetros dos elementos: Un identificador del evento dentro de la base de datos y el nombre del evento. Esta petición lanzará un script desarrollado en Python llamado createCSV.py. Este script se encargará de crear el CSV con los datos obtenidos de la base de datos y dejarlo en una ruta específica dentro del servidor, donde más tarde se recogerá y se mandara vía HTTPS al usuario administrador.

Esta petición tiene dos caminos:

- Si el administrador quiere un evento en particular, el ID del evento corresponderá a su identificador dentro de la base de datos. El servidor devolverá un CSV que llenará con los datos obtenidos de la base de datos en función de ese identificador recibido.
- Si el administrador quiere obtener todos los eventos, el id del evento será 0. Cuando obtenemos este identificador el servidor creará un archivo comprimido (en formato

---

<sup>3</sup>Express es el framework web más popular de Node.js, y es la librería subyacente para un gran número de otros frameworks web de Node.js populares. Proporciona mecanismos para: Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).

.zip) que contendrá varios CSV con los eventos en los que haya más de un alumno inscrito. Estos CSV tendrán el mismo formato y la misma información que el CSV devuelto en el caso anterior.

## 4.2. Modelo de datos

### 4.2.1. MongoDB

Para el almacenamiento de las distintas aulas disponibles, los alumnos inscritos y los horarios de las distintas charlas y seminarios tenemos los siguientes modelos de datos almacenados en la base da datos de MongoDB:

- Eventos: En este modelo de datos tendríamos almacenados JSON del con el formato que se puede apreciar en la figura 4.2.

Eventos	
aula	string
hora	<i>Horario</i>
nombre_evento	string
descripcion	string
id_evento	integer
id_qr_evento	integer
id_usuarios	[Arr. <i>Usuarios</i> ]

Figura 4.2: Modelo Eventos.

- Aula: Nombre correspondiente al aula donde se impartirá la charla o seminario almacenada en el Modelos de datos de *Salas*.
- Hora: Hora correspondiente a la que se impartirá la charla o seminario, corresponderá a una hora almacenada en el Modelo de datos de *Horario*.
- Nombre\_evento: Nombre correspondiente al Evento que se expondrá a la hora y en el aula especificado en los dos puntos anteriores.
- Descripción: Descripción el evento presentado.
- Id\_Evento: Identificador interno del evento presentado.
- Id\_qr\_evento: Identificador del QR correspondiente al evento presentado.
- Id\_usuarios: Array del mode de datos de *Usuarios* donde se especifica los usuarios apuntados al evento presentado.

- Usuarios: En este Modelo de datos tendríamos almacenados JSON del con el formato que se aprecia en la figura 4.3.

Usuarios	
nick	string
password	string
id_usuario	integer
logueado	boolean
nombre_apellidos	string
DNI	string

Figura 4.3: Modelo Usuarios.

- Nick: Nombre correspondiente al alumno reconocido internamente en la URJC.
- Password: Contraseña encriptada en AES192.
- Id\_usuario: Identificador interno del usuario.
- Logueado: Indicador True (conectado) o False (desconectado) del usuario.
- Nombre\_apellidos: Nombre y apellidos del Usuario.
- DNI: DNI del Usuario.
- Salas: En este Modelo de datos tendríamos almacenados JSON del con el formato que se puede apreciar en la figura 4.4.

Salas	
aula	string
tam	string

Figura 4.4: Modelo Salas.

- Aula: Nombre correspondiente al aula y como se la reconoce internamente en la URJC.

- Tam: Tamaño máximo de la sala.
- Horario: En este Modelo de datos tendríamos almacenados JSON del con el formato de la figura 4.5:

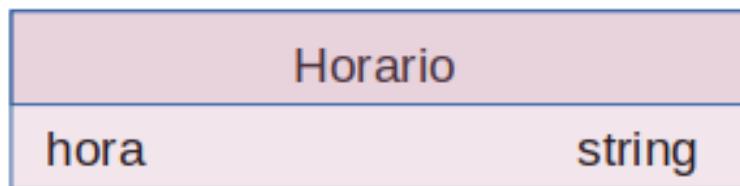


Figura 4.5: Modelo Salas.

- Hora: Horas disponibles para todas la aulas en los días especificados.

#### 4.2.2. Registros de Alumnos y Eventos

Este apartado solo será visible por parte el administrador de la aplicación o del docente encargado en gestionar la semana cultural donde se impartan estas charlas y seminarios. Como vimos anteriormente, el administrador tiene la posibilidad de descargarse un archivo en formato CSV con toda la información de un Evento o de todos ellos, obteniendo un listado de los alumnos que se han apuntado a una determinada charla y que han validado su asistencia por medio de nuestra aplicación. El formato que tienen estos CSV corresponden a la figura 4.6.

	A	B	C
1	Usuario	Id Usuario	Validado
2	dendrina		1 True
3	mperez		53 False
4	jserrano		44 True
5	agarcia		78 True
6	jlopez		56 False

Figura 4.6: Modelo del archivo CSV.

- Usuario: Corresponde a un string almacenado en la base de datos de MongoDB con el formato de modelo de datos de *Usuarios*

- Id\_usuario: String correspondiente campo id\_usuario del modelo de datos de *Usuarios*
- True o False en el campo “validado” hace referencia a si el alumno ha asistido al evento y ha escaneado el código QR disponible al finalizar el mismo dando constancia así que asistió a él, campo a True, o por el contrario está apuntado pero no asistió a él siendo en este caso False el valor de este campo.

#### 4.2.3. Plantilla documento de obtención créditos

El PDF que obtiene el alumno es un documento oficial de la Universidad Rey Juan Carlos. En este documento se le otorgan un determinado número de créditos (fijados por el responsable de los seminarios o charlas) por acudir a las los seminarios o charlas que previamente se apuntó mediante la aplicación de la que estamos tratando en este TFG. La obtención de este certificado se hará de forma automática una vez que el alumno haya acudido a un número mínimo de seminarios fijados por el responsable de las mismas.

 Universidad  
Rey Juan Carlos

**1** D.Gregorio Robles Martínez Subdirector de Investigación, Relaciones Internacionales y Relaciones con la Industria de la Universidad Rey Juan Carlos

CERTIFICA: Que el **2** alumno/a de la Universidad **Rey Juan Carlos**, Don/ña **3** con DNI nº **4** ha participado en el Foro de Empleo de la URJC durante los días **5** **6 y 7 de Marzo de 2020** y ha cumplimentado satisfactoriamente los requisitos exigidos para la obtención de: **5** **0,5 CREDITOS ECTS.**

Y para que conste, y a petición del interesado, expido el presente certificado en Madrid, a **de** **de 202**

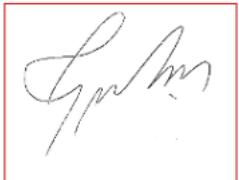
**6** 

Figura 4.7: Plantilla PDF créditos.

El formato que tendrá el documento para la obtención de los créditos será el indicado en la figura 4.7.

1. Nombre del profesor encargado de los seminarios y charlas.
2. Nombre del alumno al que se le otorgarán los créditos.
3. DNI del alumno.
4. Fechas correspondientes a los días en los que tendrán lugar las charlas o seminarios.
5. Número de créditos que obtendrá el alumno por asistir a los seminarios.
6. Firma del profesor o responsable encargado de otorgar los créditos al alumno.

## 4.3. Diseño e implementación por funcionalidad

### 4.3.1. Cliente

#### Login

La aplicación guarda una estética distintiva de la universidad, con el uso de sus logos y colores, como podemos apreciar en la figura 4.8. Esta aplicación se integra dentro de las aplicaciones de la Universidad Rey Juan Carlos para el uso tanto de profesores como de alumnos.

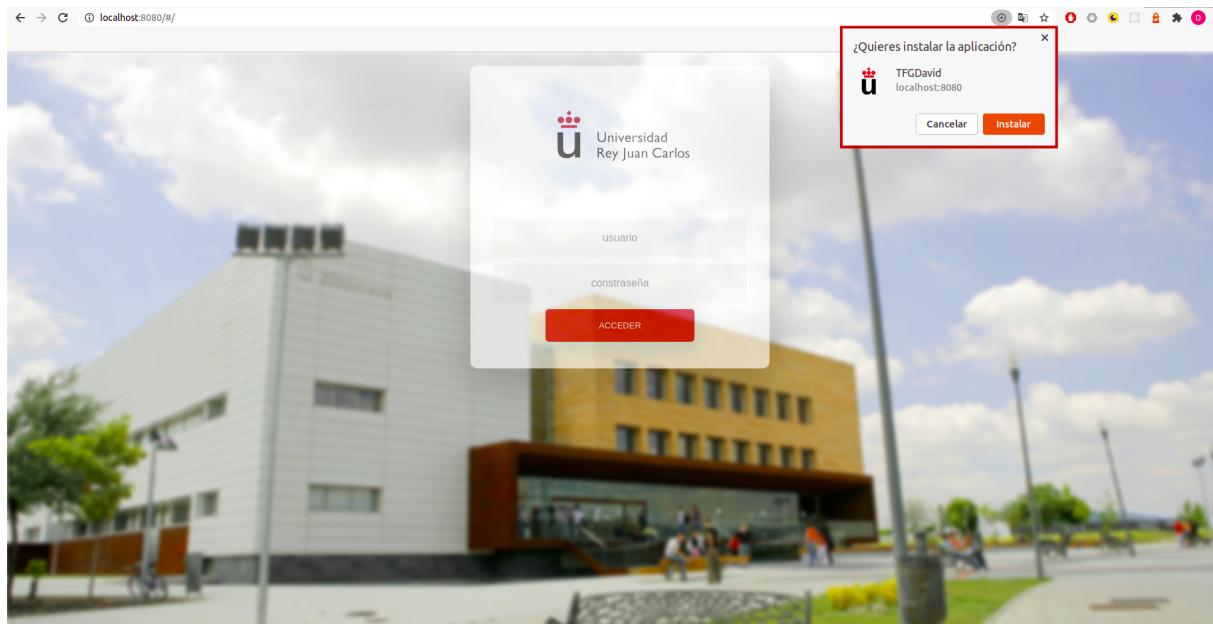


Figura 4.8: Página Login

La página de *login* consta de un menú de inicio de sesión, donde el usuario ingresará sus credenciales propias de la Universidad Rey Juan Carlos. En esta página destacamos dos puntos claves de por qué esta aplicación es una PWA.

En primer lugar, tal y como se observa en la figura 4.8, el navegador tanto en ordenador como en dispositivos móviles nos da la opción de instalar la aplicación en nuestro dispositivo, creando un acceso directo a ella como si de una aplicación nativa Android (en el caso móvil) o como una extensión Chrome (en el caso de utilizar un navegador Chrome en el ordenador) se tratase.

En segundo lugar, como observamos en la figura 4.9, la aplicación dispone también de un *service worker*.

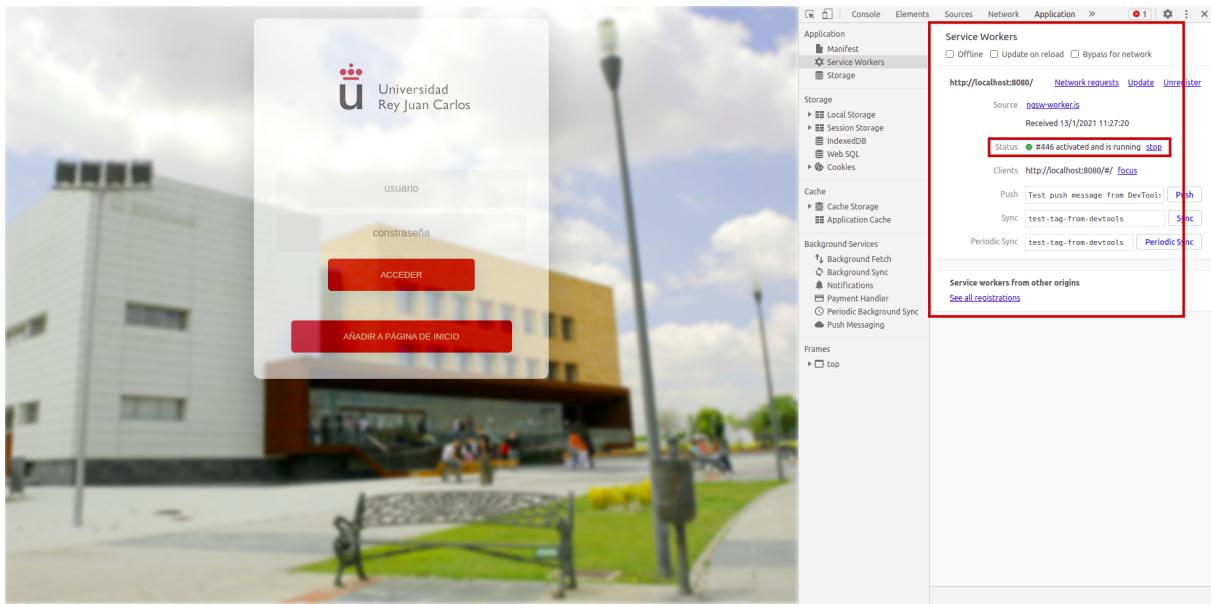


Figura 4.9: Página Login

Un *service worker* es un script que permite interceptar y controlar los requests de la red y el almacenamiento del cache del navegador. Gracias a los *service workers*, los programadores web pueden crear sitios web con una experiencia offline y utilizar la aplicación sin conexión únicamente con las páginas que previamente se hayan guardado en la caché del dispositivo.

Como los *service workers* tienen acceso al manejo de los requests, pueden llegar a ser muy poderosos (y peligrosos), especialmente si son usados con mala intención. Esto es uno de los motivos por lo cuales usar *HTTPS* es muy importante, para prevenir ataques de terceros.

## Usuario

Nada más ingresar en la aplicación con el nombre de usuario y contraseña, se cargará la página principal. Durante todas las ventanas de la aplicación dispondremos de un menú, en el que estarán accesibles las 3 páginas principales de nuestra aplicación (1,2,3) y un botón para hacer *logout* (4) de la misma, como observamos en la figura 4.10:



Figura 4.10: Menú principal de la aplicación

1. Página principal: En la siguiente vista dispondremos de la información sobre las charlas y seminarios a los que el alumno se inscribió.

- En el campo **Título** se describe el título de la charla o seminario.
- En el campo **Aula** se informa del aula dentro de la Universidad Rey Juan Carlos donde tendrá lugar la charla.
- En el campo **Hora** se especifica la hora a la que tendrá lugar el evento.

Título	Aula	Hora
Telefonica	aula 323	11:00-11:30
BBVA	aula 324	10:00-10:30
Kabel	aula 321	10:00-10:30

Figura 4.11: Página de inicio.

Dentro de cada fila del evento en el que estamos apuntados y tenemos previsto ir aparecerán dos botones: “Validar” (1) y “Desapuntarse” (2).

- El botón validar: Solo estará accesible una vez el evento haya finalizado, para que los alumnos que han podido asistir puedan validar mediante la aplicación que han

ido a dicha charla. Una vez validado el evento y habiendo recibido la comprobación por parte del servidor, debajo del campo aula aparecerá un ícono (3) mostrando que la validación se ha desarrollado con normalidad. En la siguiente figura 4.11 podemos observar cada uno de los botones previamente indicados:

Una vez presionemos el botón de validar (ya que hemos asistido al evento en cuestión y queremos dar fe de ello), la aplicación nos llevará a la pestaña de validación, figura 4.12. En esta pestaña la aplicación utilizará la cámara del dispositivo, ya sea un móvil, tablet u ordenador portátil para leer el código QR correspondiente al evento que queremos validar.

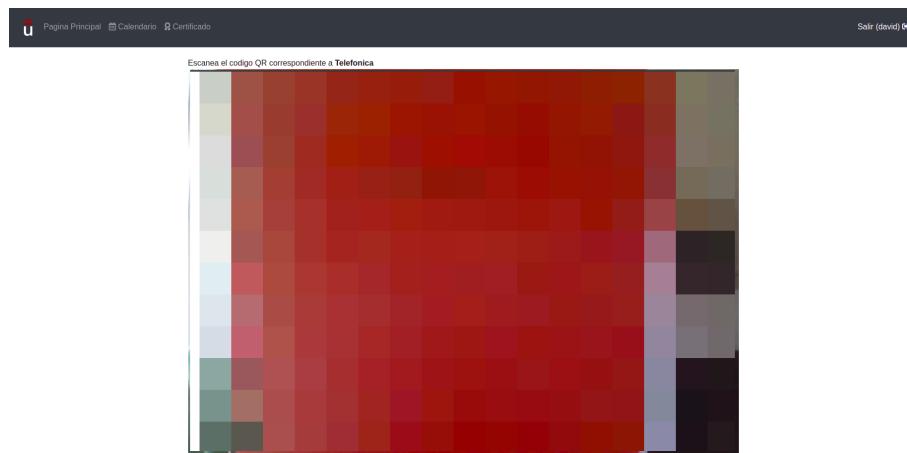


Figura 4.12: Pestaña de validación de QR

Para validar el evento, basta con mostrar un QR válido y la aplicación obtendrá un mensaje de aceptación por parte del servidor. Si por el contrario mostramos un QR que no correspondería al QR del evento en cuestión, se nos avisará con un mensaje en la pantalla que el QR proporcionado no es válido. Un ejemplo de este caso puede verse representado en la figura 4.13, en el que escaneamos un QR correspondiente al evento del BBVA cuando debería ser el QR del evento Telefónica.

- **El botón desapuntarse:** Este botón estará accesible desde el momento en el alumno se haya apuntado al evento. Al presionarlo, el evento en cuestión desaparecerá de la lista de eventos que se muestra en esta pantalla de inicio.

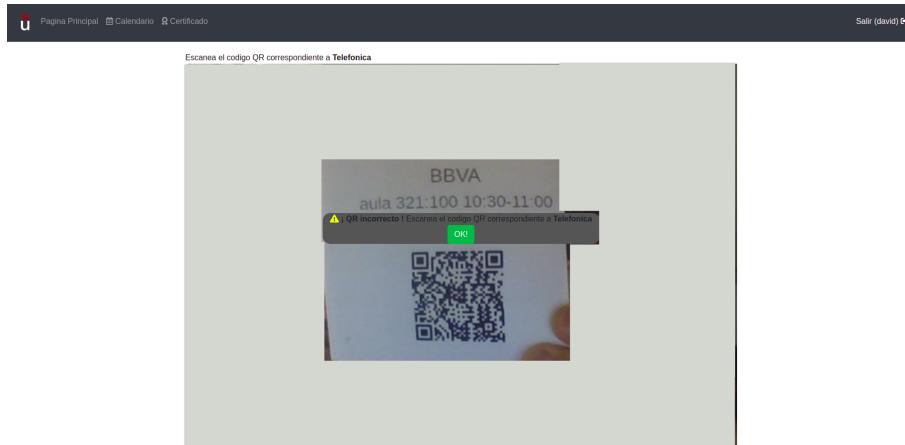


Figura 4.13: Pestaña de validación de QR con mensaje de error

2. Calendario: En la vista siguiente (ver figura 4.14) obtenemos un calendario con las principales charlas y seminarios disponibles durante las jornadas habilitadas por la Universidad Rey Juan Carlos.

A screenshot of a calendar page. The top navigation bar is identical to Figure 4.13. The main content is a grid table with five columns: 'HORAS / AULAS' and 'AULA 323', 'AULA 321', 'AULA 322', and 'AULA 324'. The rows represent time slots: '10:00-10:30', '10:30-11:00', '11:00-11:30', and '11:30-12:00'. Each cell contains the name of the event or company scheduled for that specific slot and location.

HORAS / AULAS	AULA 323	AULA 321	AULA 322	AULA 324
10:00-10:30	Huawei	Kabel	Thales Alenia Space	BBVA
10:30-11:00	Westcon	IBM	Accenture	
11:00-11:30	Telefónica	Askey-ASUS	SAP	
11:30-12:00	CT Ingenieros	Deloitte		

Figura 4.14: Página de calendario

Cada evento ocupa una posición en el calendario. Así mismo cada uno de ellos actúa como un botón en el que presionándolo nos redirige a la página de información del mismo, como se muestra en la figura 4.15. En esta página de información es donde el alumno se podrá apuntarse (dando *click* en “Apuntarse”) y expresar su intención de asistir el mismo. Completando la página hay una pequeña descripción del evento que estamos viendo.

The screenshot shows a web page for a seminar at Aula 323 from 11:00-11:30. It features a photo of a man speaking at a podium, a brief description of the collaboration between Fundación Telefónica and Universidad Rey Juan Carlos, and a green 'Apuntarse' (Sign Up) button.

Figura 4.15: Página de descripción del evento y botón de apuntarse

3. Certificado: Esta vista, figura 4.16, se compone de un botón que el alumno tendrá disponible únicamente cuando tenga la oportunidad de obtener los créditos que le corresponden por asistir a estos seminarios. El requisito de mínimo de asistencia para poder optar a tener el certificado de los créditos lo impondrá el profesor encargado de ello o en su defecto el administrador que se encargue del mantenimiento de la página y de dichas jornadas aquí propuestas.

The screenshot shows a page for receiving credits. It includes a message about obtaining a certificate for attending seminars, and a green 'Recibir Créditos' (Receive Credits) button.

Figura 4.16: Página para obtener el certificado de créditos

Una vez el alumno tiene disponible el botón de “Recibir Créditos” un pop-up saldrá en pantalla sirviendo una vista previa del PDF que será el certificado que le otorgue los créditos, como se puede observar en la figura 4.17. Estará a su vez disponible un botón de descarga del PDF para poder almacenarlo en su dispositivo.

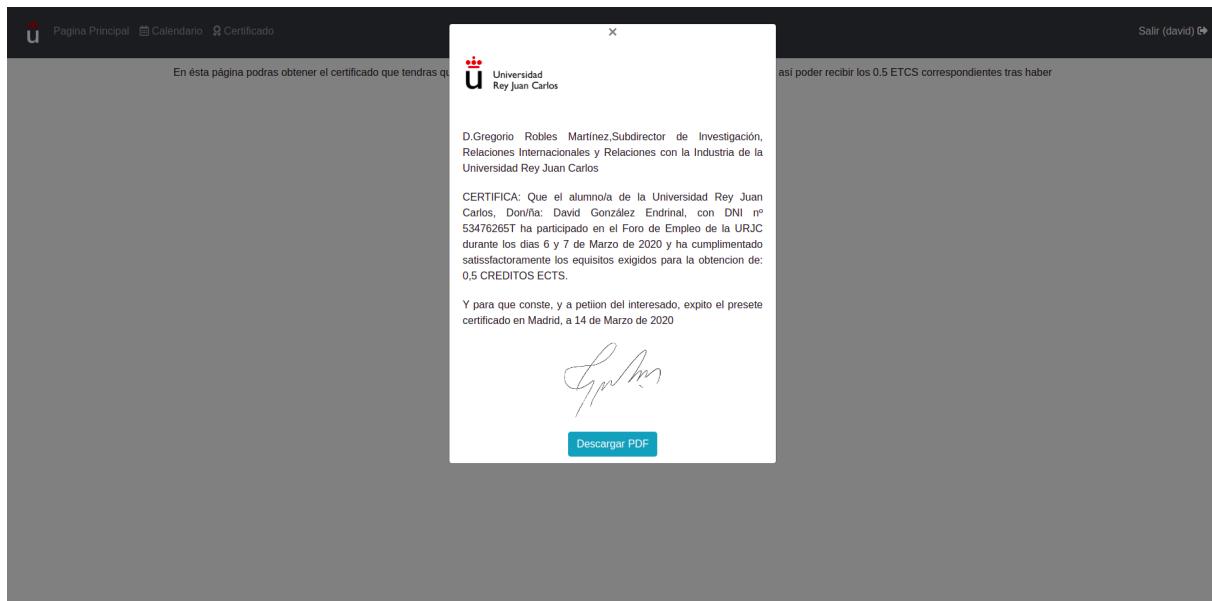


Figura 4.17: Pop-up con la vista del certificado

## Administrador

Una vez iniciado sesión con la cuenta del administrador o de la persona encargada de mantener y administrar la aplicación, veremos dos pestañas claramente diferenciadas:

1. Lista de Eventos: En esta pestaña representada en la figura 4.18, el administrador de la aplicación tendrá un listado completo de todos los eventos o seminarios que habrá durante la semana o los días que duren cada uno de ellos. En esta primera vista se dará la información de: Nombre del evento, aula donde se impartirá y la hora del día a la que tendrá lugar dicho evento.

The screenshot shows a dark-themed user interface for event management. At the top, there's a navigation bar with a logo, a search bar, and buttons for 'Salir' (Logout) and 'Descargar PDF Todos' (Download PDF All). Below the navigation is a horizontal menu bar with three tabs: 'Lista Eventos' (selected), 'Calendario Eventos', and 'Estadísticas'. The main content area displays a table of events:

Nombre	Lugar	Tiempo
Telefonica	aula 323:200	11:00-11:30
Askey-ASUS	aula 321:100	11:00-11:30
Accenture	aula 322:40	10:30-11:00
BBVA	aula 324:150	10:00-10:30
CT Ingenieros	aula 323:200	11:30-12:00
Deloitte	aula 321:100	11:30-12:00
Kabel	aula 321:100	10:00-10:30
Hauwei	aula 323:200	10:00-10:30
IBM	aula 321:100	10:30-11:00
SAP	aula 322:40	11:00-11:30
Thales Alenia Space	aula 322:40	10:00-10:30
Westcon	aula 323:200	10:30-11:00

Figura 4.18: Vista de administrador de la pestaña Lista de Eventos

Destacamos en esta vista 4.19, el botón de “*Descargar PDF Todos*”, presionándolo enviaremos una petición al servidor que nos contestará con un archivo comprimido en formato ZIP llamado “Eventos.zip” que contendrá todos los CSV de los Eventos en los que haya apuntada más de una persona. Este CSV es del mismo formato que el CSV que se descargaría dando uno por uno a cada evento con el botón “*Generar CSV*”, que explicaremos más adelante en el siguiente punto. El archivo comprimido en formato ZIP descargado quedaría de la siguiente manera:

Haciendo click en cada una de las filas correspondientes a cada uno de los eventos, entraremos en una ventana representada en la figura 4.20, donde nos informará más detalladamente sobre el evento en cuestión. Con una breve descripción del evento (1), su QR de validación correspondiente (2) y el listado de los usuarios inscritos a él, así como que usuario ha validado o no la asistencia al evento con el campo “*Validado*” True o False (3). Destacamos de esta vista el botón “*Generar CSV*” (4) que descargaría en nuestro equipo en formato CSV el mismo listado de usuarios que estamos viendo en esta misma página.

2. Calendario Eventos: En esta pestaña, figura 4.21, el administrador tendrá de forma visual la información de cuando y donde va a tener lugar un evento cualquiera. Seguido del nombre del evento. Entre paréntesis, hay un número correspondiente al número de alumnos apuntados a dicho evento.

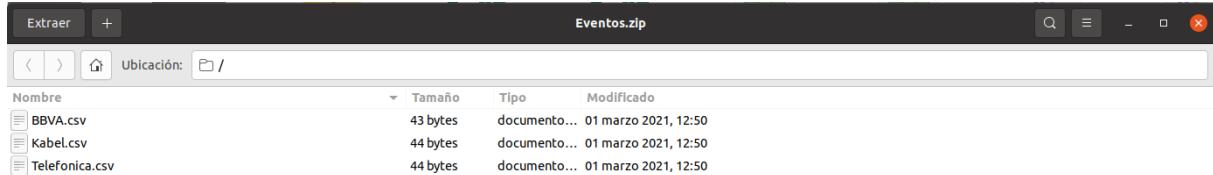


Figura 4.19: Contenido de Eventos.zip descargado al presionar *Descargar PDF Todos*

**Telefonica**

aula 323:200 11:00-11:30

1

<p>La Fundación Telefónica, en colaboraci&ocute;n con la Universidad Rey Juan Carlos est&araacute; presente en el seminario donde contribuir a la mejora de la actividad profesional de los profesores y el desarrollo de sus alumnos y, por ello, del sistema educativo. Con este evento se busca contribuir al desarrollo y adaptaci&ocute;n del sistema educativo a un entorno fuertemente digitalizado y cambiante: formaci&ocute;n de los profesores del siglo XXI, para que puedan promover un aprendizaje del siglo XXI con estudiantes del siglo XXI.</p><p><a></a></p>

<p>El seminario est&araacute; dirigido a todas aquellas personas interesadas en conocer las posibilidades que ofrecen las herramientas digitales en el aula. En las sesiones se intercalar&acutearon ponencias y talleres participativos llevados a cabo por referentes nacionales en el &aaacute;mbito de la educaci&ocute;n y la tecnolog&iacute;a. </p>

2

3

4

Usuarios Inscritos: 1

Generar CSV

Id Usuario	Usuario	Validado
1	david	false

Figura 4.20: Vista de administrador de la ventana informativa del evento de Telefónica

Destacamos varias cosas en esta pestaña: En primer lugar, el botón “*Optimizar*”. Este botón envía un mensaje al servidor que lanzará un script<sup>4</sup> de optimización programado en Python para reubicar los eventos en horarios y aulas de la forma más óptima posible, más

<sup>4</sup>En informática, un script es un término informal que se usa para designar a un programa relativamente simple.

adelante, en el siguiente punto, cuando se explique la parte correspondiente al servidor desarrollaré más su funcionamiento. Este botón modificará la posición de todos o la gran mayoría de los eventos dentro del marco horario mostrado en esta pestaña.

Por otro lado, es posible modificar de forma individual un evento. Para ello bastará con hacer click sobre él y toda su información se volcará en los desplegables que se muestran en la parte inferior de la Figura 4.21, se podrá modificar tanto el aula como las horas que estén disponibles para todos los eventos, en este caso entre las aulas 323, 321, 322 y 324 y en los horarios de 10:00 a 12:00.

Tras presionar el botón “Modificar” se enviarán las actualizaciones que hayamos hecho en los desplegables serán enviadas al servidor y podremos verlas en esta misma vista.

Lista Eventos		Calendario Eventos				Estadísticas	
HORAS / AULAS		AULA 323:200	AULA 321:100	AULA 322:40	AULA 324:150		
10:00-10:30		Hauwei_(0)	Kabel_(1)	Thales Alenia Space_(0)	BBVA_(1)		
10:30-11:00		Westcon_(0)	IBM_(0)	Accenture_(0)			
11:00-11:30		Telefonica_(1)	Askey-ASUS_(0)	SAP_(0)			
11:30-12:00		CT Ingenieros_(0)	Deloitte_(0)				

Figura 4.21: Pestaña Calendario Eventos

3. Estadística: En esta pestaña el administrador tendrá de forma rápida y visual un resumen de lo que están siendo las jornadas culturales durante el día o los días que duren los eventos.

En esta vista se pueden obtener datos como:

- Número de alumnos totales que han asistido a todas las charlas disponibles durante las jornadas culturales.
- El nombre de los alumnos y al número total de charlas que han asistido.

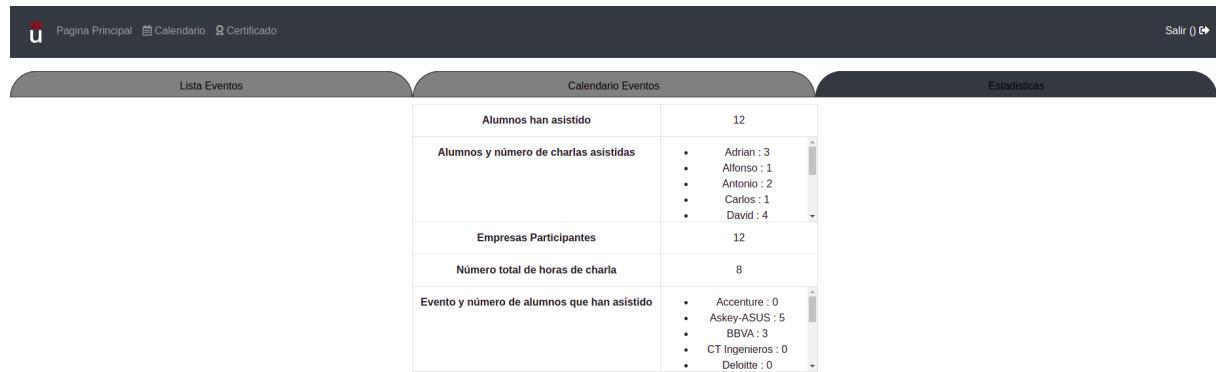


Figura 4.22: Pestaña Estadística

- Número de empresas participantes durante todas las jornadas culturales.
- El número total de horas de charlas que se han impartido.
- Número de alumnos totales que han asistido a todos los eventos disponibles durante las jornadas culturales.
- El nombre de las diferentes charlas propuestas y el número de alumnos que han asistido a ellas

Esta información es muy valiosa a la hora de sacar una estadística completa de lo que están siendo las jornadas culturales en tiempo real y permiten mostrar el interés que están provocando en los alumnos de la universidad.

### 4.3.2. Servidor

El servidor está desarrollado en un entorno de trabajo que funciona en tiempo de ejecución, de código abierto y multiplataforma: este entorno es *Node.js*. Añade un soporte para APIs donde se incluye una comunicación HTTP entre el cliente y él mismo. Por todo esto obtenemos un gran rendimiento de la aplicación y también una gran escalabilidad. Dentro de *Node.js*, el framework más usado y que uso en este caso es **Express**, cuya biblioteca se compone de un gran número de framework populares dando mecanismos para el manejo de peticiones HTTP y HTTPS (en mi caso) y mantener así una comunicación segura cliente-servidor. Los métodos de comunicación usados en el servidor mediante HTTPS son los típicos usados en este tipo de aplicaciones:

- GET: El método GET solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.
- POST: El método POST se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.
- PUT: El modo PUT reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.
- DELETE: El método DELETE borra un recurso en específico.

Al margen ya de la parte Node.js del servidor, hay dos programas desarrollados en Python de los que el servidor hace uso cuando el cliente así lo requiere. Es importante saber que el único cliente o usuario que puede realizar una petición al servidor y poder lanzar así estos programas es el usuario bajo el nick de “admin”, en los siguientes puntos pasaremos a explicar el funcionamientos de dichos programas:

1. CreateCSV.py: Cuando el administrador presiona dentro de la pestaña “Lista Eventos” cualquier fila correspondiente a un evento, y dentro de ella el botón “GenerarCSV” o bien en la misma pestaña de “Lista Eventos” presiona el botón “Generar PDF Todos”, el servidor dentro de la URL `'/zip/:id_evento/:nombre_evento'` recibe la petición del cliente que quiere obtener o bien un CSV o bien un .zip con todos los CSV de los eventos dentro de él. Desde Node.js se lanza Python mediante el comando `./createCSV.py -idevento` (el parámetro idevento solo se pondrá en el caso en el que se quiera el CSV de un evento en concreto, para obtener el .zip de todos el parámetro se omitirá).

Una vez el programa en Python de createCSV.py ha sido lanzado, pasa a buscar en la base de datos mediante un SELECT de idevento o recorriendose todos los eventos que haya. Obteniendo así, una lista de usuarios asociados al evento, con el campo validado True o False según corresponda para cada uno de los usuarios.

Cuando se obtiene esta lista y se finaliza la ejecución del SELECT, pasamos a volcar los datos obtenidos en un fichero .CSV. Cuando terminamos de llenar el fichero, en el caso de querer uno en concreto, la ejecución terminaría aquí y sería devuelto el fichero al cliente que solicitó tal acción al servidor. Por otro lado, en el caso de querer todos los

eventos, se irá completando la acción pasada con todos los eventos en los que haya más de una persona inscrita en él. Cuando termine pasará a crear un .zip con todos estos .CSV creados. Una vez terminado, como en el caso anterior, le será devuelto al cliente un .zip como requirió con su petición inicial.

La forma en la que el servidor y el cliente están conectados para poder enviar la información en formato .CSV o en formato .ZIP es mediante un pipe stream.<sup>5</sup>

La funcionalidad y el esquema de dicho funcionamiento quedan representadas en la figura 4.23.

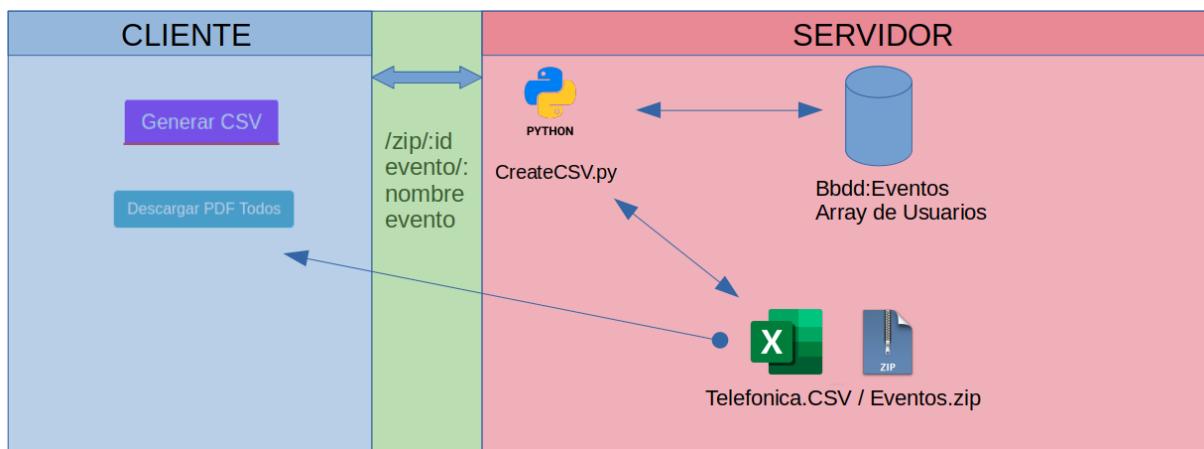


Figura 4.23: Esquema funcionamiento createCSV.py

2. optimizarSalas.py: El siguiente programa se ejecuta cuando el administrador dentro de la pestaña “Calendario Eventos” presiona el botón “Optimizar”. Este botón lanza un mensaje a la ruta ‘/optimizar’ dentro del endpoint donde está escuchando el servidor, que ejecuta el comando ‘./optimizarSalas.py’ lanzando el programa en Python.

Este programa tiene como objetivo la optimización y reubicación de los eventos en función de los huecos libres que quedan con respecto a la gente que se ha apuntado o quiere asistir al evento, con los huecos libres que quedan en la sala. Funciona de la siguiente manera:

El programa, una vez se ejecuta realiza las siguientes acciones:

---

<sup>5</sup>Una tubería (pipeline o cauce) consiste en una cadena de streams conectados de forma tal que la salida de cada elemento de la cadena es la entrada del próximo. Permiten la comunicación y sincronización entre procesos. Es común el uso de buffer de datos entre elementos consecutivos.

- Lee de la base de datos cada uno de los eventos que están programados que se van a llevar a cabo en los días que transcurran todos ellos.
- De cada evento obtiene el número de alumnos o personas que están apuntados a él.
- Con este número obtenido en el paso anterior calcula los huecos libres que tienen en todas las aulas disponibles, desechando los números negativos que se obtienen cuando van a asistir mas personas que huecos libres tiene la sala
- Una vez tenemos relacionados todos los eventos con las aulas en función de los huecos libres que hay, el diccionario se ordena de menor a mayor, teniendo prioridad de posición los eventos que tengan menor número de huecos libres en una sala determinada.

La adjudicación del evento a una sala se va realizando de forma ascendente y ordenada hasta que el aula en cuestión complete todas las horas disponibles que tiene dentro de los días que están programados que tengan lugar estos eventos. En nuestro caso, se completará cuando en una sala se haya completado el horario de 10:00 a 12:00. Cuando esto ocurre el evento no tendrá la opción de ser colocado en esa aula y pasará a la siguiente aula, donde en función del tamaño, las personas que vayan y el número de huecos disponibles sea el óptimo.

Si se llega a el caso en el que un evento no encuentra lugar dentro de ningún aula, este evento quedará excluido y no se le adjudicará ninguna sala. El evento se mostrará fuera de la matriz de “Calendario Eventos” y tendrá que ser el administrador quien lo posicione a mano, dentro de un aula en concreto teniendo en cuenta la limitación de aforo mediante el botón “Modificar” que se encuentra en la parte inferior de la pestaña de “Calendario Eventos” (ya explicamos su funcionamiento en el apartado anterior).



# Capítulo 5

## Pruebas y funcionamiento

El principal objetivo sobre el que gira este proyecto de fin de grado es el de crear una aplicación web híbrida que funcione tanto en dispositivos móviles como en navegadores web, esto lo hemos conseguido creando una Aplicación Web Progresiva o PWA.

Para ver que realmente hemos conseguido que nuestra aplicación sea una PWA basta con realizar las siguientes pruebas:

### 5.1. Server Workers

Nuestra aplicación contiene *server workers* que son propios del paquete `@angular/pwa v.0.1101.1`. Para ello basta con mirar en la consola de desarrolladores de Google Chrome o de cualquier navegador, ir al apartado service workers y ver que está activado y ejecutándose como observamos en la figura 5.1.

### 5.2. Instalación en un dispositivo móvil

Podemos instalar nuestra aplicación en un dispositivo móvil como si de una aplicación nativa, Android o IOS se tratase. Para ello presionamos el botón “Añadir a pagina de inicio” una vez que estemos navegando con el dispositivo móvil en cuestión por nuestra aplicación. Una vez hayamos aceptado y se haya instalado, nuestra aplicación PWA se verá como una aplicación nativa dentro de nuestro dispositivo móvil como podemos observar en la figura 5.2.

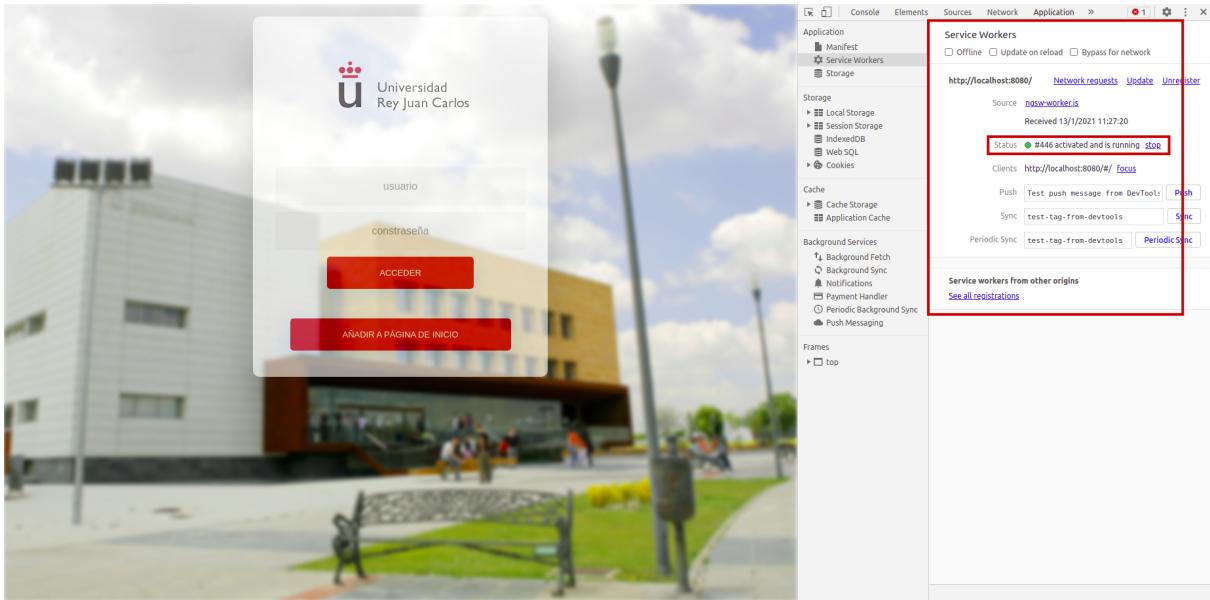


Figura 5.1: Service Workers funcionando en la PWA

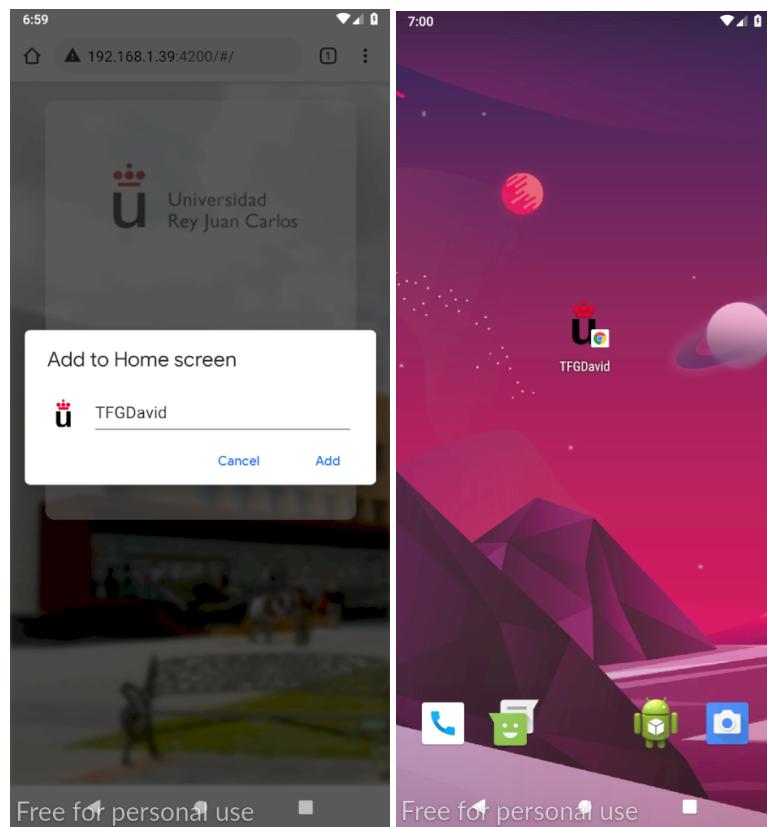


Figura 5.2: PWA en la pantalla de inicio del dispositivo móvil

### 5.3. Optimización

En cuanto a la parte del servidor, un punto importante a tener en cuenta sobre la validación del proyecto y el buen funcionamiento de este es que Python es el encargado de optimizar las

salas en función al número de alumnos apuntados y al número de huecos disponibles funcione correctamente. Para ello podemos lanzar manualmente el propio Python encargado de ello con el comando *python optimizarSalas.py* obtendríamos un resultado como vemos a continuación en la figura 5.3.

```
david@davidubuntu:/escritorio/TFG/backEnd$ python optimizarSalas.py
...
('Kabel': 86, 'Westcon': 94, 'IBM': 134, 'Huawei': 74, 'BBVA': 0, 'Askey-ASUS': 4, 'Thales Alenia Space': 109, 'Deloitte': 188, 'Telefonica': 139, 'Accenture': 7, 'CT Ingenieros ': 68, 'SAP': 191)
10:00-10:30
Al evento Kabel van (86) personas que va a al aula aula 321 a la hora 10:00-10:30 con 14 huecos libres
.....
10:30-11:00
Al evento Westcon van (94) personas que va a al aula aula 321 a la hora 10:30-11:00 con 6 huecos libres
.....
10:00-10:30
Al evento IBM van (134) personas que va a al aula aula 324 a la hora 10:00-10:30 con 16 huecos libres
.....
11:00-11:30
Al evento Huawei van (74) personas que va a al aula aula 321 a la hora 11:00-11:30 con 26 huecos libres
.....
10:00-10:30
Al evento BBVA van (0) personas que va a al aula aula 322 a la hora 10:00-10:30 con 40 huecos libres
.....
10:30-11:00
Al evento Askey-ASUS van (4) personas que va a al aula aula 322 a la hora 10:30-11:00 con 36 huecos libres
.....
10:30-11:00
Al evento Thales Alenia Space van (109) personas que va a al aula aula 324 a la hora 10:30-11:00 con 41 huecos libres
.....
10:00-10:30
Al evento Deloitte van (188) personas que va a al aula aula 323 a la hora 10:00-10:30 con 12 huecos libres
.....
11:00-11:30
Al evento Telefonica van (139) personas que va a al aula aula 324 a la hora 11:00-11:30 con 11 huecos libres
.....
11:00-11:30
Al evento Accenture van (7) personas que va a al aula aula 322 a la hora 11:00-11:30 con 33 huecos libres
.....
11:30-12:00
Al evento CT Ingenieros van (68) personas que va a al aula aula 321 a la hora 11:30-12:00 con 32 huecos libres
.....
10:30-11:00
Al evento SAP van (191) personas que va a al aula aula 323 a la hora 10:30-11:00 con 9 huecos libres
.....
Actualizo en la tabla: Kabel a la hora 10:00-10:30 con aula 321:100
Actualizo en la tabla: Westcon a la hora 10:30-11:00 con aula 321:100
Actualizo en la tabla: IBM a la hora 10:00-10:30 con aula 324:150
Actualizo en la tabla: Huawei a la hora 11:00-11:30 con aula 321:100
Actualizo en la tabla: BBVA a la hora 10:00-10:30 con aula 322:40
Actualizo en la tabla: Askey-ASUS a la hora 10:30-11:00 con aula 322:40
Actualizo en la tabla: Thales Alenia Space a la hora 10:30-11:00 con aula 324:150
Actualizo en la tabla: CT Ingenieros a la hora 11:30-12:00 con aula 321:100
Actualizo en la tabla: Telefonica a la hora 11:00-11:30 con aula 324:150
Actualizo en la tabla: Accenture a la hora 11:00-11:30 con aula 322:40
Actualizo en la tabla: Deloitte a la hora 10:00-10:30 con aula 323:200
Actualizo en la tabla: SAP a la hora 10:30-11:00 con aula 323:200
```

Figura 5.3: Salida tras ejecutar Python optimizarSalas.py

Revisando dichos resultados nos damos cuenta que el programa tiene en cuenta los huecos libres de las aulas, el tamaño de la misma y los usuarios inscritos optimizando su distribución.

## 5.4. Seguridad

Se use el protocolo HTTPS (HyperText Transfer Protocol Secure, Protocolo de transferencia de hipertexto), que es un protocolo de comunicación de Internet que protege la integridad y la confidencialidad de los datos de los usuarios entre sus ordenadores y el sitio web. En este caso al usar partes vulnerables del móvil, como puede ser la cámara al leer el código QR o en el momento de autenticar al usuario, es necesario el uso de una capa más de seguridad.

En el caso de la autenticación la contraseña del usuario, viaja cifrada dentro de este protocolo y dentro de la base de datos también, por si esta base de datos sufre un ataque no se vea comprometida la seguridad de los usuarios de la aplicación. El cifrado de la contraseña es del tipo AES192 (Advanced Encryption Standard)<sup>1</sup> y para validar este cifrado basta con

<sup>1</sup>El Advanced Encryption Standard, abreviado AES, se usa con el fin de cifrar datos y de protegerlos contra

ver la base de datos y el valor del campo “password” dentro de la tabla “Usuarios”, en este caso para el usuario *david* el valor del campo es: *4232a5e307a998a3ef2601c7d0a4c704*, un valor esperado para este tipo de cifrados.

---

cualquier acceso ilícito. El método criptográfico emplea para este objetivo una clave de longitud variada y se denomina según la longitud de clave usada AES-128, AES-192 o AES-256.

# Capítulo 6

## Resultados

El presente proyecto pretendía ofrecer una herramienta de gestión y administración de los eventos o seminarios que se ofrecerán durante la semana cultural, una semana donde los alumnos tienen la oportunidad de interactuar con las distintas empresas del sector en el que están desarrollando su formación universitaria.

La aplicación desarrollada en este proyecto pretende abordar y resolver los objetivos descritos en el punto 2, donde hemos obtenido los siguientes resultados:

- Hemos conseguido, mediante un script desarrollado en Python, automatizar el proceso de reparto de las clases/horas para las distintas charlas y seminarios en función de los alumnos que se apuntaron a ellas. Este era un objetivo primordial cuando se abordó este TFG, ya que con la automatización de este proceso le quitamos mucha carga de trabajo al encargado de tal función, en este caso un docente de la universidad.
- La aplicación está capacitada para crear informes, mediante archivos CSV, sobre los alumnos que se han apuntado y han asistido a cada una de las charlas, estos informes serán muy valiosos tanto para la universidad como para las propias empresas que impartirán estas charlas para así poder enfocarlas en una determinada dirección logrando así llamar más la atención de los alumnos y conseguir levantar un interés mayor en ellos.
- Para los alumnos: se ha implementado una aplicación de fácil manejo, muy accesible y que permite mantener informado al alumno en todo momento del horario, tipo de charla que se impartirá y el contenido de la misma. También facilita y agiliza el proceso de

verificación de la asistencia a a cada una de ellas y por consiguiente, adquirir los créditos correspondientes.

- En cuanto a las tecnologías usadas, se demuestra que disponemos de herramientas muy potentes para poder desarrollar aplicaciones híbridas, que no son muy conocidas actualmente, pero que en un futuro podrán imponerse frente a las aplicaciones nativas. Con este tipo de desarrollos se podrá optimizar el tiempo que conlleva crear una aplicación de este tipo, reduciendo el número de personal necesario para abarcar un mayor público y tener un mayor abanico de posibilidades a la hora de orientar tu aplicación.

# **Capítulo 7**

## **Conclusiones**

### **7.1. Aplicación de lo aprendido**

En cuanto a conocimientos y habilidades obtenidas durante todo el Grado de Ingeniería Telemática son importantes todos los relacionados con la programación, tanto front como back, destacando los siguientes lenguajes:

1. Todo lo relacionado con Web (HTML, CSS y JavaScript) en la Asignatura de *Aplicaciones Telemáticas* con los profesores Jesús M. González Barahona y Gregorio Robles Martínez.
2. Seguridad Web con la asignatura *Seguridad en redes de Ordenadores* con el profesor Enrique Soriano.
3. Parte back con el lenguaje de programación Python en la asignatura de *Laboratorio de Administración y Gestión de Redes y Sistemas* con el profesor Miguel Ortúño.
4. Por último destacaría todo mi paso general por la carrera ya que de una u otra manera he ido adquiriendo conceptos, conocimientos, habilidades y el manejo de distintas herramientas en todas las asignaturas en mayor o menor medida.

## 7.2. Lecciones aprendidas

Con este TFG he podido aprender y adquirir conocimientos que si bien en las distintas asignaturas de la universidad he podido tener la primera toma de contacto, aquí he podido desarrollarlos a unos puntos claramente diferenciados:

1. Desarrollo Web: Se trata quizás del punto más importante de todos. La aplicación aún siendo híbrida y teniendo la capacidad de ser instalada en un dispositivo móvil, en esencia, es una aplicación Web desarrollada en Angular que por medio de un módulo le permite ser una aplicación híbrida. Esta tecnología en desarrollo web no llegamos a impartirla del todo en la universidad, por lo que por medio de este TFG he teniendo que aprender a manejarla de principio a fin, con todo lo que ello conlleva.
2. PWA: Al inicio de este proyecto, cuando tuve la primera reunión con Gregorio, no tenía demasiado conocimientos sobre las aplicaciones híbridas, Gregorio insistió mucho en que la aplicación tendría que poder instalarse en cualquier dispositivo móvil, fue ahí cuando descubrí este tipo de aplicaciones y todos los requisitos que deben cumplir las aplicaciones Angular para poder ser una aplicación de este tipo.
3. Módulos Angular, SSL y automatización: Para poder tener una aplicación funcional, de fácil manejo y con un hilo de ejecución automatizado ha sido necesario entender el manejo de estos tres puntos principales. En cuanto a los módulos de angular he tenido que investigar, aprender e integrar en la aplicación módulos como el lector de QR y generador de QR, unos módulos que tienen gran proyección de futuro y en los que su uso en el futuro será mas que necesario.

Automatización de todo el back de la aplicación con bibliotecas en Python, con consultas a bases de datos y generador de documentos y estadísticas en función de los resultados obtenidos. En cuanto al lenguaje Python tenía una base muy sólida en el, no obstante he tenido que aprender y estudiar cada biblioteca utilizada para entender su funcionamiento y poder integrarla satisfactoriamente en este proyecto.

Uno de los puntos claves de esta aplicación es la seguridad. Para que una PWA tenga acceso a las diferentes funcionalidades del móvil, la aplicación tiene que cumplir una serie de requisitos que permitan garantizar la seguridad del dispositivo. Toda esta seguridad la

conseguimos con herramientas criptográficas y comunicación cifrada Cliente-Servidor. Este protocolo de comunicación que protege la integridad y la confidencialidad de los datos del usuario es HTTPS. En cuanto a este tipo de seguridad, en la carrera he tenido la oportunidad de adquirir unos conocimiento básicos. No obstante, a lo largo de este TFG me he topado con distintos errores, problemas y puntos que tenía que resolver para poder obtener una aplicación totalmente funcional y segura.

En resumen, este TFG me ha dado la oportunidad de poder aprender tecnologías punteras, que me apetecían aprender y que muy seguro en un futuro me será de gran ayuda haber podido tener contacto con ellas, en mayor o menor medida.

### 7.3. Trabajos futuros

Esta aplicación podrá ser todo lo compleja y podrá abordar todos los puntos necesarios que el administrador o encargado de las charlas que se imparten en la universidad requiera. En primer lugar, tendrá que poderse integrar en todo el funcionamiento interno de la universidad, esto es, que esté conectada a la base de datos donde estén todos los alumnos de la misma.

Al ser una aplicación que se usará mayoritariamente en dispositivos móviles un objetivo claro que veo es la posibilidad de mandar alertas y notificaciones a los usuarios cuando se acerque un evento en el que está apuntado o tener la posibilidad de sincronizar el calendario interno del móvil a los eventos apuntados.

En cuanto al lado de la seguridad, poner un poco más difícil que el alumno pueda hacer trampas indicando que ha asistido a una charla mediante el QR pero sin haber ido a ella. Esto se podría conseguir teniendo acceso al posicionamiento GPS del dispositivo, para saber que si está leyendo el QR y está en el lugar donde esta disponible el QR para leer, el alumno habrá asistido (muy probablemente) a la charla indicada.

La forma de guardar las charlas y eventos en la base de datos, a día de hoy cuando voy a crear un evento tengo que insertar manualmente el evento en la base de datos, su descripción y todo lo relacionado con él. Esto en un futuro se podría solucionar creando una aplicación web donde las distintas empresas tuvieran acceso a ella y rellenaran todos los campos que a día de hoy se tienen que insertar a mano.



# Apéndice A

## Manual de usuario

- Front: Dentro de `/TFGDavid/` ejecutamos el proyecto de Angular: **ng serve -ssl** (si queremos que corra bajo HTTPS) o `ng serve` en HTTP normal, destacamos que ambos comandos permiten ejecutar el proyecto en angular en modo “desarrollo” en `localhost:4200`, si queremos ejecutarlo en producción, como se ejecutaría en un servidor final para verlo en internet, tenemos que realizar dos procedimientos, uno ejecutar **ng build -prod**, compilará la aplicación Angular en un directorio de salida llamado `dist/` y por otro lado, para ejecutar el proyecto compilado en producción usaremos el comando:

```
http-server -p 8080 -c-1 ~/dist/TFGDavid
```

- Back: Dentro de `/backEnd/` para lanzar el servidor:

```
node app.js
```

En el caso de querer lanzar el programa Python de optimizarSalas o createCSV

```
python optimizarSalas.py o python createCSV.py
```

- Base de datos MongoDB: Se deberá ejecutar y mantenerse escuchando en `localhost:27017` dirección que por defecto utiliza esta base de datos.

```
sudo systemctl start mongod
```

Para comprobar que esta corriendo el proceso podemos usar el comando:

```
sudo systemctl status mongod
```



# Bibliografía

[1] Página CSS3.

[http://www.w3schools.com/css/css3\\_intro.asp/](http://www.w3schools.com/css/css3_intro.asp/).

[2] Página de Angular pwa.

<https://www.npmjs.com/package/@angular/pwa>.

[3] Página de Angular.

<https://angular.io>.

[4] Página de Angular Cli.

<https://cli.angular.io>.

[5] Página de Angular Material.

<https://material.angular.io>.

[6] Página de Bootstrap.

<https://getbootstrap.com/>.

[7] Página de Csv reader.

<https://docs.python.org/3/library/csv.html>.

[8] Página de Expressjs.

<https://expressjs.com/es/>.

[9] Página de Git.

<https://git-scm.com>.

[10] Página de Mongo client.

[https://pymongo.readthedocs.io/en/stable/api/pymongo/mongo\\_client.html](https://pymongo.readthedocs.io/en/stable/api/pymongo/mongo_client.html).

- [11] Página de Mongodb.  
[https://www.mongodb.com.](https://www.mongodb.com)
- [12] Página de Mongoose.  
[https://mongoosejs.com.](https://mongoosejs.com)
- [13] Página de Ngx-scanner.  
[https://www.npmjs.com/package/@zxing/ngx-scanner.](https://www.npmjs.com/package/@zxing/ngx-scanner)
- [14] Página de Node.js.  
[https://nodejs.org/es/.](https://nodejs.org/es/)
- [15] Página de Nodemon.  
[https://nodemon.io.](https://nodemon.io)
- [16] Página de npm.  
[https://www.npmjs.com/.](https://www.npmjs.com/)
- [17] Página de Pymongo.  
[https://pypi.org/project/pymongo/.](https://pypi.org/project/pymongo/)
- [18] Página de Python.  
[https://www.python.org/.](https://www.python.org/)
- [19] Página de Typescript.  
[https://www.typescriptlang.org/.](https://www.typescriptlang.org/)
- [20] Página de Zipfile36.  
[https://pypi.org/project/zipfile36/.](https://pypi.org/project/zipfile36/)
- [21] Página Fontawesome.  
[https://fontawesome.com/v4.7.0/.](https://fontawesome.com/v4.7.0/)
- [22] Página HTML.  
[https://www.w3schools.com/html/html5\\_intro.asp.](https://www.w3schools.com/html/html5_intro.asp)