UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Desarrollo de una aplicación web como catálogo de buenas prácticas de accesibilidad

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Rodríguez Yáñez, Adrián

Tutor/a: Albert Albiol, Manuela

Cotutor/a: Torres Bosch, María Victoria

CURSO ACADÉMICO: 2021/2022

Resumen

En este Trabajo Final de Grado (TFG) se propone llevar a cabo la creación de una aplicación web que sirva como catálogo de buenas prácticas en el desarrollo de aplicaciones web y móvil accesibles.

Aunque actualmente existe normativa para el desarrollo accesible (https://www.w3.org/WAI/) con guías y algunas herramientas de ayuda, ésta se da en forma de documentación muy densa que no resulta práctica cuando un programador se enfrenta al desarrollo accesible. Para mejorar esta situación se propone este TFG, cuyo objetivo principal es facilitar a los desarrolladores la tarea de identificación de las normas de accesibilidad y las buenas prácticas que pueden aplicar para cumplir estas normas.

En particular, el catálogo que se propone se centrará en la construcción de aplicaciones web y Android. Dicho catálogo reunirá un conjunto de buenas prácticas que se explicarán de una manera intuitiva utilizando ejemplos, y mostrándolos tanto con enlaces al código en Github, como en la misma aplicación web mediante imágenes.

Estas buenas prácticas se organizarán atendiendo a los criterios de la normativa actual sobre accesibilidad, utilizando, entre otros, la categorización de la accesibilidad en los niveles A, AA y AAA. Se usará el framework Vue.js para desarrollar la aplicación web.

Palabras clave: Accesibilidad, desarrollo, web, Android, catálogo, Vue.js, Front-End.

Abstract

This Final Degree Project (TFG) aims to create a web application that serves as a catalogue of good practices in the development of accessible web and mobile applications.

Although there are currently standards for accessible development (https://www.w3.org/WAI/) with guidelines and some help tools, this is in the form of very dense documentation that is not practical when a programmer is faced with accessible development. In order to improve this situation, this TFG is proposed, whose main objective is to make it easier for developers to identify accessibility standards and the good practices they can apply to comply with these standards.

In particular, the proposed catalogue will focus on building web and Android applications. The catalogue will bring together a set of best practices that will be explained in an intuitive way using examples, and showing them both with links to the code on Github, as well as in the web application itself through images.

These good practices will be organised according to the criteria of the current accessibility regulations, using, among others, the categorisation of accessibility in levels A, AA and AAA. The Vue.js framework will be used to develop the web application.

Keywords: Accessibility, development, web, Android, catalogue, Front-End, Vue.js.

Índice de contenidos

Índice de figuras

1 Introducción

La accesibilidad es definida por la R.A.E. como "*De fácil acceso o trato*", entre otras acepciones. Un dispositivo accesible sería aquel que pueda ser usado fácilmente; en el campo de aplicaciones móviles y/o webs el significado es el mismo. Por lo tanto, una aplicación accesible, es aquella que puede ser usada fácilmente por cualquier usuario independientemente de sus capacidades.

En España un 9% de la población presenta algún tipo de discapacidad (Observatorio estatal de la discapacidad, 2022). La Figura 1 muestra el porcentaje de personas con discapacidad en España por rango de edad. Estas discapacidades pueden estar relacionadas con limitaciones físicas, mentales, intelectuales o sensoriales, que, cuando se encuentran con diversas barreras, impiden la acción del individuo en su plenitud.

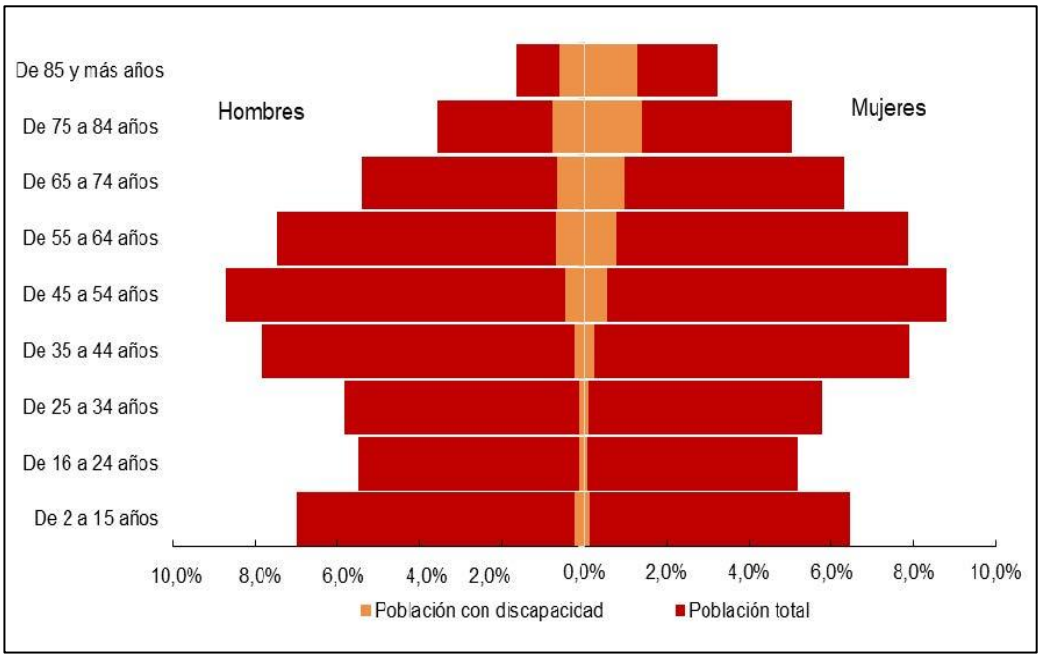


Figura 1 Porcentaje de personas con discapacidad en España

Por todo esto, se hace indispensable el desarrollo de aplicaciones accesibles. Cuando se está programando una aplicación que va a salir al mercado, siempre hay que pensar en cómo conseguir los máximos usuarios posibles, y no es una buena estrategia de negocio impedir el uso de tu producto a algunos usuarios.

Ese es uno de los motivos por los cuales se hace necesario que las aplicaciones sean accesibles. A parte de eso, por motivos éticos se debe evitar excluir a diferentes personas de usar un producto. Además de todo esto, desde el año 2018, existe una ley que obliga a todas las aplicaciones móviles y webs, exclusivamente del sector público, a tener un grado de accesibilidad mínimo (Carreras, 2018).

Los desarrolladores de aplicaciones se pueden apoyar en diferentes guías, para conseguir aplicaciones accesibles. El World Wide Web Consortium (W3C) (World Wide Web Consortium, 2018), es la organización referente para desarrollo web y móvil accesible y es un punto de referencia para los desarrolladores.

1.1 Motivación

La idea de este trabajo surge cuando el autor se encuentra con la necesidad de realizar una aplicación accesible para Android mientras se realizaban las prácticas curriculares. Se necesitaba alguna guía de accesibilidad, que no fuera ni muy extensa ni muy técnica, para ayudar a conseguir desarrollar una aplicación accesible.

Aunque existen diferentes guías de accesibilidad, encontramos que éstas pueden llegar a ser muy densas y poco prácticas para un desarrollador. Esto nos dio la idea de llevar a cabo un trabajo en el que se diera respuesta a este vacío.

1.2 Objetivos

El objetivo principal de este trabajo de fin de grado es el de ayudar a los desarrolladores de aplicaciones Android y web a conseguir que sus aplicaciones sean accesibles y que lo puedan hacer de una manera fácil, rápida y sencilla. Se pretende ayudar a desarrolladores software con cualquier nivel de experiencia, sin importar si acaban de comenzar o si tienen un nivel mucho más avanzado en el desarrollo software.

Este objetivo principal se puede desgranar en los siguientes subobjetivos:

- Desarrollar una aplicación web que reúna un catálogo de buenas prácticas de accesibilidad.
- Conseguir una aplicación accesible, que esté al alcance de todo el mundo, y que pueda ser consultada y utilizada por cualquier desarrollador.

- Adquirir conocimientos sobre el proceso completo de la creación y despliegue de una aplicación. Durante los estudios en el Grado de Ingeniería Informática, especialmente en la rama de Ingeniería de Software, los conocimientos enseñados y aprendidos se refieren siempre al trabajo en equipo, por lo tanto, se consideraba importante el hecho de aprender a realizar todo este tipo de cosas individualmente.

1.3 Impacto esperado

El impacto que se espera a raíz de este proyecto es el de que los desarrolladores de aplicaciones Android y aplicaciones web, puedan realizar su aplicación accesible, tanto desde el inicio del desarrollo, como teniendo la aplicación desarrollada. Estos desarrolladores serán hispanohablantes, ya que la aplicación está en español. No obstante, como veremos más adelante en la sección *7.2 Trabajo futuro,* es una tarea pendiente añadir la traducción a varios idiomas.

Además, se espera también, que los usuarios de la aplicación entiendan mejor lo que significa la accesibilidad y la importancia que tiene, dado que existe un gran número de usuarios que tienen la necesidad de estas características en las aplicaciones.

1.4 Estructura

Este trabajo se ha dividido en los siguientes capítulos:

- **Introducción:** Se detalla una introducción a este trabajo, los objetivos que se van a perseguir, la motivación que incita a realizar el proyecto y el impacto que se espera.
- **Estado del arte:** Se exponen diferentes aplicaciones que son similares y que han sido analizadas.
- **Análisis del problema:** Se introduce qué tipo de aplicación es el catálogo y se comenta el presupuesto que ha tenido el autor.
- **Diseño de la solución:** Se define el primer diseño de la aplicación web y como se ha estructurado la información del catálogo.
- **Desarrollo de la solución:** Trata sobre cómo ha sido el desarrollo de la web.
- **Pruebas:** Se exponen diferentes pruebas de usabilidad que se han realizado a diferentes usuarios para valorar la usabilidad de la aplicación.

- **Conclusiones:** Conclusiones a las que se ha llegado después de todo este Trabajo Final de Grado.

2 Estado del arte

En este capítulo se ha realizado un análisis de las diferentes herramientas y recursos que existen para desarrollar aplicaciones Android y webs accesibles, similares a la desarrollada, para así compararla y entender qué cosas necesita. Las herramientas analizadas provienen de ámbitos oficiales y de entornos donde el desarrollo de aplicaciones está presente.

2.1 Android

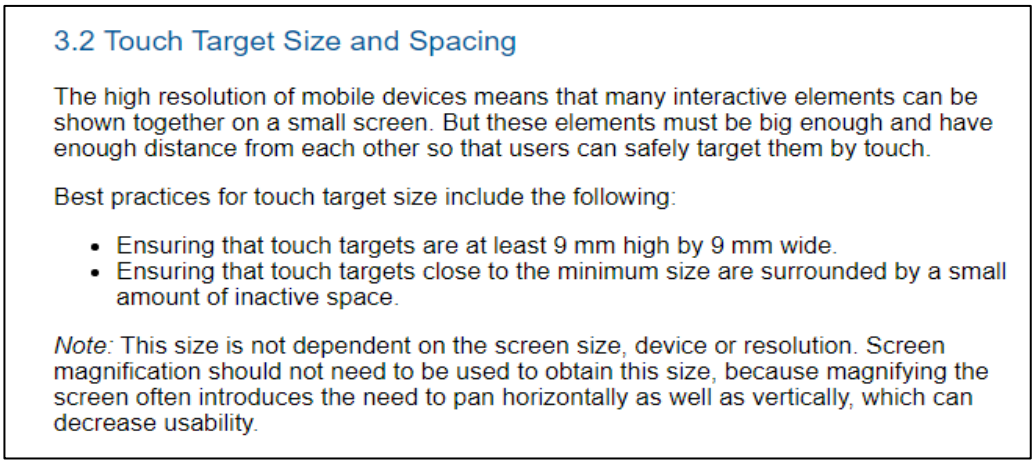
2.1.1 W3C

La web World Wide Web Consortium (W3C)[1], reserva un espacio para comentar sobre accesibilidad móvil. Pero no solo habla de móviles, sino que se dedica a comentar sobre diferentes dispositivos, como pueden ser teléfonos, tabletas, televisores, "Internet de las cosas", etc. Además de eso, se consideran diferentes cuestiones físicas como pantallas táctiles, pequeños tamaños de pantallas, diferentes tipos de entrada, etc… Y tampoco hay que olvidarse de que trata sobre el contenido web en móviles.

El inconveniente que nos presenta este apartado del W3C es que no especifica en ningún momento sobre algún sistema operativo móvil, por ello se nos puede complicar la fase del desarrollo (World Wide Web Consortium (Mobile Accesibility)). Además, todo lo que explica este recurso sobre la accesibilidad está en inglés.

Hay que mencionar, que a pesar de ser este recurso el oficial para el desarrollo móvil se ha decidido seguir mirando y apoyándose en otros recursos, puesto que éste no nos ofrece exactamente lo que estamos buscando, que es accesibilidad en aplicaciones Android. De todas formas, sí que se tiene en cuenta para el contraste con los demás recursos. En la Figura 2, podemos ver un pequeño ejemplo de cómo realiza estas explicaciones el recurso analizado.

[1] https://www.w3.org/WAI/standards-guidelines/mobile/



3.2 Touch Target Size and Spacing

The high resolution of mobile devices means that many interactive elements can be shown together on a small screen. But these elements must be big enough and have enough distance from each other so that users can safely target them by touch.

Best practices for touch target size include the following:

- Ensuring that touch targets are at least 9 mm high by 9 mm wide.
- Ensuring that touch targets close to the minimum size are surrounded by a small amount of inactive space.

Note: This size is not dependent on the screen size, device or resolution. Screen magnification should not need to be used to obtain this size, because magnifying the screen often introduces the need to pan horizontally as well as vertically, which can decrease usability.

Figura 2 W3C Accesibilidad para dispositivos móviles

2.1.2 Material Design de Google

Otro de los recursos que nos podemos encontrar es el del framework de Front-End para aplicaciones Android (entre otros), llamado Material Design, creado por Google. La web oficial de este framework dispone de una sección, que está enfocada a hablar sobre la accesibilidad en aplicaciones móviles.

Como podemos observar, este recurso es bastante útil y teórico. Su contenido es similar al comentado anteriormente, pero en este caso nos brinda menos información técnica. Destacaría la falta de parte práctica, como podría ser código de ejemplo, dado que es la documentación del framework oficial de Google para el desarrollo Front-End. También destacaría el uso de imágenes de ejemplo, útil para entender lo que estamos leyendo (Material Design). En la Figura 3 vemos un ejemplo, de una de las secciones de este recurso.

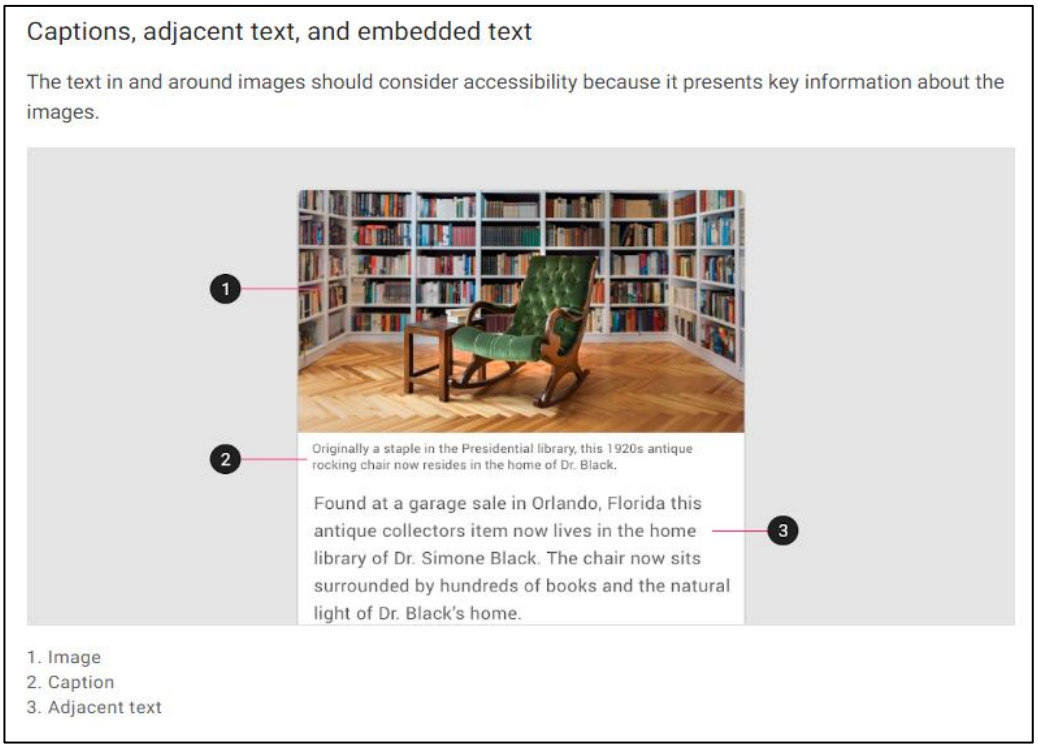


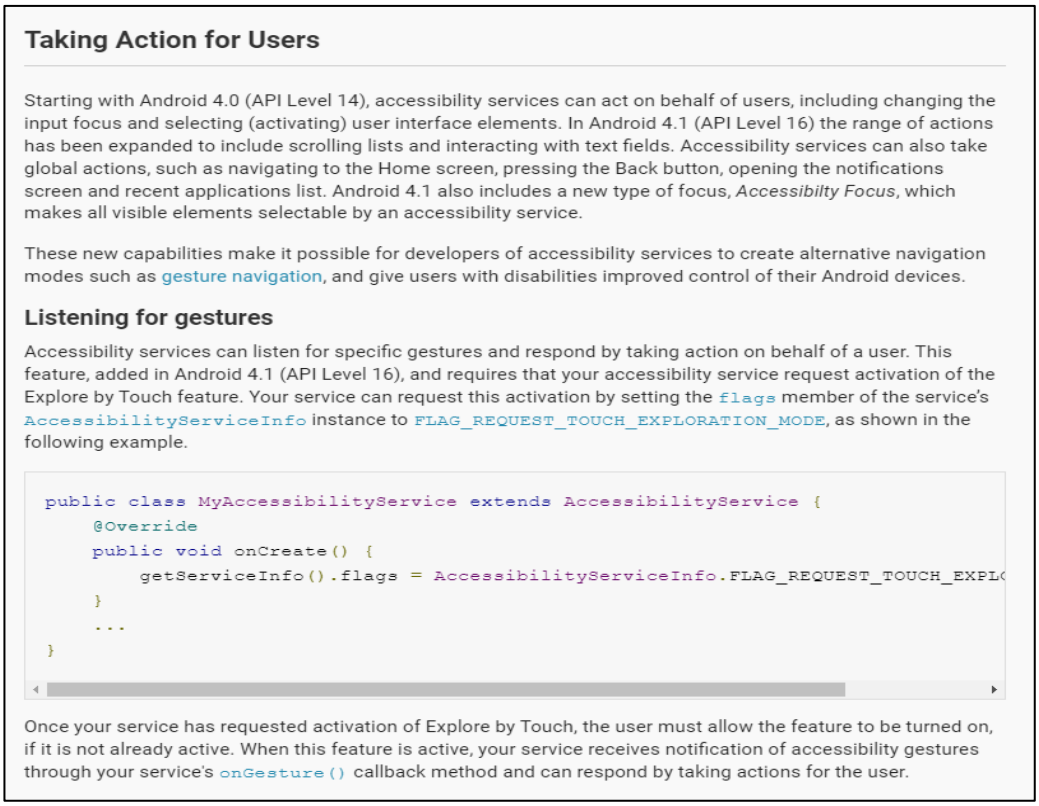
Figura 3 Muestra de la página de Material Design

2.1.3 MIT

El siguiente recurso que se ha analizado es el de la web oficial del MIT, concretamente la parte que está dirigida para los desarrolladores de Android[2]. Esta vez, cuando se refiere a la parte de accesibilidad lo hace de una forma mucho más técnica incluyendo hasta bloques de código, y, a diferencia de los anteriores ejemplos, éste se focaliza sobre Android.

Adicionalmente, cabe destacar que este recurso destina ligeramente un espacio a comentar cómo construir un servicio de accesibilidad en Android, pero eso ya no forma parte de este trabajo final de grado (Massachusetts Institute of Technology (MIT)). En la Figura 4, vemos una muestra de la página del recurso analizado.

[2] https://stuff.mit.edu/afs/sipb/project/android/docs/index.html



Taking Action for Users

Starting with Android 4.0 (API Level 14), accessibility services can act on behalf of users, including changing the input focus and selecting (activating) user interface elements. In Android 4.1 (API Level 16) the range of actions has been expanded to include scrolling lists and interacting with text fields. Accessibility services can also take global actions, such as navigating to the Home screen, pressing the Back button, opening the notifications screen and recent applications list. Android 4.1 also includes a new type of focus, *Accessibilty Focus*, which makes all visible elements selectable by an accessibility service.

These new capabilities make it possible for developers of accessibility services to create alternative navigation modes such as gesture navigation, and give users with disabilities improved control of their Android devices.

Listening for gestures

Accessibility services can listen for specific gestures and respond by taking action on behalf of a user. This feature, added in Android 4.1 (API Level 16), and requires that your accessibility service request activation of the Explore by Touch feature. Your service can request this activation by setting the `flags` member of the service's `AccessibilityServiceInfo` instance to `FLAG_REQUEST_TOUCH_EXPLORATION_MODE`, as shown in the following example.

```
public class MyAccessibilityService extends AccessibilityService {
    @Override
    public void onCreate() {
        getServiceInfo().flags = AccessibilityServiceInfo.FLAG_REQUEST_TOUCH_EXPL
    }
    ...
}
```

Once your service has requested activation of Explore by Touch, the user must allow the feature to be turned on, if it is not already active. When this feature is active, your service receives notification of accessibility gestures through your service's `onGesture()` callback method and can respond by taking actions for the user.

Figura 4 Muestra de la página del MIT

2.1.4 Libro Android Accessibility

El siguiente recurso junta un poco todo lo que acabamos de ver. Se trata del libro "*Android Accessibility*", escrito por Victoria Gonda, ingeniera de software y desarrolladora de Android.

Este libro está publicado por la página web *https://www.raywenderlich.com/*. Esta página está destinada, a enseñar desarrollo iOS, Swift, Android, Kotlin, Dart y Flutter, es decir, todo aquello destinado al desarrollo móvil. Disponen de más de 6000 tutoriales, más de 4000 vídeos de desarrollo móvil y han publicado más de 50 libros.

En *Android Accessibility*, nos podemos encontrar mucha documentación acerca de la accesibilidad en Android y de cómo aplicarla en base a los cuatro principios. Además, este libro, dispone de ejemplos prácticos, y hasta incluso de ejercicios.

2.2 Web

Por la parte de web, el principal de los recursos que nos podemos encontrar, al igual que el anterior, es el de la web oficial, World Wide Web Consortium(W3C). Ahí nos podemos encontrar con el WCAG, Web Content Accessibility Guidelines, el cual son unas guías para realizar el contenido web más accesible para las personas con necesidades. No hay que olvidarse, de que este recurso no es una página cualquiera, ni siquiera una introducción a la accesibilidad, sino que es el estándar para la accesibilidad. Esto quiere decir, que es lo oficial y, por lo tanto, para la parte de web, es el recurso que más se ha tenido en cuenta. Aunque también se han buscado otros recursos, se ha preferido centrarse en este, que es el oficial.

En esta guía nos encontramos tres versiones diferentes, que son la versión 2.0 publicada en diciembre de 2008, la versión 2.1 publicada en junio de 2018 y la versión 2.2 que ahora mismo existe un borrador y se publicará la versión final, a finales de este año 2022. Por estos motivos se ha preferido darle el enfoque, durante todo este trabajo final de grado, a las versiones 2.0 y 2.1. (World Wide Web Consortium (WCAG Overview))

La guía de accesibilidad nos introduce los documentos que nos explican cómo aplicar la accesibilidad para el contenido web. Este documento está bastante completo en cuanto a información teórica, ya que explica cómo aplicar la accesibilidad y cómo aplicar los diferentes criterios de conformidad según cada principio. También está bastante completo en cuanto a información práctica puesto que nos ofrece bastantes ejemplos con código acerca de cómo cumplir los requisitos. Además, incluye información acerca de los criterios de conformidad, que explicaremos en el capítulo *4.2 Arquitectura de la información*.

Sin embargo, el recurso carece de imágenes para mostrar como quedaría el resultado final aplicando las instrucciones indicadas (World Wide Web (WCAG QuickRef), 2018).

2.3 Problemas identificados

Vistos ya los diferentes recursos con los que se ha decidido trabajar, hemos identificado las siguientes carencias:

- Algunos de esos recursos no disponen de código o ejemplo prácticos, como puede verse en el apartado 2.1.1 y 2.1.2. Esto es un gran inconveniente, puesto que el desarrollador podría sentirse mucho más ayudado, si puede visualizar lo que le están explicando. Y, además, también le serviría para saber qué es lo que tiene que programar. Recordemos que ésta, es una aplicación hecha para desarrolladores, y, por lo tanto, no tendría mucho sentido indicar únicamente datos teóricos.
- Otros de estos recursos, a diferencia del anterior punto, sí que poseen ejemplos, pero éstos suelen ser con explicaciones complejas que dificultan el aprendizaje. No solo basta con ver código, también es necesario entenderlo.
- Los recursos analizados, suelen ser guías muy extensas, que a menudo, desmotivan al desarrollador, puesto que leer y entender todo requiere un gran esfuerzo y trabajo. Si contamos las palabras del recurso analizado en el apartado 2.2, vemos que existen más de 15.000. En el recurso que hemos analizado de Android, concretamente el apartado 2.1.1 vemos que puede llegar a tener 40.000 palabras aproximadamente. Además, en el apartado 2.1.4 hemos analizado la información expuesta en un libro de más de 200 páginas. Como vemos, son bastante extensos los recursos analizados.
- Especialmente para el apartado *2.1 Android*, existen recursos que hablan sobre desarrollo accesible en móviles, pero no existen muchos que hablen más concretamente sobre Android y que sean gratuitas.
- Todas estas guías, o recursos, les hace falta alguna funcionalidad para buscar más fácilmente el contenido. Normalmente, un desarrollador no busca leerse toda una guía de golpe, sino que va viendo los bloques que le interesan. Convendría crear algún componente que haga de buscador.

A partir de las siguientes carencias, se propone una herramienta que sea capaz de ofrecer una información que no sea complicada de entender para cualquier desarrollador y que no resulte densa.

3 Análisis del problema

Como hemos visto en el capítulo 1 y 2, tras analizar las guías de accesibilidad existentes, se detectó una carencia de herramientas para desarrolladores web y Android que ayudaran en el desarrollo de aplicaciones accesibles.

Esto nos llevó a plantearnos el desarrollo de un catálogo de accesibilidad que complementara a las guías de accesibilidad actuales y cubriendo las carencias que estas tenían. En este capítulo se ha llevado a cabo, un análisis acerca de las diferentes opciones que se disponen para desarrollar este proyecto.

3.1 Tipo de aplicación

Dado que la guía de accesibilidad que planteamos estaba dirigida a desarrolladores web y Android, había dos opciones posibles para el tipo de aplicación: una aplicación Android o una aplicación web. El despliegue de una aplicación Android, habría que realizarlo en la plataforma oficial de Android (Google Play Store). Para llevar a cabo ese despliegue, es obligado crearse una cuenta de desarrollador y eso lleva un único pago de 25 dólares. En cambio, realizar un despliegue de una aplicación web se puede hacer de forma gratuita. Una forma de realizar este despliegue es mediante GitHub Pages (Github) y únicamente es necesario disponer de una cuenta en Github. Por último, y es un punto clave, cuando se realiza el despliegue de una aplicación web, cualquiera con un dispositivo con conexión a internet puede acceder a él, pero, por el contrario, cuando se despliega una aplicación en la plataforma de Android, esa aplicación solo estaría disponible para quién disponga de un dispositivo Android.

3.2 Requisitos

Vistas las carencias en los capítulos 1 y 2, podemos establecer ciertos requisitos que debe de cumplir nuestra aplicación:

- **Imágenes de código:** Se debe mostrar el código de los ejemplos siempre que se pueda, ya que eso facilita el trabajo al desarrollador.
- **Explicaciones sencillas:** Las definiciones y explicaciones que se muestren no deben ser complejas, ya que eso dificultaría la fase de desarrollo. Además, esto sería mucho más perjudicial, para un desarrollador con poca experiencia, ya que le costaría mucho más trabajo entenderlo.
- **Longitud:** La longitud de las explicaciones no debe ser extensa. Se tiene que tratar que el texto escrito sea lo más conciso posible, para no desmotivar al usuario.
- **Accesible por los desarrolladores:** La aplicación debe de estar disponible para cualquier usuario. No sería una buena estrategia pagar para usar el catálogo ni que se excluyera a usuarios por cerrar la implementación a ciertos tipos de dispositivos o sistemas operativos.
- **Función de buscador:** Se ve necesario, la implementación de una función que realice diferentes búsquedas para encontrar más fácil el texto.

3.3 Presupuesto

Este proyecto se ha realizado sin ningún coste. Un punto bastante positivo para un estudiante. Pero, a pesar de ello, se va a realizar una breve explicación de cuanto podría costar realizar el proyecto y que cosas se ofrecerían, en el caso de querer escalarlo.

Primero conviene destacar el coste que se debería tener para el hardware, pero en realidad, no es necesario nada especial, ya que las tecnologías utilizadas no demandan mucha potencia, por lo tanto, se abre un abanico de muchas posibilidades baratas, o incluso sin coste alguno si ya se tiene un ordenador con el que se pueda desarrollar.

En relación con el coste del software, como se ha comentado anteriormente, la aplicación se puede realizar sin ningún coste puesto que las tecnologías utilizadas son gratis. Pero, sí que es verdad que, en cuanto al despliegue, se puede realizar un despliegue que lleve un coste y que ofrezca mayores prestaciones.

Una de esas opciones es la de usar la herramienta de alojamiento web de la empresa Hostgator. Esta herramienta nos ofrece tres planes diferentes cuyos costes son 2.75 dólares, 3.50 dólares y 5.25 dólares al mes durante 1 año. Estos planes, ofrecen desde alojamiento para 1 sola web, hasta alojamiento con un número ilimitado de webs, herramientas gratis para el SEO, una dirección IP dedicada para la web o incluso, un dominio, entre otras cosas. Como podemos observar, estos precios son bastante asequibles.

Bluehost es otra empresa, la cual nos ofrece otra herramienta que podemos utilizar para nuestro alojamiento web. Con esta herramienta, nos encontramos precios similares e incluso llegando a ser un poco más caros. Estos precios son 2.88 euros, 5.33 euros, y 13.64 euros al mes durante 1 año. Como con Hostgator, estos planes van desde el alojamiento de 1 sola web con atención al cliente 24 horas y 10GB de almacenamiento SSD, hasta ilimitado alojamiento de webs, con atención al cliente 24 horas y 100GB de almacenamiento SSD. Además, en Bluehost destaca que, para cualquier plan, tenemos gratis el dominio durante 1 año, que a diferencia de Hostgator, solo veíamos esa característica en el plan más caro.

4 Diseño de la solución

En este capítulo se va a presentar el diseño de la solución a desarrollar para el catálogo de accesibilidad.

4.1 Diseño de la interfaz gráfica de usuario

Cuando se acordó este Trabajo Final de Grado, se diseñaron primeramente unos mockups, para hacer referencia a lo que iba a ser desarrollado.

El diseño del catálogo web constaba de una página de bienvenida y una página principal, que contenía toda la información. Lo podemos ver en la Figura 5.

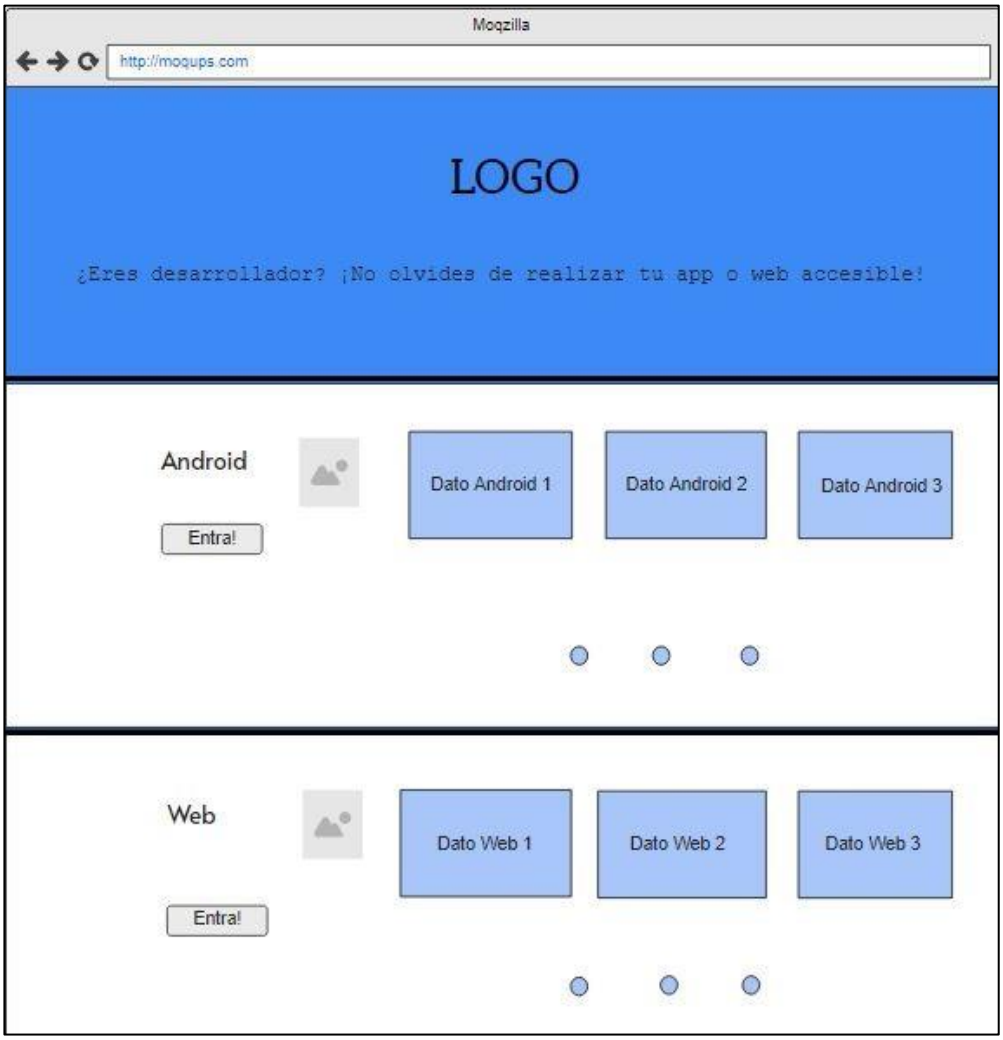


Figura 5 Página bienvenida del primer mockup

A priori, podríamos pensar que la página de bienvenida es una Landing Page, porque es una página realizada para atraer a los desarrolladores para que puedan hacer uso del catálogo. Pero no lo es, dado que una Landing Page siempre busca conseguir datos de los usuarios, como un correo electrónico o información personal. Esta página de bienvenida estaba hecha para que los usuarios tuvieran conocimiento sobre qué se pueden encontrar en la web y también lleva a la página principal.

Las Figuras 6 y 7 muestran los mockups de las páginas sobre accesibilidad en Android y accesibilidad Web, que son las páginas principales. Estos mockups como se puede apreciar en las figuras son bastante básicos. Se construyeron con el objetivo de tener una primera versión rápida de la interfaz gráfica a partir de la cual poder debatir e iterar sobre ella para llegar a la versión final de la interfaz. Además, como se puede observar en las Figuras 6 y 7, todo el contenido se plasmaba en una única página, y se decidió cambiar esa estructura.

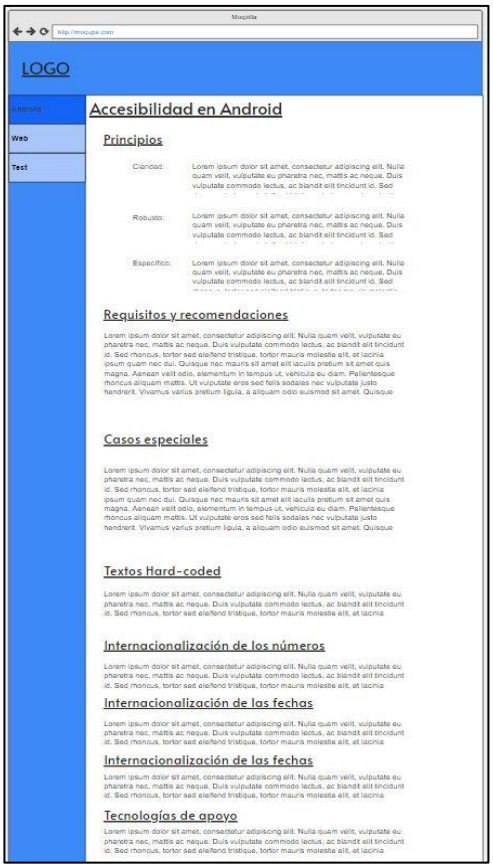


Figura 6 Página "Accesibilidad en Android" del primer mockup

Figura 7 Página "Accesibilidad en Web" del primer mockup

4.2 Arquitectura de la información

En este apartado vamos a explicar cómo se ha definido la arquitectura de la aplicación.

Toda la información sobre la accesibilidad (tanto para Android como para Web) se decidió sintetizarla en torno a los cuatro principios de la accesibilidad, que son:

- **Perceptible:** La información y los componentes de la interfaz de usuario deben presentarse a los usuarios de manera que puedan percibirlos.
- **Operable:** Los componentes de la interfaz de usuario y la navegación deben ser utilizables por todos los usuarios.
- **Entendible:** La información y las operaciones en la interfaz de usuario se tienen que entender.
- **Robusto:** El contenido se tiene que presentar de forma que pueda ser interpretado por una gran variedad de agentes de usuario, incluidas tecnologías de asistencia.

Dentro de cada principio, se han mostrado diferentes categorías, que nos ayudan a entender como cumplir cada principio. Toda la información plasmada dentro de estas categorías, e incluso las mismas categorías, se ha conseguido a partir de la lectura y el entendimiento de los recursos analizados en el capítulo 2.

Con el fin de que se entienda correctamente la información mostrada en el catálogo, se ha decidido añadir imágenes con código dentro de algunas categorías, que muestran como programar lo que se está explicando. Estas imágenes surgen a raíz de los recursos analizados en el capítulo 2 y por el entendimiento del autor. Cabe destacar que no todas las categorías disponen de imágenes, ya que no se les puede aplicar una solución por código a todas.

Luego, se decidió tratar también el tema de los criterios de conformidad. Estos criterios hacen referencia al nivel de accesibilidad que puede tener una página web. Los niveles de accesibilidad solo existen para las páginas web, ya que para Android no existe tal cosa. Estos niveles de accesibilidad se consiguen cumpliendo una serie de requisitos o *Criterios de éxito*. A pesar de que para Android no existan niveles de accesibilidad, se recomienda seguir estos criterios para una buena aplicación de la accesibilidad. Los niveles de accesibilidad que pueden tener una página web son:

- Nivel A: Es el primer nivel y el más bajo. Este nivel indica que se satisfacen los *Criterios de Éxito* del nivel A.

- Nivel AA: Es el segundo nivel. Este nivel indica que se satisfacen los *Criterios de Éxito* del nivel AA. Si se cumple este nivel significa que se cumple el anterior y este.
- Nivel AAA: Es el tercer nivel y el más alto. Este nivel indica que se satisfacen los *Criterios de Éxito* del nivel AAA. Si se cumple este nivel, quiere decir que se cumplen también los demás.

Al igual que para los principios comentados anteriormente, estos criterios se consiguieron a partir de las mismas fuentes del capítulo 2.

Por otra parte, también se decidió añadir en la página principal, información acerca de diferentes herramientas para evaluar la accesibilidad de nuestra aplicación Android o aplicación web.

La lectura y entendimiento de los recursos mencionados en el capítulo 2, han supuesto una gran inversión de tiempo, puesto que algunos han llegado a ser bastante extensos y densos, pero esto ha servido para obtener las explicaciones. Las imágenes han sido creadas por el autor, en base a algunos ejemplos que presentaban algunos recursos y a las explicaciones dadas por esos recursos.

5 Desarrollo de la solución

En esta sección se va a profundizar en cómo se ha desarrollado este proyecto. En primer lugar, se va a mencionar sobre la selección de herramientas que se ha realizado. En segundo lugar, se explicará cómo ha sido el desarrollo del catálogo y se expondrá cómo se ha plasmado toda la información recogida de los recursos, comentados en el capítulo 2. Por último, se hace referencia a cómo se ha llevado a cabo el despliegue.

5.1 Selección de herramientas y tecnologías

Primeramente, vamos a hablar sobre la tecnología de desarrollo web que se han analizado. Varias opciones son posibles en este punto, entre ellas, AngularJS, React, Vue.js o Javascript con HTML y CSS. Cabe destacar que ninguna tecnología es mejor, ni peor, ya que simplemente, cada una tiene sus ventajas y desventajas. De igual modo, todas estas tecnologías parten de la misma base, que es Javascript. Veamos las diferencias. Una de las opciones era no usar ningún framework y usar Javascript con HTML5 y CSS. Esta opción dejaba de lado muchas funcionalidades que ofrecen los frameworks actuales, y que harían más potente a nuestra aplicación. Además, como se explicó en el capítulo 1 en el apartado de Motivación, una de las motivaciones, era la de aprender, y la opción de aprender algún framework era mucho más llamativa.

A continuación, pasamos a comparar los tres frameworks nombrados anteriormente, que son AngularJS, Vue.js y React.

Para empezar, la curva de aprendizaje de cada uno es un aspecto muy importante para elegir una tecnología. Esta curva de aprendizaje no es ni más ni menos que una gráfica que describa que tan complicado es aprender algo, y como de pronto se pueden obtener beneficios y resultados. Por lo que respecta a los tres, AngularJS es el que más tiempo necesita para dominarlo. En cuanto a React y Vue.js, en estos apartados son muy similares, pero el framework creado por Facebook, Vue.js, presenta una ligera ventaja, porque su ecosistema es mucho más uniforme, es decir, las librerías de Vue.js son muy similares.

Seguidamente, vamos a comentar, que, en cuanto a la robustez, Angular y React se suelen utilizar mucho más que Vue.js para aplicaciones a gran escala. A pesar de que empresas de gran envergadura como Netflix o Facebook utilicen Vue.js, los

frameworks Angular y React son usados por la gran mayoría de empresas grandes como por ejemplo Uber, Google, Paypal o The New York Times. Además de eso, Angular es un framework que ha sido creado para albergar grandes proyectos. Por esto mismo se podría pensar que Angular tiene una ventaja aquí, y la tiene, pero en este caso, este proyecto no es un proyecto a gran escala.

Por la parte de la experiencia de desarrollo, es importante mencionar que las tres contienen características muy similares, y al final, las herramientas que usemos con el framework, las extensiones o los atajos de teclado, entre otros, suelen ser bastante subjetivos. Lo más importante es que nos encontremos cómodos al desarrollar.

En cuanto a la experiencia de mantener una aplicación, detectar bugs, anticiparse a los errores y solucionarlos, se aprecia una dificultad en AngularJS puesto que el uso del inyector de dependencias puede hacer que algo, sea difícil de entender cuando se trata de encontrar un bug. Por otro lado, Vue y React son mucho más expresivos, de esta manera el flujo de información es más claro y detectar un error es más fácil.

Sobre la característica del entorno de trabajo, es importante destacar que el tamaño del proyecto es un factor para tener en cuenta, ya que proyectos grandes quizás necesiten muchas herramientas. En este caso, al no ser grande, nos importa primar la ligereza y en este aspecto Vue y React están por delante de AngularJS.

Finalmente, como ya hemos comentado, todas las herramientas tienen sus ventajas y desventajas, por tanto, es vital saber escoger la que mejor convenga para cada proyecto.

Respecto al software de control de versiones, podemos comentar que un software de control de versiones es un sistema para llevar el control de los cambios realizados en un proyecto. Con solo la definición podemos ver que es un punto con importancia.

Existen tres principales softwares de control de versiones que son Git, Subversion y Mercurial. En una encuesta realizada este año 2022, por la plataforma líder de preguntas y respuestas por y para desarrolladores, Stack Overflow (Stack Overflow), se puede observar el colosal éxito de Git frente a los demás. De hecho, resalta el tan bajo uso de Mercurial. Por eso mismo, de este último no se ha llevado a cabo su estudio. En la Figura 6, podemos observar el resultado de la encuesta.

Figura 6 Resultados encuesta Stack Overflow

Sin embargo, se va a realizar la comparación entre Git y Subversion, porque a pesar de la diferencia de uso marcada por la encuesta, resulta interesante ver que ofrece cada uno.

Git es un SCVD (sistema de control de versiones distribuido) y Subversion es un SCVC (sistema de control de versiones centralizado). La principal diferencia entre los dos tipos es que, en el centralizado solo existe una copia del proyecto y un solo lugar en el cual todas las versiones de un proyecto se encuentran guardadas. Por otro lado, en el distribuido trabaja con que una copia del proyecto original es guardada en cada máquina local y los cambios deben ser subidos al repositorio online y bajados desde el repositorio online para actualizar la versión que cada desarrollador tiene.

El primer punto para comentar es que el acceso a la historia de cada repositorio es más veloz en Git, ya que cada usuario tiene una copia completa guardada del proyecto.

Además, dado que en Git cada usuario almacena su propia copia, se puede usar sin conexión. Y aparte de esto, esta característica también afecta a que si alguna de las copias queda corrupta solo los cambios que hayan sido únicos para ese repositorio serán perdidos, pudiendo así recuperar en gran parte el proyecto. En Subversión, al contrario, si el repositorio central queda corrupto o se pierde, habría que recuperarlo mediante una copia de seguridad previamente hecha.

Pero justamente esto, se puede convertir en una ventaja para Subversion, puesto que si se necesita una copia de seguridad solo hay un sitio donde ir a buscarla.

Una de las características más llamativas de Git es la gestión de las ramas. Con las ramas podemos crear diferentes aspectos del proyecto y después fusionarlos con el repositorio principal, de esta manera podemos ir desarrollando paralelamente diferentes funcionalidades sin necesidad de tocar el repositorio principal, así, nos evitamos el

riesgo que siempre existe si se trabaja solamente con el principal. En Git resulta sencillo este proceso, pero en Subversion no, ya que se basan en crear directorios.

Respecto al control de acceso, en Git es el dueño del repositorio quién controla quien puede participar en el proyecto y qué cambios aplicar. Lo que no se hace es controlar que los usuarios puedan realizar ciertas funciones, no así en Subversion, ya que en este último sí que se gestiona el acceso y se puede denegar que el usuario pueda hacer commits. En otras palabras, en Git el usuario puede tener control total de su proyecto, pero en Subversion, está controlado por el usuario del repositorio principal.

Los sistemas de control de versiones se basan en la regla de que la mayoría de los archivos serán fusionables. El problema es que esto no se puede aplicar a los archivos binarios. Por eso Subversion implementa un modelo llamado Bloquear-Modificar-Desbloquear (Lock-Modify-Unlock) y por tanto convierte mucho más fácil la tarea de fusionar archivos binarios. En contraste con Git, éste no admite bloqueos de archivos exclusivos.

Por último, los comandos de cada uno son bastantes similares, para realizar tareas comunes.

El proyecto se tiene que alojar sobre una plataforma, para que cualquiera pueda hacer uso de él. Este paso no es el más importante, pero sí que requiere especial atención, porque la plataforma elegida nos brindará diferentes características. Asimismo, es un paso que no podemos obviar, puesto que es una muy buena práctica no tener tu proyecto alojado en tu máquina y tenerlo en una plataforma online, sin importar si se trabaja con un equipo o solo. Con ello conseguiremos múltiples ventajas, entre las que destacan mantener el código, hacer seguimiento de los errores, compartir el código con diferentes desarrolladores para ayudarse mutuamente y tener una copia de seguridad. Muchas opciones se presentan, como por ejemplo Bitbucket, GitLab o GitHub. Veamos que nos ofrece cada una.

- Bitbucket es una plataforma muy popular para subir proyectos basados en código abierto. Permite tener repositorios públicos y privados y se puede intercambiar archivos entre los usuarios de esta plataforma. Además, Bitbucket nos ofrece tener alojar nuestra página web e incluso formar equipos dentro de la plataforma de hasta 5 usuarios.
- GitLab también es una plataforma bastante popular, sobre todo después de la compra por parte de Microsoft a Github. Ésta nos ofrece ciertas características similares a la anterior comentada, pero también nos

ofrece la posibilidad de que sus usuarios puedan instalar la plataforma en sus servidores particulares, lo cual no solo permite que un usuario pueda asociar un dominio personal a su cuenta de GitLab, sino que también puede asociar un servidor web propio, lo cual permitirá le permitirá gozar de mayor seguridad y privacidad en los contenidos.

- GitLab suele ser una plataforma escogida para buscar mayor privacidad, seguridad y velocidad en sus cuentas, aunque, por el contrario, este tipo de acciones deberemos de configurarlas previamente y no suelen ser sencillas para usuarios participantes.
- GitHub es la plataforma de alojamiento de código más usada en el mundo, la cual nos permite tener nuestros propios repositorios públicos y privados. Además, nos permite al igual que en las anteriores, tener alojada nuestra página web gratuita e incluso redirigir nuestra página web a un dominio propio. También nos permite compartir nuestros repositorios entre diferentes usuarios de la plataforma, para así poder trabajar en equipo.

En cuanto al despliegue, este catálogo es una aplicación web de Frontend, por eso mismo se puede prescindir de herramientas de despliegue como Amazon Web Services (AWS), Heroku o Microsoft Azure, entre otras. Estas herramientas nombradas suelen estar dirigidas al despliegue de Backend, puesto que ofrecen servicios para poder desplegar APIs, bases de datos, etc, es decir, todo lo que corresponda a la parte del servidor. Entonces lo que nos interesa es poder desplegar nuestra aplicación, y que no se muestre una página en blanco o en formato HTML, sino que se pueda renderizar todo el HTML y el CSS y que se pueda navegar por ella, es decir, un despliegue de Front-End.

Se pueden encontrar diferentes herramientas para el despliegue de aplicaciones Frontend. Algunas de las que nos podemos encontrar son GitLab Pages, Netlify, Bitbucket Cloud… pero no se ha preferido estudiar sobre ellas, a diferencia de lo ya comentado, ya que como se explicará más adelante, la plataforma donde se aloja código es GitHub, y ésta ofrece su propio servicio para desplegar aplicaciones Front-End, llamada GitHub Pages.

5.2 Proceso de desarrollo

Una vez realizada la elección de las tecnologías y herramientas, comienza la fase de diseño. En ella se corrigieron los problemas que se vieron en el apartado *4.1 Diseño de la interfaz gráfica de usuario* y se llevó a cabo un nuevo diseño. Una vez realizado este diseño podemos empezar la fase de desarrollo, donde se entra ya en contacto con Vue.js y con Git y donde se puede ver cómo va creciendo el catálogo.

La aplicación se fue creando poco a poco, pero siempre pensando en el modelo MVP (producto mínimo viable). Este modelo se caracteriza por fragmentar el proyecto entero en pequeños entregables. Cuando el proyecto llegó a un punto donde ya era funcional, pero tampoco estaba terminado completamente, se procedió a desplegar la aplicación, así se podría conseguir evitar errores futuros.

5.3 Desarrollo del catálogo

En la página de bienvenida podemos observar una presentación de la aplicación y de lo que un usuario se puede encontrar. Por supuesto, está página contiene un botón que lleva a la página principal. Esto lo vemos en la Figura 7.

Figura 7 Página Bienvenida

La página principal es la que se encarga de brindar toda la información. Nos encontramos con una barra lateral de navegación que con ella podremos navegar por toda la aplicación. Está dividida en cuatro categorías. La primera de ellas se llama “Principios”. Esta categoría es la parte más importante de nuestra aplicación web, ya que es la que nos explica toda la información. Se ha decidido que este proyecto se basara en estos principios, ya que es necesario entenderlos y saber cómo alcanzar cada uno, para poder aplicar correctamente la accesibilidad. En la Figura 8 vemos un ejemplo.

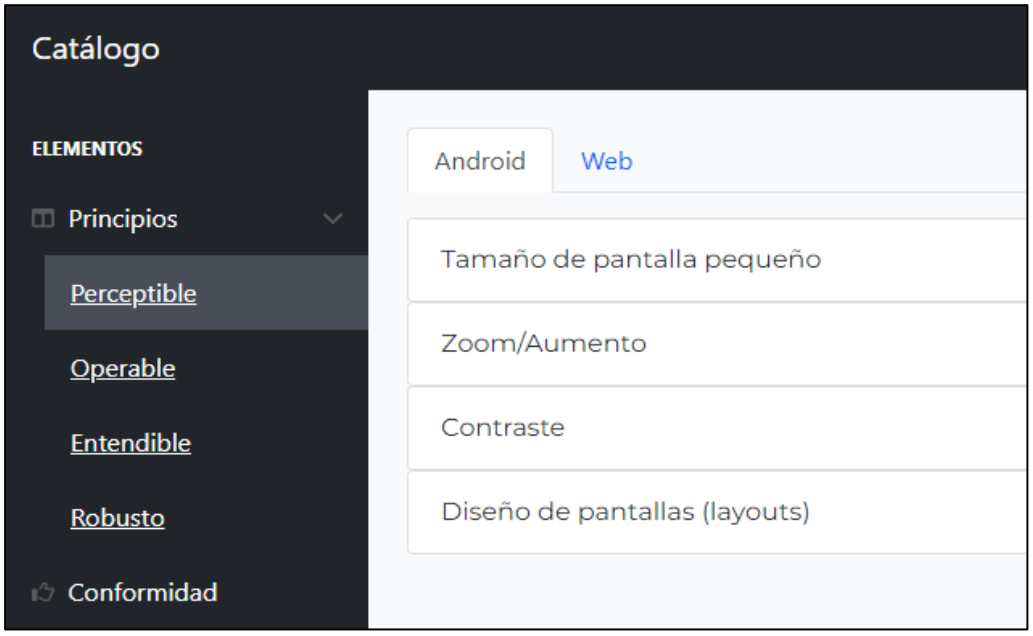


Figura 8 Apartado Principios

Al acceder a cada uno de ellos, podemos ver cómo mejorar cada aspecto de nuestra aplicación Android o web según el principio. Dentro de “Principios” nos podemos encontrar dos pestañas, haciendo referencia cada una a Android y Web y, como es obvio, cada una representará la información sobre accesibilidad en Android y en contenido web. En la Figura 9 lo podemos ver.

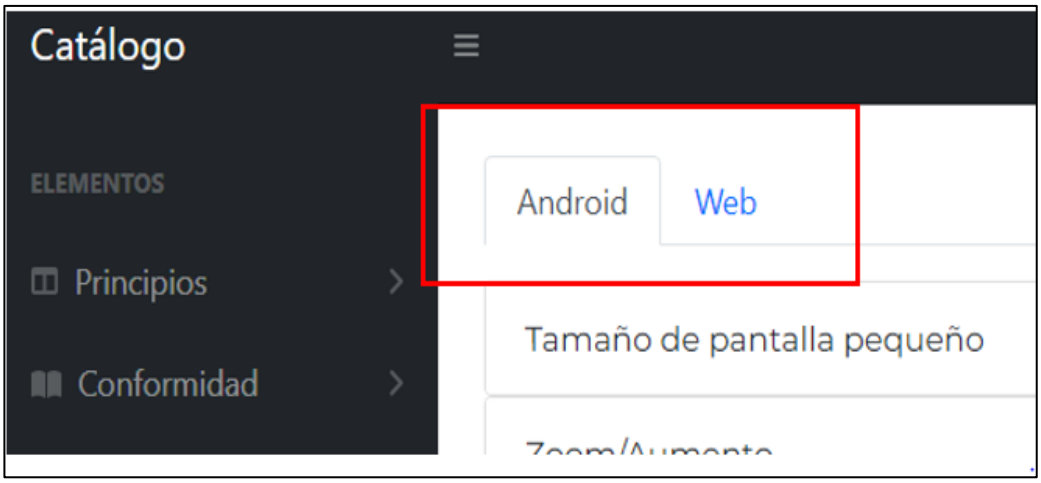


Figura 9 Pestañas Android y web

Para la visualización del contenido del apartado *Principios* se ha decidido utilizar, lo que se conoce en Vue como Mixin. En la Figura 10 podemos ver un ejemplo de como se muestra la información.

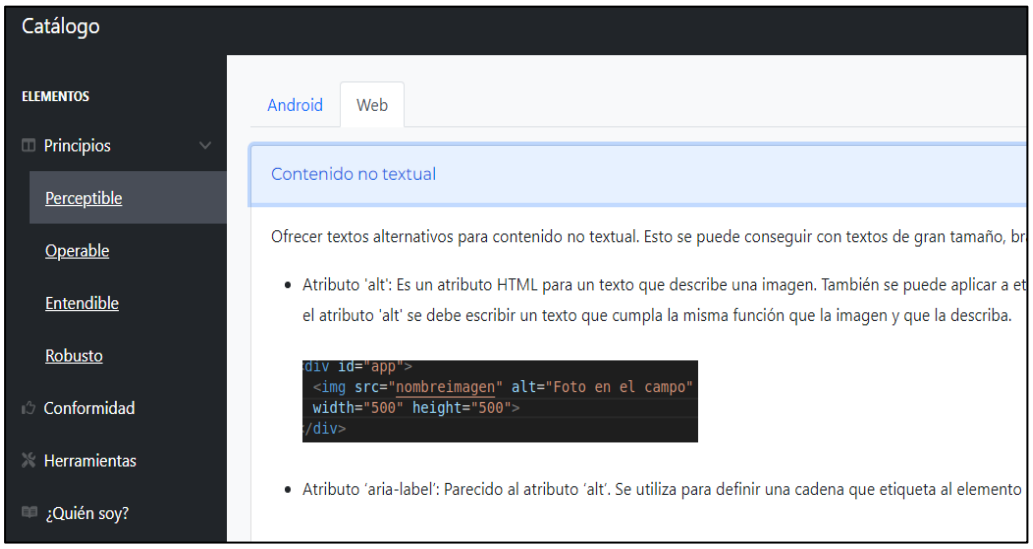


Figura 10 Imagen Principios desplegado

Un Mixin es una propiedad de Vue que podemos usar para cuando tenemos dos o más componentes que poseen una funcionalidad bastante parecida, pero no se pueden fusionar en uno solo porque resultaría demasiado complejo. En nuestro caso, el buscador no es ningún componente, pero, entonces ¿por qué se usa?

Antes de contestar, es necesario aclarar que existen otras formas, pero de esta manera resulta bastante conveniente, ya que en el fichero del mixin, será donde esté

todo el texto a mostrar en la pantalla principal. Todas las definiciones y explicaciones de cada principio estarán ahí.

Respondiendo a la pregunta, hemos aprovechado la funcionalidad de que el mixin pueda ser leído desde diferentes ficheros, para conseguir más comodidad al leer el código y, un mejor mantenimiento, porque la página principal no estará sobrecargada y la función que realiza la búsqueda será más limpia.

La segunda categoría de este menú lateral es el de “Conformidad”. En ella podemos encontrar todo lo relativo a los niveles de conformidad de la accesibilidad. También está dividido por pestañas y en este caso son 3, una por cada nivel. Dentro de cada pestaña, nos podemos encontrar una columna llamada “Requisito”, que corresponde a los requisitos que debe tener para llegar a ese nivel, y otra columna “Explicación”, que nos detalla como cumplir ese requisito. Lo podemos ver en la Figura 11.

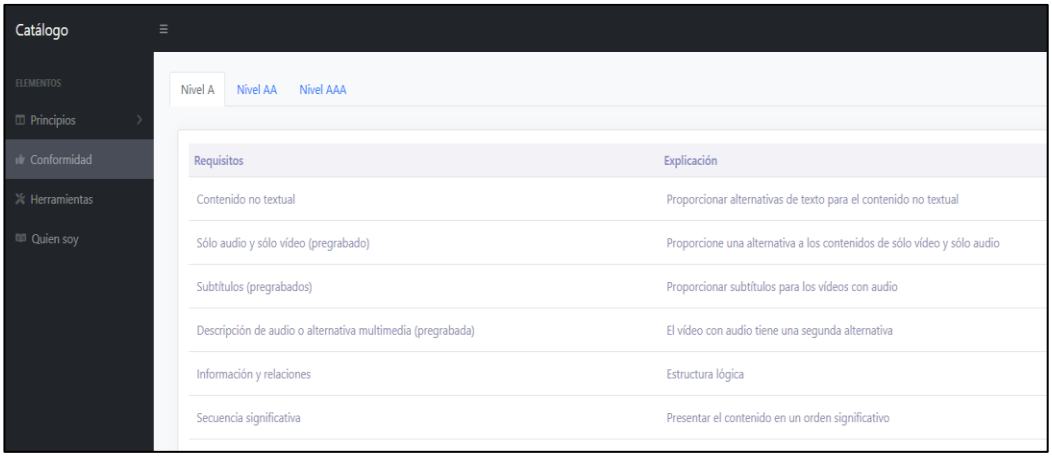


Figura 11 Vista Conformidad

Una manera de navegar por la web sería la de buscar el requisito que se intenta cumplir, y después buscarlo en el apartado de principios, mediante el buscador.

El buscador es la caja de texto que nos encontramos en la parte derecha de la página. Este buscador se encarga de encontrar más fácilmente la información del apartado *Principios.* En la Figura 12 se muestra un ejemplo de una búsqueda realizada con la palabra *móviles*.

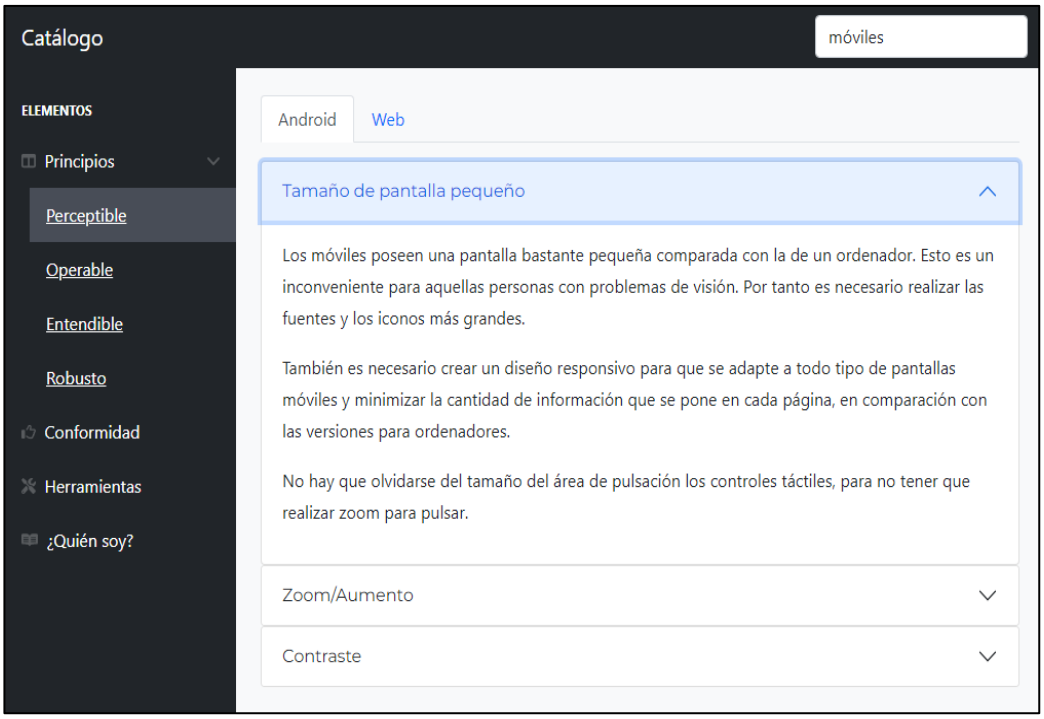


Figura 12 Imagen Buscador

La tercera categoría del menú trata sobre diferentes herramientas que nos podemos encontrar para valorar la accesibilidad tanto en nuestro dispositivo Android como en nuestra página web. Lo podemos ver en la Figura 13.

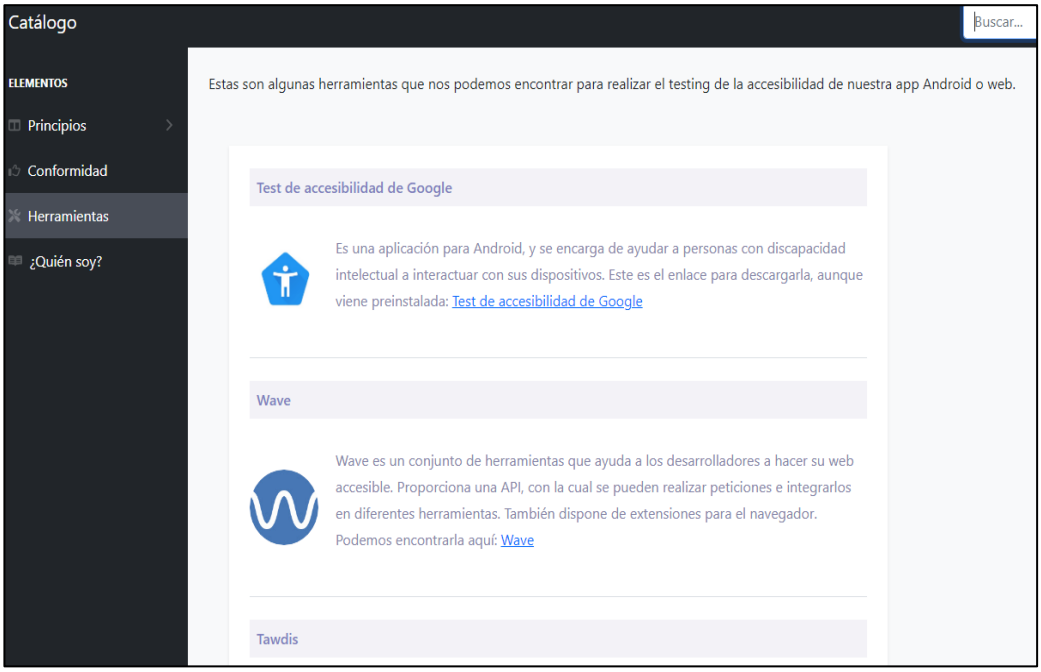


Figura 13 Vista Herramientas

En la cuarta categoría, podemos ver un apartado que hace una breve introducción al autor, e indica los enlaces al repositorio de Github donde está alojada la aplicación. También se indica el enlace al repositorio Gist, donde está alojado el código que se muestra en las imágenes del apartado “Principios”. Lo podemos ver en la Figura 14.

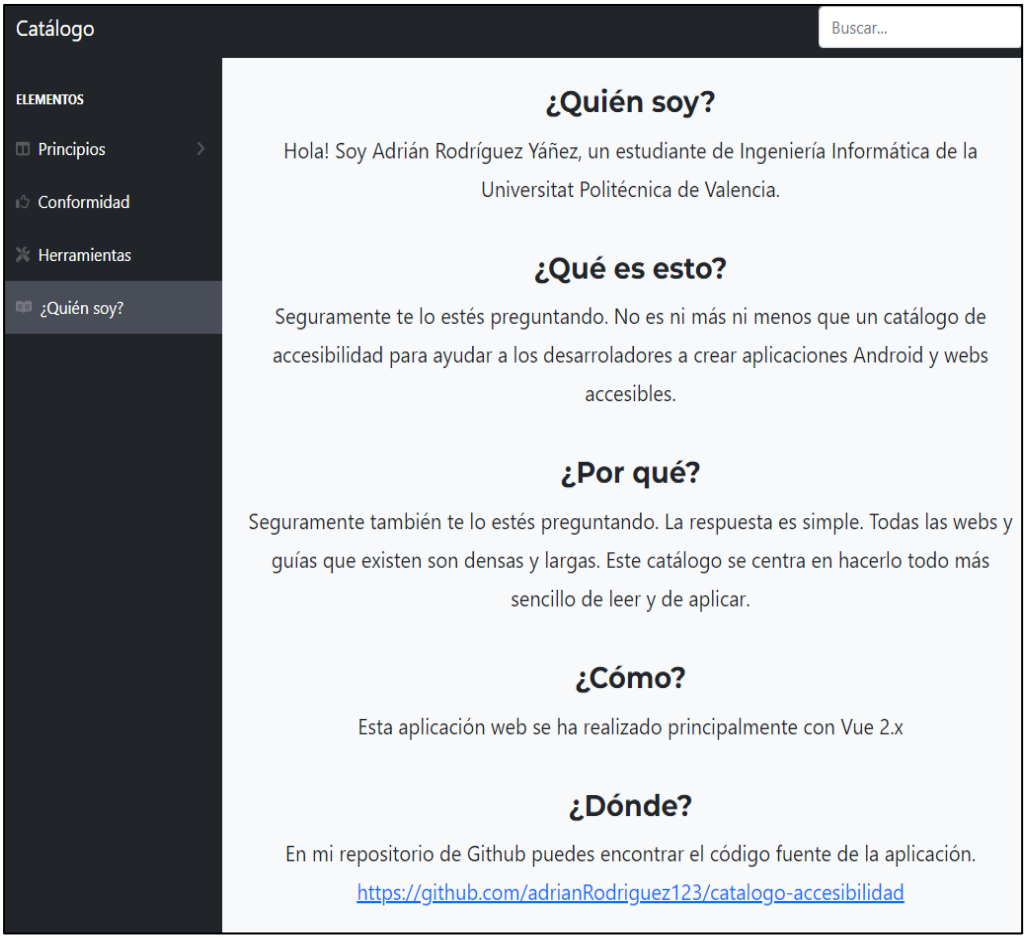


Figura 14 Vista ¿Quién soy?

Cabe destacar que este menú lateral sigue un patrón de diseño llamado *Menú de acordeón* en el apartado de *Principios*, que nos beneficia en buscar rápidamente cada sección de nuestra aplicación. Además, también nos ayudar a tener un menú mucho más entendible y corto. En la Figura 8 y en la Figura 10 podemos ver el menú desplegado.

A parte del anterior patrón de diseño comentado, también se ha visto conveniente usar el patrón llamado *Home Link*, donde básicamente consiste en un elemento, normalmente un icono o el logo, que enlaza a la página de bienvenida o

“home”. Este patrón lo representamos gracias al título de la esquina superior izquierda. En la Figura 15 se puede ver.

Figura 15 Título Home Link

Como se ha comentado en el capítulo anterior, al ser este proyecto una aplicación Front-End, no dispone de base de datos ni de ninguna API. Tampoco existe Back-End, por tanto, no existe diseño lógico UML.

Vue.js es un framework que sigue una arquitectura basada en componentes. Esto quiere decir que toda la aplicación se puede dividir en pequeños componentes. Esta característica resulta beneficiosa para facilitar el desarrollo y evitar la sobrecarga en las páginas. Además de eso, los componentes en Vue.js tienen la principal función de ser reusables en cualquier página y también posee la particularidad de que usa una representación virtual del DOM de una página web. Esto puede resultar muy útil porque se puede renderizar la página sin necesidad de modificar y refrescar, el árbol (DOM) entero cada vez. Para el desarrollo de la página principal, se diseñó un componente de Vue.js para que mostrara todo el texto en pantalla.

5.4 Implantación

Para llevar a cabo el despliegue de esta aplicación web, como ya se ha ido comentando se ha usado GitHub Pages. En GitHub Pages es necesario que exista un directorio llamado *dist*, pues éste será el que se encargue de contener todos los recursos necesarios para desplegar la web.

Como nos podemos imaginar, *dist* es una abreviatura de *distributable*. Los ficheros que almacene este directorio serán los que se muestran en la web. Este directorio lo creamos con el comando:

```
uadrian@tfg:~/git/miproyecto$ npm run build
```

Cuando se crea este directorio, tenemos que inicializarlo con Git para poder subirlo al repositorio con todos los ficheros del proyecto, sin olvidarnos de crear una rama llamada *gh-pages* para subirlo desde ahí. Una vez subido este directorio, en este caso desde GitHub, habrá que gestionar cual es la rama que queremos llevar a desplegar. Todos estos pasos los podemos y debemos automatizarlos mediante un fichero.

Es importante matizar que no es necesario poner los nombres como se indica en la explicación, como puede ser el caso con *dist* o *gh-pages*, pero es altamente recomendado, puesto que son los que la comunidad suele usar y nos ayudarán a obtener un fácil y cómodo despliegue. En la Figura 16 podemos ver el fichero de despliegue.

```
#!/usr/bin/env sh
# abort on errors
set -e
# build
npm run build
# navigate into the build output directory
cd dist
# if you are deploying to a custom domain
# echo 'www.example.com' > CNAME
git init
git add -A
git commit -m 'despliegue'
git push -f git@github.com:adrianRodriguez123/catalogo-accesibilidad.git master:gh-pages
cd -
```

Figura 16 Fichero de despliegue

6 Pruebas

En este capítulo se comentarán las pruebas realizadas sobre la aplicación. En concreto, se ha realizado la medición de la usabilidad utilizando la Escala de Usabilidad de Sistemas (UIFromMars).

El Sistema de Escalas de Usabilidad, por sus siglas SUS (System Usabilty Scale), es un método para evaluar la usabilidad de cualquier sistema. Este método nos permite varias cosas. Primero de todo, nos evalúa la eficacia de nuestra aplicación para saber si los usuarios pueden alcanzar sus objetivos. Después, podemos evaluar la eficiencia, es decir, cuanto esfuerzo es necesario para que un usuario pueda alcanzar esos objetivos. Por último, pero no menos importante, podemos saber la satisfacción del usuario con el producto.

Este sistema consta de 10 preguntas predefinidas, donde el usuario tendrá que contestar según la escala de Likert. Esta escala se mide simplemente con un número de respuestas disponibles, en nuestro caso 5, donde el usuario tendrá que escoger según su grado de conformidad. Las respuestas varían del 1 al 5, siendo posibles las siguientes respuestas:

1. Totalmente en desacuerdo
2. En desacuerdo
3. Neutro
4. De acuerdo
5. Totalmente de acuerdo

Las preguntas que se han realizado son las siguientes:

1. Creo que me gustaría utilizar esta aplicación con frecuencia.
2. Encontré la aplicación innecesariamente compleja.
3. Pensé que la aplicación era fácil de usar.
4. Creo que necesitaría el apoyo de un técnico para poder utilizar esta aplicación.
5. Encontré que las diversas funciones de esta aplicación estaban bien integradas.
6. Pensé que había demasiada inconsistencia en esta aplicación.
7. Me imagino que la mayoría de la gente aprendería a utilizar esta aplicación muy rápidamente.
8. Encontré la aplicación muy complicada de usar.

9. Me sentí muy seguro usando la aplicación.
10. Necesitaba aprender muchas cosas antes de empezar con esta aplicación.

Una vez recibidas las puntuaciones de los usuarios, se procede a calcular la puntuación de cada usuario mediante una fórmula, la cual es la siguiente:

- La suma de las puntuaciones de los enunciados impares restando 5.
- La suma de las puntuaciones de los enunciados pares restando 25.
- Sumar ambos resultados y multiplicar por 2.5.

Para calcular la puntuación final, solamente habrá que realizar la media en todas las puntuaciones de los usuarios. Es conveniente destacar, que cualquier puntuación por debajo de 68 se considera que existen bastante cosas para corregir, y por debajo de 50 la aplicación tiene demasiados errores. Lo podemos ver en la Figura 17. El resultado será sobre 100, pero este resultado no es un porcentaje.

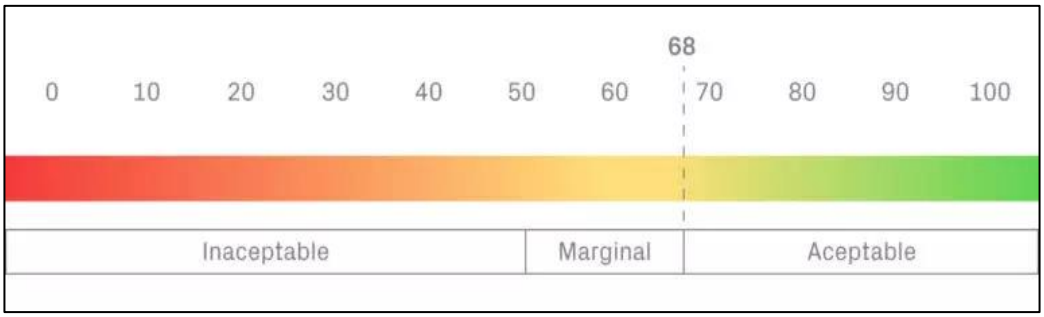


Figura 17 Como se valora una aplicación según su puntuación

Se ha realizado el cuestionario a 10 usuarios que se dedican al desarrollo de aplicaciones. Los usuarios han visitado la aplicación web y han estado navegando a través de ella. Solo se les explicó lo que se iban a encontrar una aplicación con información sobre accesibilidad. En la Figura 18 podemos ver el resultado de cada usuario por cada pregunta.

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4	Pregunta 5	Pregunta 6	Pregunta 7	Pregunta 8	Pregunta 9	Pregunta 10
Usuario 1	3	2	3	2	4	2	3	2	3	3
Usuario 2	3	1	5	1	4	2	5	1	5	1
Usuario 3	3	1	4	2	5	1	4	1	3	1
Usuario 4	2	2	4	2	4	1	4	1	3	2
Usuario 5	4	2	3	2	3	2	4	2	4	1
Usuario 6	5	3	2	1	4	2	3	1	3	1
Usuario 7	3	1	5	1	5	1	5	1	5	1
Usuario 8	2	3	5	2	5	1	3	2	5	2
Usuario 9	3	2	4	3	3	2	4	1	4	1
Usuario 10	4	1	3	2	4	1	4	2	4	1

Figura 18 Resultados de los cuestionarios

En la Figura 19 podemos ver el resultado final de cada usuario. Para calcular el resultado final total, solo habría que realizar la media de los resultados finales de cada usuario. Como se puede observar, este resultado final tiene un valor de 77'5, por lo tanto, podemos afirmar que la aplicación está en un buen rango y no tiene grandes aspectos a corregir.

Usuario 1	62,5
Usuario 2	90
Usuario 3	82,5
Usuario 4	72,5
Usuario 5	72,5
Usuario 6	72,5
Usuario 7	95
Usuario 8	75
Usuario 9	72,5
Usuario 10	80

Figura 19 Resultados finales de cada usuario

7 Conclusiones

Tras todo lo ya comentado, vamos a realizar una valoración para comprobar si los objetivos, expuestos en el capítulo 1, han sido alcanzados.

Por un lado, hemos observado que en este trabajo se ha abordado el desarrollo de una aplicación Front-End, la cual servirá de ayuda para aquellos que vayan a programar una aplicación web o Android.

Podemos afirmar que, las herramientas que existen hoy en día para que los desarrolladores se informen acerca de la accesibilidad pueden ser extensas, con pocos ejemplos prácticos, o incluso pueden estar basadas en sitios web no oficiales.

Este catálogo web, sí que está basado en sitios reconocidos y oficiales, y se ha pretendido que las explicaciones y definiciones dadas hayan sido simples y sin dar rodeos. Además, se muestran imágenes de ejemplos de código, que resultan muy útiles para el desarrollo y el código de esas imágenes se puede consultar en el enlace indicado en la web.

Por otro lado, se ha conseguido diseñar, desarrollar e implementar una aplicación web desde cero, sin conocimientos de las tecnologías utilizadas. Del mismo modo, también se ha aprendido a desplegar una aplicación Front-End. Y aparte de eso, también he conseguido nociones en las tecnologías que han sido analizadas, pero no escogidas.

Todos los conocimientos comentados a lo largo de este Trabajo de Fin de Grado los he adquirido gracias a la motivación de querer ir más allá, y no quedarme con los que ya tenía.

A nivel personal, considero que he crecido en cuanto a poder programar una aplicación desde cero. Desde la fase de elección de la tecnología, pasando por su aprendizaje, hasta la del despliegue. No obstante, creo que lo que más ha supuesto una dificultad ha sido la búsqueda y la comprensión de la información, puesto que resulta complicado compaginar el trabajo con lecturas muy largas de información que muchas veces no era provechosa.

7.1 Relación con las asignaturas del Grado

Este Trabajo Final de Grado se ha visto relacionado con varias asignaturas que se han cursado a lo largo de la carrera.

Cabe mencionar, que este proyecto se ha visto afectado en mayor o menor medida por todas las asignaturas de la rama de Ingeniería de Software, pero especialmente vamos a comentar dos.

En primer lugar, mencionamos la asignatura Diseño de Software, porque en ella se estudiaron por qué y cómo aplicar los patrones de diseño al mundo del Software, y como hemos visto en el capítulo 4, se han aplicado algunos patrones.

En segundo lugar, destacamos la asignatura Calidad del Software, porque como vemos en el capítulo 7, hemos analizado la usabilidad de la aplicación, y éste era uno de los puntos importantes de esta asignatura.

7.2 Trabajo futuro

Finalizado este trabajo, podemos concluir que el catálogo ha cumplido los objetivos. No obstante, se presentan a continuación nuevas funcionalidades con el fin de mejorar la herramienta, ya que por cuestiones de tiempo disponible no han podido llegar:

- **Idiomas:** Sería interesante añadir varios idiomas en la página, principalmente inglés, para ampliar el número de usuarios que puedan utilizar la herramienta.
- **Realizar tu propio test:** Conviene tener un apartado, donde puedas introducir una URL y puedas realizar un test de la accesibilidad de tu página web. Cabe mencionar, que, para Android, esta opción no sería interesante puesto que habría que desarrollar una aplicación móvil.
- **Machine Learning:** Esta rama de la inteligencia artificial, bastante novedosa en los últimos años, podría aportarnos muchas mejoras impactantes. Por ejemplo, en el buscador que se ha implementado, estaría bien ir un poco más adelante y que mostrara imágenes previas sobre la búsqueda, e incluso una página aparte que presente una lista con el número de búsquedas y con las búsquedas encontradas.

7.3 Agradecimientos

Por último, me gustaría dedicar un apartado para agradecer a aquellas personas que me han apoyado durante todos estos años.

En primer lugar, me gustaría agradecerle a mi familia, ya que en las buenas o en las malas siempre han estado. Ya sea por independizarme, o por dejar un trabajo o por cualquier otro motivo, siempre han estado ahí.

En segundo lugar, a aquellos amigos que han celebrado conmigo mis victorias y me han consolado en mis derrotas.

Y, por último, pero no menos importante, quería agradecérmelo a mí mismo, por creer en mí y por no rendirme nunca.

8 Bibliografía

AngularJS. Angular Página oficial. *Angular Página oficial.* [En línea] https://angular.io/.

Carreras, Olga. 2018. *Olga carreras Blog.* [En línea] 20 de 09 de 2018. https://olgacarreras.blogspot.com/2018/09/real-decreto-11122018-sobre.html#:~:text=El%20Real%20Decreto%201112%2F2018,financiaci%C3%B3n%20p%C3%BAblica%2C%20a%20ser%20accesibles.

Github. Github Pages. *Github Pages.* [En línea] https://pages.github.com/.

Massachusetts Institute of Technology (MIT). *Making Applications Accessible.* [En línea] https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/ui/accessibility/apps.html.

Material Design. Web Oficial Material Design. [En línea] Google. https://material.io/design/usability/accessibility.html#understanding-accessibility.

Observatorio estatal de la discapacidad. 2022. [En línea] 19 de 04 de 2022. https://www.observatoriodeladiscapacidad.info/un-total-de-438-millones-de-personas-949-de-cada-mil-habitantesafirmaron-tener-algun-tipo-de-discapacidad/#:~:text=Abr%2C2022-,Un%20total%20de%204%2C38%20millones%20de%20personas%20(94%2C,tener%20alg%C3%BAn%20.

Stack Overflow. Stack Overflow. *Stack Overflow.* [En línea] https://survey.stackoverflow.co/2022/#section-version-control-version-control-systems.

Start Bootstrap. Agency. [En línea] https://startbootstrap.com/theme/agency.

Start Bootstrap Agency. Start Bootstrap Agency Theme. *Start Bootstrap.* [En línea] https://startbootstrap.com/.

Start Bootstrap Sb-admin. Start Bootstrap. *Start Bootstrap Sb-admin Theme.* [En línea] https://startbootstrap.com/template/sb-admin.

UIFromMars. UIFromMars. *UIFromMars SUS.* [En línea] https://www.uifrommars.com/como-medir-usabilidad-que-es-sus/.

World Wide Web (WCAG QuickRef). 2018. *How to Meet WCAG.* [En línea] 05 de 06 de 2018. https://www.w3.org/WAI/WCAG21/quickref/.

World Wide Web Consortium. 2018. [En línea] 05 de 06 de 2018. https://www.w3.org/TR/WCAG21/.

World Wide Web Consortium (Mobile Accesibility). *Mobile Accesibility at W3C.* [En línea] https://www.w3.org/WAI/standards-guidelines/mobile/.

World Wide Web Consortium (WCAG Overview). *WCAG 2 Overview.* [En línea] https://www.w3.org/WAI/standards-guidelines/wcag/.

ANEXO

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**		X		
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**				X
ODS 9. **Industria, innovación e infraestructuras.**				X
ODS 10. **Reducción de las desigualdades**.		X		
ODS 11. **Ciudades y comunidades sostenibles**.				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Los objetivos que se pueden relacionar con este Trabajo Final de Grado son principalmente el de Educación de Calidad, puesto que la herramienta desarrollada puede ayudar a enseñar a programar aplicaciones accesibles y el de Reducción de las Desigualdades, ya que las herramientas desarrolladas pueden ser de utilidad para aquellas personas que normalmente presentan dificultades.

El objetivo de Educación de Calidad consiste “Garantizar una educación inclusiva, equitativa y de calidad y promover oportunidades de aprendizaje durante toda la vida para todos”. El catálogo web implementado nos puede ayudar, en lo relativo a poder desarrollar aplicaciones accesibles, y así, todo el mundo poder disfrutar de ellas. Estas aplicaciones, pueden ser de cualquier tipo, pero, nos podemos fijar especialmente en las que son de contenido educativo, o incluso, que pertenecen a algún órgano del sector público.

En el año 2018, se estableció el Real Decreto 1112/2018, donde se obligaba a todos los sitios web y apps nativas de la Administración Pública, o que reciban financiación pública, a ser accesibles. Ello, hace que aumente la importancia de realizar la accesibilidad. Imaginemos que el Ministerio de Educación desarrolla una aplicación para enseñar a niños a leer. Dicha aplicación tendría que ser accesible para todos los niños, para que todos pudieran aprender a leer.

En cuanto a “Reducción de las desigualdades”, y con mucha relación al anterior, la aplicación desarrollada en este Trabajo Final de Grado hace posible que al haber más personas utilizando la aplicación, todos puedan llegar a ser iguales. Apoyándonos en el ejemplo anterior sobre la aplicación para enseñar a leer, crearía bastante desigualdad, que no todos los niños tuvieran las mismas oportunidades para aprender a leer.

El ODS “Reducción de las desigualdades”, hace referencia a “Reducir la desigualdad en y entre los países”. Por lo tanto, podemos observar que cuantas más personas puedan utilizar nuestra aplicación, mayor será el beneficio.

Esta aplicación web se basa en el código abierto. Eso quiere decir que cualquiera puede desarrollar y ayudar al crecimiento de la web. Como se suele decir, “dos mentes piensan mejor que una”, y siempre se encontrarán más ideas y errores para mejorar que en vez de trabajar uno solo.

El objetivo final de esta aplicación siempre ha sido el de poder ofrecer ayuda a los desarrolladores para que puedan realizar aplicaciones que puedan ser disfrutadas por todos. Con aplicaciones accesibles, las personas podrán, al menos, no sentirse excluidas por no saber cómo utilizarla o por no poder hacer uso de ella.

En una situación como la de hoy en día, donde nos vemos afectados por el COVID-19 y por más inconvenientes que afectan a la desigualdad, es importante que una persona tenga las mismas posibilidades que otra, sin importar donde haya nacido, cuánto porcentaje de visión tiene, cuánto porcentaje de audición tiene, si tiene algún problema motriz, etc… es decir, sin importar si tiene alguna discapacidad. Es por ello que la principal ventaja de esta aplicación es la de poder aportar al mundo, aplicaciones que todo el mundo pueda utilizar.