UNIVERSITAT POLITÈCNICA DE VALÈNCIA

CAMPUS D'ALCOI

Navegador web con separación de sesiones

MEMORIA PRESENTADA POR:

Alejandro Jiménez Marqués

GRADO DE *INGENIERIA INFORMÁTICA*

Convocatoria de defensa: *09/2020*

Resumen

Actualmente los navegadores web comparten sesión, por lo que no es posible iniciar sesión en varias cuentas de forma simultánea sin hacer uso de más de un navegador. La aplicación que se ha creado permite el inicio de varias sesiones de forma simultánea en la misma web evitando el uso de varios navegadores y el tedioso proceso que esto conlleva.

Palabras clave: navegador, web, sesión, cookies, cache

Abstract

Currently, web browsers share session, so it isn’t possible to log in into multiple accounts simultaneously without using more than one web browser. The application that has been created allows several sessions to be started simultaneously on the same website, avoiding the use of more than one browser and the tedious process that this entails.

Keywords : browser, web, session, cookies, cache

Tabla de contenidos

1. Introducción 10
   1.1. Motivación 10
   1.2. Estudio de Mercado 11
      1.2.1. Plugins y extensiones 12
      1.2.2. Navegadores web 13
      1.2.3. Otras aplicaciones 14
   1.3. Objetivos del Proyecto 14
   1.4. Estructura del resto de la memoria 14
2. Especificación de requisitos 16
   2.1. Introducción 16
      2.1.1. Propósito 16
      2.1.2. Ámbito 16
      2.1.3. Definiciones, acrónimos y abreviaturas 17
      2.1.4. Referencias 19
   2.2. Descripción general 19
      2.2.1. Perspectiva del proyecto 19
      2.2.2. Funciones del producto 19
      2.2.3. Características del usuario 21
      2.2.4. Restricciones 21
      2.2.5. Supuestos y dependencias 22
   2.3. Requisitos específicos 22
      2.3.1. Interfaces externas 22
      2.3.2. Requisitos funcionales 23
      2.3.3. Requisitos de rendimiento 26
3. Diseño 28
   3.1. Arquitectura de la aplicación 28
   3.2. Tecnologías utilizadas 29
      3.2.1. Electrón js 29

3.2.2. React js 30
3.3. Modelo de la aplicación 31
3.3.1. Ventanas 32
3.3.2. Capas 33
3.3.3. Modificaciones 37
3.4. Implementación 38
4. Resultado y Tour por la aplicación 41
5. Conclusiones y trabajos futuros 46
6. Bibliografía 47
7. Anexos 48

Índice de figuras

Figure 1 23
Figure 2 28
Figure 3 29
Figure 4 31
Figure 5 34
Figure 6 35
Figure 7 36
Figure 8 38
Figure 9 39
Figure 10 40
Figure 11 41
Figure 12 41
Figure 13 42
Figure 14 42
Figure 15 43
Figure 16 43
Figure 17 44
Figure 18 44
Figure 19 45

Índice de tablas

Table 1 24
Table 2 24
Table 3 24
Table 4 25
Table 5 25
Table 6 25
Table 7 26
Table 8 26
Table 9 26
Table 10 30

A mis padres y mi pareja,

sin ellos todo esto no habría sido posible.

1. Introducción

A lo largo de los últimos años y con la evolución de las nuevas tecnologías el uso de los navegadores web ha ido en aumento, no solo para su uso de ocio sino también para su uso de forma laboral.

En el mundo de la informática, dentro del campo del desarrollo web, es muy importante el uso del navegador a la hora de crear y gestionar proyectos, por lo que se hace necesario el uso de una buena aplicación. De este modo, el navegador es posiblemente el programa más importante de su equipo y el uso de uno adaptado a sus necesidades puede mejorar muy gratamente la experiencia de desarrollo, así como hacerla más eficiente.

Uno de los problemas que comúnmente se encuentran los desarrolladores web a la hora de enfrentarse a un proyecto es la gestión de cache, cookies y sesión. Estos problemas no solo se dan en desarrolladores sino también pueden darse en gestores de sistemas, comunity manager, project manager, front-end developer, etc.

1.1. Motivación

Dentro de las motivaciones que me han instado a la realización de esta aplicación hay dos que merecen mención.

La primera de ellas, nace de la necesidad de resolver un problema que encontrábamos diariamente en la empresa para la que trabajo y que suponía un retraso constante en las tareas que realizábamos.

Para entender bien el problema, la mejor opción es explicarlo mediante un ejemplo: Supongamos que usted es un community manager de una gran empresa, quien ha de acceder a 5 cuentas de Facebook para manejar diferentes servicios y publicaciones. En primer lugar, se accedería a Facebook desde un navegador a la primera cuenta. Sin embargo, al mismo tiempo se tiene que trabajar con otra cuenta por lo que se abriría otra pestaña para acceder a esta. Problema, la pestaña está en la misma cuenta ya que el navegador comparte sesión. Al abrir una nueva ventana no se soluciona el problema ya que seguimos teniendo el mismo comportamiento. Decidimos abrir una ventana de incognito e iniciamos sesión en la segunda cuenta.

Por último, para abrir la tercera cuenta seguimos el procedimiento anterior y abrimos una nueva ventana de incognito y para nuestra sorpresa encontramos que todas las ventanas de incognito comparten sesión. ¿Qué podemos hacer? Abrir un navegador diferente e iniciar sesión.

Por lo tanto, finalmente acabamos teniendo la necesidad de usar 3 navegadores para manejar las 5 cuentas diferentes de forma simultanea lo cual, ralentiza nuestro trabajo y lo hace muy tedioso al tener que ir cambiando entre ventanas, identificar donde esta cada una etc.

En consecuencia, encontramos el problema en cualquier trabajo en el que se nos exija utilizar muchas cuentas en servicios que no permiten iniciar sesión de forma simultánea. Esto, a su vez, nos obliga a tener distintos navegadores de forma concurrente perjudicándonos gravemente en nuestro trabajo, así como en los recursos consumidos por el sistema.

En segundo lugar, la escasa existencia de aplicaciones y alternativas que cumplen los requisitos necesarios para ser consideradas una solución asequible.

Mi proyecto de final de grado pretende dar solución a los problemas mencionados mediante el uso de un navegador web reducido. Así pues, deberá estar adaptado a las necesidades de los usuarios permitiendo la gestión de diferentes proyectos y separando la cache, cookies y sesión de una forma amigable y sencilla. Se busca crear un navegador que todo el mundo sea capaz de utilizar.

1.2. Estudio de Mercado

Previamente al inicio del proyecto y partiendo de la idea general de lo que se pretendía hacer, un navegador web con separación de sesión, se procedió a la búsqueda de aplicaciones y programas que hicieran tareas similares. De esta forma se podría ver si el proyecto era factible y si podía ofrecer alguna característica o funcionalidad novedosa/mejor a la encontrada.

Para ello se empezó por la solución más básica, un plugin o extensión que permitiese tanto la separación de sesión como de cache. Como alternativa se buscó por navegadores con esta funcionalidad para finalmente investigar cualquier tipo de aplicación que lo consiguiese.

Uno de los requisitos indispensables era que debía de poder ser utilizado desde cualquier sistema operativo, ya que, por necesidad de la empresa en la que trabajo, diferentes empleados usan diferentes sistemas operativos.

1.2.1. Plugins y extensiones

Dentro de este apartado podemos encontrar plugins y extensiones para varios navegadores que pretenden modificar dicha funcionalidad de los mismos, sin tener que sacrificar el uso de nuestro navegador favorito. Entre ellas podemos encontrar:

SessionBox

Una de las extensiones más conocidas que permite el uso de múltiples sesiones al mismo tiempo. Las sesiones se sincronizan de forma segura entre varios dispositivos utilizando una cuenta de SessionBox.

MultiLogin

Permite al usuario abrir otra sesión en cualquier página, aislando una pestaña determinada.

Easy Acount Switcher

Esta extensión permite iniciar sesión en un sitio y guardarlo como una sesión individual. Posteriormente nos permite alternar entre las diferentes sesiones.

El problema de esta extensión es que no nos permite tener varias sesiones, más bien guardarlas e ir cambiando entre ellas.

Swap my cookies

Esta extensión tiene un funcionamiento similar a 'Easy Acount Switcher' permitiendo al usuario almacenar la sesión y las cookies en un perfil para posteriormente alternar entre ellos.

Estas extensiones, si bien añaden funcionalidades interesantes a nuestro navegador, estos tienen una gran desventaja, el consumo de recursos que tienen. Si a este consumo le añadimos el consumo del propio navegador podemos llegar a tener una disminución de los recursos que puede afectar al rendimiento de nuestro pc, lo cual influye de forma bastante negativa a la hora de desarrollar desde el mismo.

Por otro lado, también hay que tener en cuenta que estas extensiones solo cumplen un papel, la separación/aislamiento de la sesión y/o cookies en el navegador, pero no la separación de cache que en gran medida puede afectarnos en el desarrollo de un proyecto. Ya que esta funcionalidad también es uno de los factores requeridos a la hora de diseñar el proyecto, si quisiéramos también esta funcionalidad como plugin o extensión deberíamos añadir otra extensión más. Esto puede influir de forma bastante negativa por lo que debemos plantearnos las siguientes preguntas: ¿la extensión/plugin para separar sesión es compatible con el plugin/extensión para separar cache? ¿Cuál es su coste en el rendimiento? ¿Cómo afecta esto a la carga del navegador?

En mi búsqueda de extensiones/plugins para separar cache o al menos manejarla, no he encontrado nada remarcable o que tenga un funcionamiento ideal que se adecue a nuestras necesidades por ello no se mencionara ninguna en este apartado.

1.2.2. Navegadores web

En lo que se refiere a navegadores web no hay mucha variedad donde elegir. Entre ellos podemos encontrar solo dos que merecen ser mencionados.

Ghost browser

Ghost browser es un navegador web, basado en chromium que permite hasta un máximo de 5 sesiones de forma gratuita. Al estar basado en chromium tiene una apariencia casi exacta a chrome, así como un funcionamiento similar.
Este navegador permite tanto la separación de sesión como la de cache y cookies. Entre sus rasgos más destacables está la posibilidad de colorear distintas pestañas para diferenciarlas, así como la de separar sesiones en diferentes workspaces.

Aunque esta parece una solución ideal, no está exenta de fallos, si bien consigue todos nuestros objetivos de forma perfecta, el cambio entre sesiones es un poco desagradable, haciendo parpadear la pantalla cada vez que se cambia entre ellas.

Otra de sus desventajas radica en el precio y es que si necesitamos separar bastantes sesiones es necesario pagar una cuota de forma mensual. Su precio no es excesivo, pero para empresas con un cierto número de empleados el precio puede llegar a escalar de forma drástica haciéndola una solución muy poco viable. Para poner un poco de contexto, en la empresa para la que trabajo somos más de 70 trabajadores. En las oficinas de Alcoy actualmente estamos 15 personas y en aumento. La cuota más baja son 10$ al mes si se paga de forma anual. Solo en nuestras oficinas de Alcoy estaríamos pagando un total de

1800$ anuales por esta funcionalidad. Si se decidiese que toda la plantilla pudiera disfrutar de estas ventajas llegaríamos a pagar 8400$, precio el cual no es asequible.

Multilogin

Con una funcionalidad casi exacta a 'Ghost browser' se presenta como una alternativa a este. Si bien incluye muchas más funcionalidades su precio excesivo lo hace una opción muy poco llamativa.

1.2.3. Otras aplicaciones

Siguiendo con mi investigación en la búsqueda de una solución que satisficiese nuestras necesidades, indague sobre aplicaciones independientes que cumplieran este propósito. Tras varios días de búsqueda no se encontró nada que tuviera funciones parecidas o similares por lo que la opción fue descartada inmediatamente.

1.3. Objetivos del Proyecto

A grandes rasgos mi proyecto de final de grado persigue la creación de un navegador web simple, amigable, fácil de usar y que cumpla una función muy concreta dentro del ámbito laboral, la separación de sesión, cache y cookies.

1.4. Estructura del resto de la memoria

La presente memoria se encuentra estructurada en los siguientes apartados:

Introducción
En este primer apartado el lector es introducido en la memoria, de forma que de un vistazo y de forma concisa entienda el proyecto, así como su motivación. También se realiza un estudio de mercado, dando paso a los objetivos del proyecto.

Especificación de requisitos
En este apartado se detalla la especificación de requisitos según la normativa IEEE 830. Se trata de definir el comportamiento de forma detallada de la aplicación, así como de los requisitos y demás funcionalidades del sistema de una forma coherente y detallada.

Análisis y definición del problema
Se trata de un análisis del problema, definido de forma muy sencilla y simple, para que cualquier lector sea capaz de comprender la complejidad del mismo, además de los requerimientos del proyecto. También se encuentran una serie de diagramas que muestran el flujo de la aplicación.

Diseño
Esta parte se centra en el diseño y solución del problema. Se explican las tecnologías utilizadas, así como las diferentes partes en las que se divide la solución y como han sido implementadas y llevadas a la práctica.

Implementación
Se tratan los componentes y el código más importante de la aplicación, realizando un resumen breve y conciso de los mismos haciendo uso de imágenes.

Resultado y tour por la aplicación
El apartado siguiente, se muestra el resultado obtenido y se realiza un tour por la aplicación de forma que se pueda ver de primera mano el flujo de la misma, y la interacción del usuario desde el primer uso de la aplicación.

Conclusiones y trabajos futuros
Por último, se realiza un análisis y evaluación del producto final obtenido. Se plantean las conclusiones a las que se han llegado tras la realización del proyecto, opiniones de los usuarios finales de la aplicación y trabajos y mejoras pensadas para un futuro.

Además, este proyecto viene acompañado con imágenes y gráficos que ayudan a ejemplificar, explicar y entender ciertos aspectos del propio proyecto. Por último, se adjunta la bibliografía y documentos utilizados, que han servido de ayuda y apoyo para realizar este proyecto.

2. Especificación de requisitos

2.1. Introducción

En este apartado se procederá con la realización de una especificación de requisitos siguiendo el estándar IEEE 830. Tiene el propósito de definir el comportamiento detallado del proyecto, objetivos, definiciones, requisitos funcionales, no funcionales y de sistema para una aplicación web, con el fin de entender por qué se ha realizado esta aplicación.

2.1.1. Propósito

El propósito de la especificación de requisitos es dar una descripción completa de la aplicación que se va a desarrollar, un navegador web con separación de sesión, así como describir las características del mismo y las necesidades de los usuarios.

Este documento pretende dar una visión global, detallada y de forma sencilla para que cualquier persona pueda hacerse una idea de la aplicación que se va a desarrollar.

La especificación de requisitos representa una parte muy importante dentro de la memoria, donde se definen las características que debe cumplir y satisfacer para cada uno de los requerimientos.

2.1.2. Ámbito

La aplicación desarrollada en este proyecto, consiste en la creación de un navegador web capaz de separar sesión, cookies y cache. Dirigida a usuarios dentro del ámbito laboral, se enfoca sobre todo en aquellas personas, que tienen la necesidad de trabajar con navegadores que les permitan un control total sobre las sesiones y las cookies o deseen mejorar la productividad. Si bien puede ser utilizado por cualquier tipo de usuario, este navegador no ofrece las mismas características que un navegador convencional.

El navegador debe ser agradable y sencillo. Debe ofrecer una alta usabilidad sin dejar de lado ninguna característica necesaria del mismo. El usuario debe ser capaz de utilizarlo

sin ningún tipo de ayuda, por lo que se ha decidido utilizar un diseño conocido y simple como el de los navegadores más conocidos y utilizados.

La aplicación debe ofrecer el mismo comportamiento en todos los sistemas operativos, así como ser instalable y ejecutable en cualquier de ellos.

2.1.3. Definiciones, acrónimos y abreviaturas

En esta sección se pasarán a definir una serie de elementos, de los cuales se debe tener conocimiento para la completa comprensión del documento.

Navegador web
Se trata de un programa capaz de traducir el código que conforma las páginas web en texto, imágenes, videos o aplicaciones web de forma que los humanos puedan entender e interactuar.

Cookies
Las cookies son archivos que contienen diversos datos acerca del navegador, la actividad web y el equipo. La función principal de estas es facilitar en lo máximo posible el uso de internet, haciendo uso de la memoria interna del navegador para recordar datos como puedan ser los artículos que un usuario almacena en el carrito de una tienda online o guardar la información de inicio de sesión, ahorrando al usuario escribir una y otra vez sus datos.

Sesión
La sesión es un tipo de cookie temporal que solo permanece en el equipo mientras se navega por la red. Estas, no guardan ningún dato sobre el usuario, pero al mismo tiempo permiten navegar entre las distintas páginas de una web sin tener que iniciar sesión cada vez. Este tipo de cookies son eliminadas al cerrar el navegador o al pasar un tiempo predefinido.

Cache
La cache del navegador es una función que tienen los navegadores, que permite cargar páginas web de forma más rápida, guardando en el ordenador parte de la información que previamente ha sido solicitada a una web. Almacenan información de webs que ya han

sido accedidas, para que, en futuras visitas a la misma, el acceso sea más rápido pues no se ha de solicitar la información que ya se tiene.

Espacio de trabajo/workspace
Adaptando el significado de la palabra a mi propio proyecto, un espacio de trabajo puede ser entendido como una ventana de navegación la cual no comparte cookies, cache o sesión con otras ventanas.

Extensión de navegador
Son pequeños programas o aplicaciones que se instalan dentro del navegador y añaden o mejoran funciones del navegador desde la propia aplicación. Las extensiones pesan muy poco y son instaladas desde un repositorio que normalmente está gestionado por el propio navegador.

Al instalarse, suelen crear un acceso directo en el navegador desde el cual las podemos abrir y ejecutar.

Plugin
Un plugin puede entenderse como una extensión del navegador, el cual pretende mejorar las funciones del navegador ya sea de forma visible para el usuario o no.
La diferencia entre plugin o extensión es que por lo general los plugins suelen venir integrados con el navegador, además de que estos no suelen permitir la personalización como lo hacen las extensiones.

Bookmark/favorito/marcador
El bookmark o marcador es una herramienta que implementan algunas aplicaciones y navegadores web que permite almacenar una lista de direcciones web las cuales el usuario puede encontrar interesantes, de forma que pueden ser recordadas y visitadas a posteriori de forma fácil y rápida.

URL
Universal Resource Locator. Sistema unificado de identificación de recursos en la red. Es una cadena que suministra el nombre y dirección de internet a un sitio Web o a un recurso World Wide Web.

Pestaña
Es un elemento de las interfaces de los programas. Permite cambiar rápidamente que se está viendo sin tener que cambiar la ventana que usa el programa o menú.

2.1.4. Referencias

La principal referencia que se ha seguido para la elaboración de este documento es el estándar IEEE para la especificación de requisitos software.

IEEE Std. 830-1998 – Recommended Practice for Software Requirements Specifications

2.2. Descripción general

Los siguientes apartados proporcionan una descripción de todos los factores que afectan a la aplicación y que influyen directamente en los requisitos de esta. Suministran un contexto que permitirá definir con detalle los requisitos en la siguiente sección.

2.2.1. Perspectiva del proyecto

La aplicación, el navegador web con separación de sesión, debe permitir al usuario navegar entre diferentes webs e iniciar sesión en estas y tras un cambio de espacio de trabajo, debe permitir el acceso a la misma web como si se tratase de la primera vez, permitiendo el inicio de sesión nuevamente.

Debe de poder ser ejecutada en cualquier sistema operativo, ya sea Windows, Linux o MacOS, al menos en sus dos últimas versiones, para así poder permitir el mayor número de usuarios dentro de cada sistema.

Para poder utilizar la aplicación, es necesaria una conexión a internet, a menos que solo se quiera utilizar para acceder a proyectos y aplicaciones de forma local, es decir, que se ejecuten en el propio ordenador.

2.2.2. Funciones del producto

La aplicación final debe cumplir las siguientes funciones:

Espacios de trabajo

Crear y eliminar múltiples espacios de trabajo: el usuario debe poder crear tantos espacios de trabajo como desee, así como también debe poder eliminarlos. Al crear un espacio de trabajo, este tiene un nombre identificativo y si se desea una imagen que permita, de un vistazo, identificarlo. Además, se crea una ventana con el navegador listo para funcionar cada vez que se inicia o crea un espacio de trabajo.

Cambiar la imagen de un espacio de trabajo: las imágenes identificativas de los espacios de trabajo son modificables y no tienen restricción en el formato de imagen. Si el usuario no está satisfecho con la imagen seleccionada puede eliminarla o cambiarla por otra.

Cambiar entre los espacios de trabajo: en cualquier momento el usuario puede alternar entre los espacios de trabajo que ha creado.

Separar la sesión, cache y cookies entre espacios de trabajo: estos son independientes, es decir, no comparten nada entre ellos salvo la funcionalidad. La ventana de cada espacio de trabajo actúa como si de un navegador nuevo se tratase.

Navegador

Navegar a una url: el navegador debe de ser capaz de acceder a una url especificada por el usuario, sin ninguna restricción, desde la barra de direcciones.

Realizar búsquedas: si el usuario ingresa un texto diferente a una url en la barra de direcciones, el navegador realizara una búsqueda en google de forma automática, mostrando los resultados obtenidos.

Controles de navegación: el navegador debe de tener al menos tres controles de navegación. Dos que permitan al usuario navegar hacia delante y hacia atrás dentro del propio histórico de direcciones que se han visitado en la pestaña. Un control que permita refrescar la pestaña actual, volviendo a cargar el contenido de la misma.

Abrir y cerrar pestañas: si el usuario desea, podrá abrir y cerrar tantas pestañas como quiera dentro de la ventana del navegador. Entre las pestañas de un mismo espacio de trabajo, sí que se compartirá la sesión, cache y cookies. No hay restricción en el número de pestañas que se pueden llegar a tener abiertas.

Abrir consola de desarrolladores: el usuario, puede abrir la consola de desarrolladores dentro de la pestaña que desee, así como inspeccionar sus elementos. Desde aquí puede gestionar todos los apartados que la consola ofrece, teniendo en cuenta, que esta solo visualiza el acceso a la web que se está visualizando y no al contenido de la propia aplicación.

Guardar y abrir favoritos:

Otros

Atajos del teclado: la aplicación ofrece una serie de atajos de teclado que maximizan la eficiencia y comodidad del usuario. Estos atajos permiten abrir nuevos espacios de trabajo, cambiar entre diferentes espacios de trabajo, abrir y cerrar pestañas e intercambiar entre estas, manejar los controles de navegación y abrir la consola de desarrolladores.

Control de memoria y rendimiento: la aplicación dispone de un control interno de memoria y CPU. Si un espacio de trabajo está inactivo durante demasiado tiempo este es cerrado. Si la aplicación está consumiendo una cantidad de recursos excesivos limita la funcionalidad para no afectar al resto del sistema.

Privacidad: el navegador no recopila ningún tipo de información por parte del usuario. Así mismo el historial de navegación es privado para el usuario y nadie más puede tener acceso a él.

2.2.3. Características del usuario

Este navegador está dirigido a usuarios dentro de un ámbito laboral, no obstante, puede ser utilizado por cualquiera.

El usuario debe tener un conocimiento básico sobre navegadores y su funcionamiento.

La aplicación está pensada para ser simple e intuitiva, permite identificar todas sus funciones de un vistazo, por lo que cualquier usuario con conocimiento básico puede utilizarla, sin embargo, gente experimentada puede llegar a sacarle más partido utilizando las ventajas que la misma ofrece.

2.2.4. Restricciones

Es imprescindible poseer una conexión a internet, en caso que se desee navegar por la red con la aplicación.

Si bien las restricciones vienen dadas por la tecnología misma, ya que al ser una aplicación basada en chromium el sistema debe cumplir los requisitos mínimos impuestos por esta para poder ser ejecutada.

2.2.5. Supuestos y dependencias

Puesto que uno de los requisitos indispensables de la aplicación es su funcionamiento bajo cualquier sistema operativo, esto no supone una limitación, si bien hay que tener en cuenta posibles limitaciones de hardware que tenga la máquina del usuario.

Se debe tener en cuenta que por cómo está concebida la aplicación, se debe tener un buen computador si se quiere utilizar un gran número de espacios de trabajo y trabajar con todos ellos al mismo tiempo de forma rápida y fluida.

La aplicación viene empaquetada y contiene de forma interna las dependencias de la misma por lo que hace necesario disponer de un software capaz de extraer la aplicación de un archivo comprimido Zip.

2.3. Requisitos específicos

Por último, se tratarán los requisitos específicos de la aplicación. Este apartado contiene los detalles suficientes para permitir la creación de un diseño que satisfaga los requisitos mencionados anteriormente.

2.3.1. Interfaces externas

La aplicación tiene un interfaz único para todos los usuarios de la misma como se puede comprobar en la siguiente imagen:

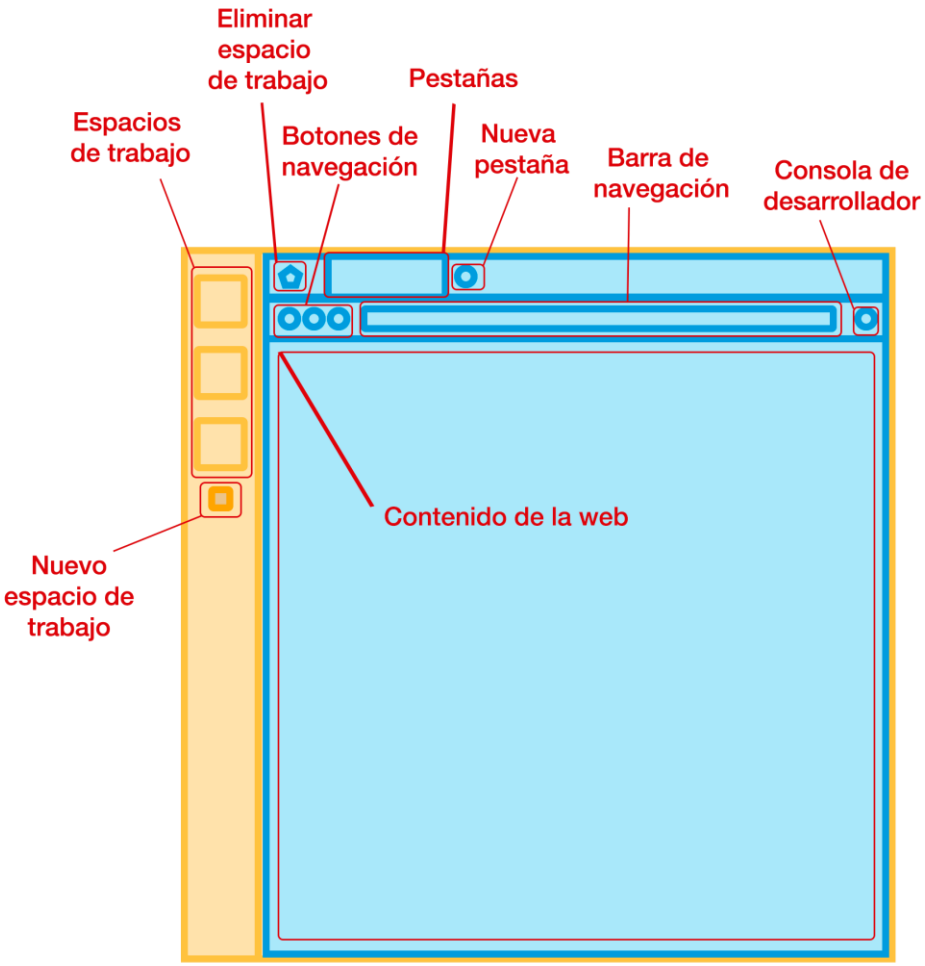


Figure 1

2.3.2. Requisitos funcionales

A continuación, se describen los requisitos funcionales que debe cumplir la aplicación. Estos, son iguales para todos los usuarios.

Espacios de trabajo

Crear espacio de trabajo	
Propósito	Crear un nuevo espacio de trabajo
Entrada	Nombre identificador. Opcionalmente imagen.
Proceso	Se crea un nuevo espacio de trabajo y se abre el navegador web del mismo, pasando a segundo plano otro espacio de trabajo si se estaba trabajando en el actualmente.
Salida	Se visualiza la nueva ventana con el navegador web.

Table 1

Eliminar espacio de trabajo	
Propósito	Eliminar un espacio de trabajo
Entrada	-
Proceso	Se elimina el espacio de trabajo y se borra la información relacionada a él.
Salida	El espacio de trabajo es cerrado y deja de mostrarse en la aplicación.

Table 2

Cambiar de espacio de trabajo	
Propósito	Alternar entre diferentes espacios de trabajo.
Entrada	-
Proceso	Se oculta la ventana del espacio de trabajo actual para crearse y o mostrarse la ventana del espacio de trabajo nuevo.
Salida	Se visualiza la ventana del espacio de trabajo seleccionado.

Table 3

Cambiar la imagen del espacio de trabajo	
Propósito	Cambiar la imagen de un espacio de trabajo.
Entrada	Identificador del espacio de trabajo y nueva imagen.
Proceso	La aplicación carga la nueva imagen, la almacena y la muestra en el espacio de trabajo.
Salida	Se visualiza la nueva imagen en el espacio de trabajo seleccionado.

Table 4

Ventana del navegador

Crear pestaña	
Propósito	Crear una nueva pestaña para navegar.
Entrada	-
Proceso	Se crea una nueva pestaña en la parte superior de la ventana con los controles y ventana de navegación.
Salida	Se muestra la nueva pestaña con su ventana de navegación.

Table 5

Abrir consola de desarrollador	
Propósito	Crear una nueva ventana con todas las herramientas de desarrollador.
Entrada	-
Proceso	Se crea una ventana embebida o independiente, mostrando las herramientas de desarrollador sobre la web que se está visualizando.
Salida	Nueva ventana o en caso de estar embebida nuevo apartado con las herramientas de desarrollador.

Table 6

Navegar hacia delante	
Propósito	Navegar hacia delante en el histórico de webs visitadas desde la pestaña.
Entrada	-
Proceso	Se busca en el historial la web posterior en la lista, si es que hay, y se carga en el navegador.
Salida	Se avanza en el historial de navegación mostrando la web siguiente visitada.

Table 7

Navegar hacia atrás	
Propósito	Navegar hacia atrás en el histórico de webs visitadas desde la pestaña.
Entrada	-
Proceso	Se busca en el historial la web anterior en la lista, si es que hay, y se carga en el navegador.
Salida	Se retrocede en el historial de navegación mostrando la web siguiente visitada.

Table 8

Recargar pagina	
Propósito	Volver a cargar la web que se está visitando.
Entrada	-
Proceso	Se envía de nuevo una petición al servidor donde está alojada la web y esta es cargada de nuevo.
Salida	Se visualiza como se re carga la web.

Table 9

2.3.3. Requisitos de rendimiento

El sistema donde se ejecuta la aplicación, independientemente del sistema operativo, debe cumplir unos requisitos mínimos. Estos son:

Procesador: 1.0 GHz.

Memoria: 256 MB de RAM.

Gráficos: Tarjeta con 64 MB de RAM.

Disco duro: 500 MB.

Para conseguir que la aplicación sea lo más eficiente posible, se recomienda superar estos límites pues a mayor número de espacios de trabajo y pestañas se quiera al mismo tiempo, más recursos son consumidos.

3. Diseño

La solución al problema mencionado anteriormente viene dada en la creación de un navegador web que agrupe todas estas necesidades en una única ventana, permitiendo un fácil manejo de diferentes espacios de trabajo de forma simultánea y con bajo consumo de recursos.

Por ello se presenta como solución este navegador, que permite la separación de sesión, cache y cookies por espacios de trabajo, el fácil cambio de contexto entre ellos, y la posibilidad de ser usado en diferentes sistemas operativos.

3.1. Arquitectura de la aplicación

Mi aplicación pone en conjunción una serie de tecnologías, electrón y react, para la creación de aplicaciones de escritorio mediante tecnología web. La fuerza reside en su posibilidad para ser ejecutada en diferentes sistemas operativos realizando una sola implementación.

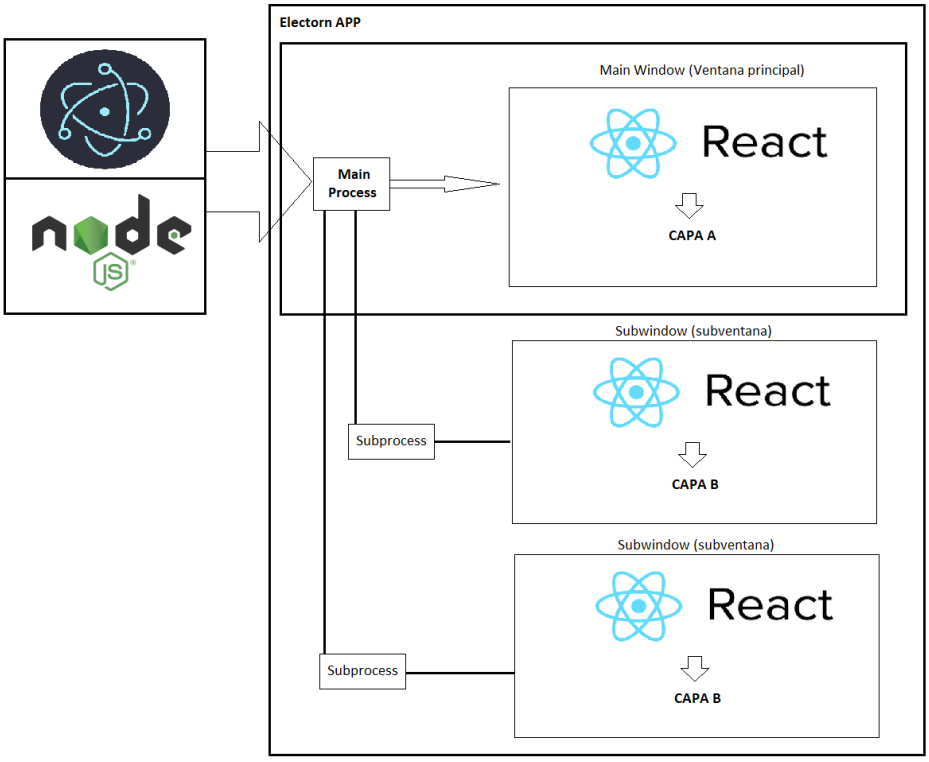


Figure 2

3.2. Tecnologías utilizadas

3.2.1. Electrón js

Electrón es un framework para JavaScript que permite el desarrollo de aplicaciones de escritorio mediante el uso de tecnologías web. Desarrollado por GitHub y de código abierto, permite el uso y la programación multiplataforma.

Funciona haciendo uso de un subconjunto mínimo de librerías de chromium. Mediante el motor de chromium hace posible el uso de librerías y Apis nativas, lo cual implica que además permite hacer enlace a librerías propias del sistema operativo. Por otro lado, su desarrollo está basado en Node.js lo que incorpora una gran cantidad de paquetes, facilitando su modificación y la implementación de nuevas funcionalidades.

El funcionamiento de electrón es muy simple. Trabaja creando dos tipos de procesos, el main que contiene el Core o núcleo de la aplicación y el proceso render. Existe un solo proceso main, del cual se pueden crear varios procesos render y estos se comunican entre ellos mediante el pase de mensajes, como si se tratase de un sistema distribuido.

Hay que entender electrón como un envoltorio de aplicaciones web, es decir, electrón crea una ventana y en el interior de la misma puede ser cargado cualquier contenido web, ya sea local o remoto.

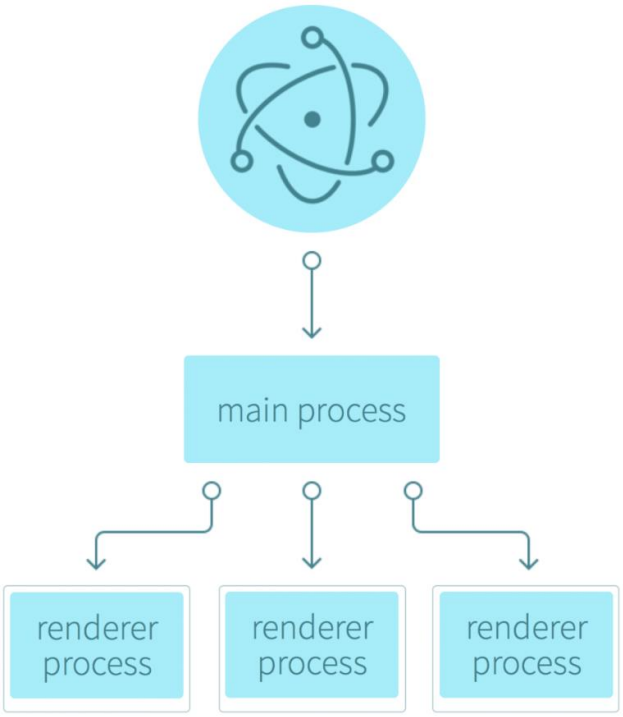


Figure 3

Grandes compañías como Docker, GitHub, Facebook e incluso Microsoft hacen uso de esta tecnología. Uno de los ejemplos más conocidos de la utilización de este framework es *Microsoft Visual Studio Code*.

3.2.2. React js

React.js es una librería JavaScript que se focaliza en el desarrollo de interfaces de usuario. Sin embargo, aunque react se define como tal, este no es todo su propósito, ya que es una excelente herramienta para crear todo tipo de aplicaciones web, spa (single page application) o aplicaciones móviles.

React está formado por un ecosistema de módulos, herramientas y componentes que permiten crear de forma rápida y con poco esfuerzo grandes aplicaciones.

En estos últimos años, el auge de react ha ido creciendo en comparación a otras librerías con funcionalidades similares, como pueden ser Angular o vue. Si bien vue está ganando una gran cantidad de seguidores, la verdad es que los usuarios siguen prefiriendo react.

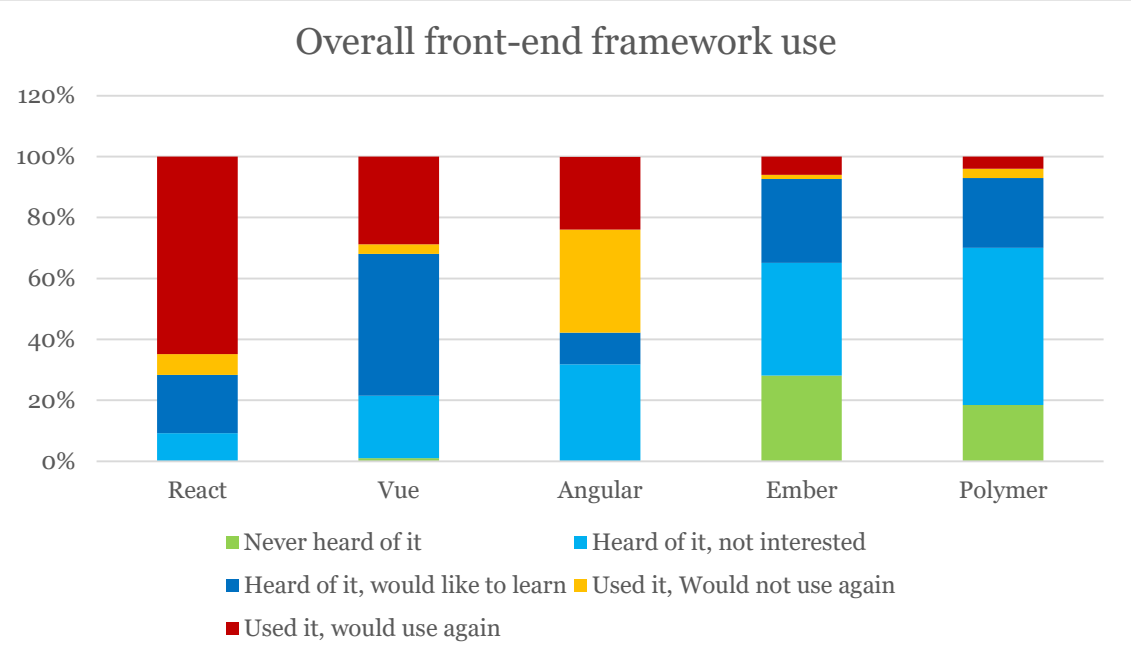


Table 10

Grafica obtenida de una encuesta realizada por **stateofjs.com** en 2018.

3.3. Modelo de la aplicación

El navegador web ha sido implementado siguiendo un modelo de capas y ventanas, diferente a lo que normalmente es utilizado en un proyecto web. Por lo general, los proyectos o aplicaciones web, se basan en cargar una serie de contenido dentro de un proceso de renderizado. Todas las páginas y secciones son cargadas y controladas dentro del mismo proceso.

El modelo elegido para este proyecto propone un proceso central, encargado de manejar y gestionar diferentes subprocesos de renderizado donde cargaran los espacios de trabajo. Por ello se pueden distinguir dos secciones bien diferenciadas: la capa 1 que contiene el core y la funcionalidad básica de la aplicación, y la capa 2 que contiene la lógica de navegación. Ambas secciones funcionaran en ventanas diferentes de la aplicación formando un conjunto.

Electrón js es el encargado de crear el contenedor de la aplicación. Este crea una ventana de aplicación donde en su interior es cargado react.

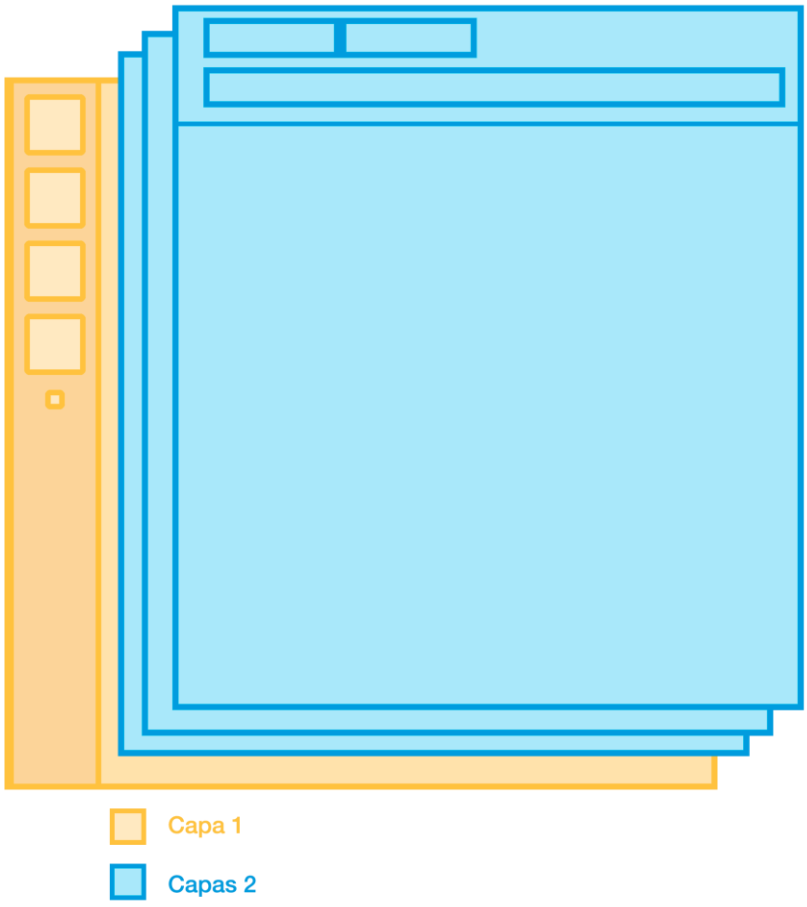


Figure 4

Como se puede observar en la imagen anterior, la capa 1 junto a su ventana no es más que un contenedor que gestiona, maneja y muestra diversas ventanas que contienen la capa 2.

La ventaja principal ofrecida por este tipo de aplicación, es que, con solo dos capas, es posible crear un sinfín de espacios de trabajo.

A continuación, se procederá a explicar las ventanas y las capas que conforman la aplicación. Se pretende comprender de forma más clara el funcionamiento de la aplicación y el papel que cumplen cada uno de los elementos.

3.3.1. Ventanas

Dentro de la aplicación existen dos tipos de ventanas con funciones diferentes. La primera de ellas es el contenedor de la aplicación, un BrowserWindow. Esta ventana es creada por defecto por electrón y conforma el core de la aplicación. Dentro de ella se muestra el contenido web que se quiera, así pues, esta es creada para mostrar nuestra aplicación de react de forma local como si de una app de escritorio se tratase. Dentro de esta ventana se cargará la capa 1, que contiene la gestión de ventanas de navegación y de los proyectos, así como la lógica básica.

Por otro lado, encontramos el segundo tipo de ventanas, el BrowserView. Este tipo de ventana es un tanto especial, ya que su funcionamiento es el mismo que el de la ventana anterior, pero con ciertas peculiaridades. La ventana, es creada sin ningún tipo de marco o borde dentro de la ventana principal (BrowserWindow). Permite ser posicionada donde queramos dentro de la ventana, tener múltiples ocurrencias de la misma y ser ocultadas o mostradas a placer. Se podría decir que actúa como un webview en tanto que carga contenido web dentro de ella, siendo su principal característica que al ocultarse el contenido web sigue ejecutándose. Por ello, esta ventana es creada para cada espacio de trabajo, mostrando la capa 2, que contiene el navegador, y es ocultada y mostrada dependiendo del espacio de trabajo activo.

La ventana principal se encarga de crear las distintas subventanas, según la necesidad y de mostrarlas u ocultarlas. Las subventanas cargan el contenido del navegador siguiendo la lógica de separación de cache, sesión y cookies.

Un dato interesante es que las subventanas forman cada una parte de un subproceso que parte de la ventana principal, por lo que permanecen aisladas de las demás subventanas, pero no de la ventana principal.

Con toda esta información, se llega al entendimiento de que la ventana principal es el envoltorio de la aplicación, mientras que las subventanas siguen un funcionamiento similar a las pestañas de un navegador. Las subventanas permanecen aisladas y ofreciendo la separación de cache, cookies y sesión que se busca.

3.3.2. Capas

Las capas que conforman el navegador no son más que la división de implementación y funcionalidades de la aplicación, como si se tratasen de clases o vistas. La capa 1 conformaría la clase o vista A y la capa 2 conformaría la clase o vista B. De esta forma, la ventana principal muestra la capa 1 con las funcionalidades que esta contiene, mientras que las subventanas creadas para cada espacio de trabajo, crean y muestran la capa 2 conformada por el navegador.

3.3.2.1. Capa1 (Vista A)

Esta capa contiene la funcionalidad básica de la aplicación. Se encarga de crear las ventanas de la capa 2 así como de gestionarlas. Se parte de una ventana que solo contiene un menú a la derecha. El resto de la ventana permanece vacía y ya que sobre ella serán mostradas las ventanas con la capa 2. Por ello esta parte solo contiene una imagen, que solo será mostrada cuando no hay espacios de trabajo abiertos.

Figure 5

Como se puede apreciar en la imagen superior, el menú principal se encuentra dividido en dos partes.
El apartado 1 muestra la lista de espacios de trabajo en los que se está trabajando actualmente. Los espacios de trabajo se identifican con una imagen de forma que el usuario pueda identificarlos rápidamente de un vistazo. Así mismo, si el ratón permanece sobre la imagen, se mostrará un texto informativo con el nombre de este. También se puede visualizar que debajo de cada espacio de trabajo aparece un texto. Este informa del atajo de teclado con el cual se puede abrir cada espacio de trabajo.

La función interna de estos botones es la siguiente:
Si la capa perteneciente al espacio de trabajo está en ejecución, pero oculta, esta pasa a mostrarse.
Si la capa no estaba en ejecución, se procederá a crear un nuevo proceso con la información almacenada de este espacio. Se carga la información, se crea la ventana con la información y se muestra la ventana al mismo tiempo que se oculta la ventana anterior.

En el apartado 2 se encuentra el botón que permite crear nuevas ventanas. Al ser presionado mostrara una ventana con el formulario para crear nuevos espacios de trabajo.

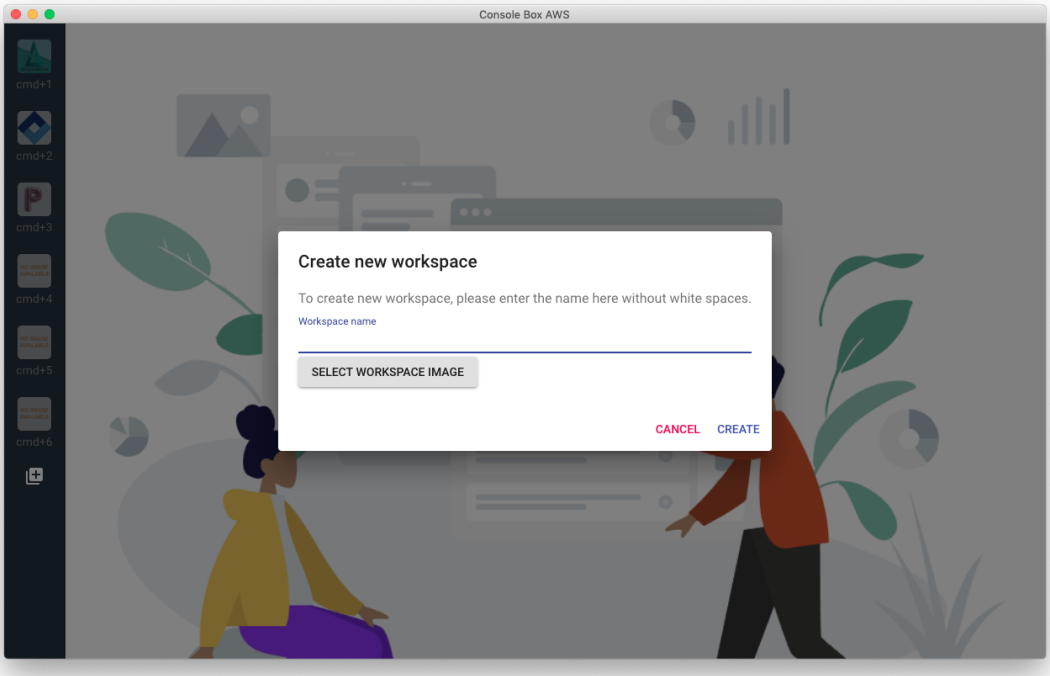


Figure 6

Solo se necesitan dos datos para la creación de un espacio de trabajo nuevo: Un nombre (este no puede estar duplicado si ya existe uno con el mismo nombre) y una imagen identificativa. Si ninguna imagen es seleccionada, se asignará una imagen por defecto, que puede ser cambiada posteriormente.

Cuando se crea el nuevo espacio de trabajo, este es añadido a un fichero local en formato JSON relativo al directorio de instalación, que contiene toda la información relativa a estos.
Las imágenes utilizadas y almacenadas por la aplicación, son convertidas y guardadas en base 64. Esto implica que no existen restricciones a la hora de utilizar imágenes, es decir, no importa el formato usado, la aplicación lo mostrará, inclusive imágenes animadas como son GIFS. Otro aspecto relevante de esta conversión es su velocidad, ya que ya no se lee una imagen, sino un String.

Un aspecto de la aplicación que se encuentra en esta capa son los atajos del teclado. Estos agilizan el movimiento dentro de la aplicación y hacen su uso más cómodo y sencillo. Se dispone de un manejador que actualiza y sincroniza estos atajos con el resto de espacios de trabajos, de forma que si se añade un espacio de trabajo se le asigna un atajo, si se reordenan los espacios de trabajo, los atajos son intercambiados, etc.

3.3.2.2. **Capa 2 (Vista B)**

Esta capa es la encargada de la navegación. Contiene el navegador con sus pestañas, así como todas sus funcionalidades.

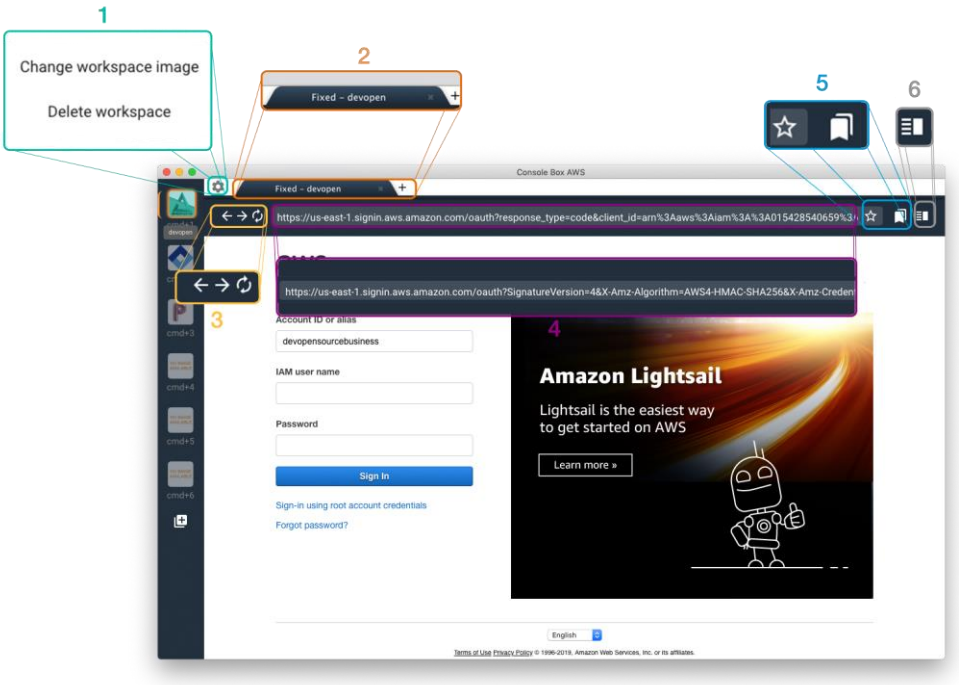


Figure 7

En el apartado 1 encontramos las propiedades de la capa 2, como son cambiar la imagen del espacio de trabajo y eliminar el espacio de trabajo. Puesto que las imágenes están situadas en la capa 1, estos botones se comunican con esta capa mediante el paso de mensajes. La capa 2 envía un mensaje indicándole a la capa 1 como debe proceder, cambiando la imagen o borrando el espacio de trabajo y eliminando el subproceso.

En el apartado 2 se pueden ver las pestañas. Estas contienen el resto de apartados de esta capa. No hay límite en el número de pestañas abiertas, y estas pueden ser reordenadas, duplicadas y cerradas. Su funcionamiento es muy simple, así como complejo. La pestaña contiene en su interior un espacio el cual es rellenado con el resto de apartados que forman un conjunto en sí, el navegador, Cada pestaña crea un nuevo conjunto permitiendo la navegación.

En el apartado 3 está situado los botones de navegación. Entre estos botones encontramos 2 flechas, encargadas de la navegación entre el historial de la ventana. Permiten avanzar o retroceder dentro de las páginas que se han visitado desde una misma ventana. Por último, se encuentra el botón encargado de refrescar la ventana que permite volver a cargar la página en la cual nos encontramos.

En el apartado 4 se sitúa la barra de navegación. Está, muestra la url de la página actual y nos permite, así mismo, realizar búsquedas. Por defecto, si se realiza una búsqueda la cual no se trate de una url, la buscará en google y la mostrará los resultados.

El apartado 5 está enfocado en los bookmarks o favoritos. Permiten guardar una dirección o url para ser abierta posteriormente desde cualquier espacio de trabajo. En principio estos bookmarks son compartidos entre todos los proyectos, no obstante, se está pensando en un sistema para poder almacenar favoritos tanto privados solo para un workspace como públicos, es decir, compartidos por todos.

El apartado 6 es el encargado de abrir la consola de desarrolladores. Esta consola es idéntica a la que encontramos en Chrome y permite debugguear la aplicación, así como inspeccionar todos sus elementos.

Por ultimo en el centro se encuentra la ventana de navegación que es la encargada de cargar las páginas. Esta se trata de un webview, con la cual se cargan en su interior contenidos web. Mediante el uso de todos los elementos mencionados anteriormente, este webview pasa a convertirse en un navegador web completo.

3.3.3. Modificaciones

React propone un funcionamiento SPA (Single Page Aplicación) donde todo se carga sobre una misma ventana. Esto tiene sus ventajas y desventajas. Cabe mencionar que mi aplicación requiere de un modelado por capas por lo que se ha tenido que implementar un sistema que transforme el SPA a una aplicación con 2 paginas principales.

Se ha creado un manejador que dependiendo en la capa en que nos situamos se cargara una página u otra como si de la principal se tratase.

Resulta interesante mencionar 2 apartados importantes de por qué se ha elegido este sistema de capas. Hay que entender el funcionamiento del sistema de capas como si se tratase de un sistema distribuido donde las dos capas se comunican entre ellas mediante mensajes.
El primer motivo es el rendimiento y carga de trabajo que supone la aplicación al sistema. Cada workspace se ejecuta en un subproceso independiente del resto por lo que si no se abre un workspace este no consume recursos. Por otro lado, si un workspace lleva cierto tiempo sin ser utilizado, se almacena su estado y se elimina el subproceso. Esto penaliza el tiempo de apertura de workspaces que llevan tiempo sin utilizarse (milisegundos) pero hace que la aplicación consuma muy pocos recursos.

Además, hay que tener en cuenta la recuperación ante errores. Si por lo que sea un workspace falla, solo es afectado ese subproceso por lo que es eliminado y vuelto a crear sin interferir en ningún otro workspace.

3.4. Implementación

Hay varios detalles de la implementación de la aplicación que resultan interesantes mencionar, ya que, son partes fundamentales de la misma y aportan una funcionalidad esencial a la misma, sin la cual la aplicación no sería posible. En este apartado no se pretende mostrar todo el código de la aplicación, solo partes fundamentales que hacen posible la aplicación y que consiguen implementar las funciones deseadas como es la separación y aislamiento de cada espacio de trabajo.

Lo primero que debe ser resaltado es una línea de código que muy poca gente utiliza, y que permite de forma automática cambiar entre el modo de desarrollo y el modo de producción si tener que tocar nada mas de código. De esta forma cuando se ejecuta el código como developer habilita las funciones para debuguear e inspeccionar la aplicación y si se compila y empaqueta la aplicación para su distribución utilizara las rutas estáticas de los diferentes archivos.

```
isDev? view.webContents.loadURL(`http://localhost:3000?view=viewB&session=${workSpace}`) :
view.webContents.loadURL(`file://${path.join(remote.app.getAppPath(),
`./build/index.html?view=viewB&session=${workSpace}`)}`);
```

Figure 8

Como se puede ver, si esta en desarrollo, se cargan los archivos desde localhost, sino desde fichero estático.

El funcionamiento de react es el de un SPA (Single Page Application) por lo que todo el contenido carga dentro de un mismo index.html. Esto tiene sus ventajas dentro de la creación de una aplicación pues por ejemplo solo se debe crear un menú del que harán uso todas las páginas. En el caso de nuestra aplicación la desventaja radica en que, al usar diferentes ventanas, si llamamos al mismo archivo para cargar las diferentes capas, siempre se mostrara la misma. Esto se ha solucionado creando un manejador de vistas, en el que dependiendo de la capa que se quiera cargar, se añade un parámetro u otro en la url y el manejador cargará ese contenido en el index.html.

```
export default class ViewManager extends React.Component {

  static Views(sess) {
    return {
      viewA: <ViewA />,
      viewB: <ViewB session={sess} />
    }
  }
  static View(props) {

    const values = queryString.parse(props.location.search);
    let name = values.view;
    let session = values.session;
    let view = ViewManager.Views(session)[name];


    if(view == null){
      throw new Error("View '" + name + "' is undefined");
    }
    return view;
  }

  render() {
    return (
      <Router>
        <div>
          <Route path='/' component={ViewManager.View}/>
        </div>
      </Router>
    );
  }
}
```

Figure 9

Por último, se muestra la lógica de creación de subventanas con la capa 2, que permite el aislamiento y separación de cookies, cache y sesión.

```
var test = BrowserWindow.getAllWindows()[0];
var workspace = this.getWorkspaceByName(workSpace);
let view = BrowserView.fromId(workspace.id);
if(view){
  test.setBrowserView(view);
  view.setBounds({ x: 73, y: 0, width: test.getSize()[0]-73, height: test.getSize()[1]-26 });
  // isDev? view.webContents.openDevTools():console.log('Production mode');
  this.setActive(workSpace);
}else{
  let view = new BrowserView({
    webPreferences: {
      nodeIntegration: true,
      partition: `persist:${workSpace}`,
      devTools: true,
    },
    parent: test
  });

  test.setBrowserView(view);
```

Figure 10

Al seleccionar un espacio de trabajo, se comprueba si este existe o no. En caso afirmativo, se abre la subventana, de lo contrario, se crea la subventana y se abre. Como se observa en la *Figura 9* la subventana (BrowserView) tiene una preferencia signada, denominada partition. Esta preferencia es la encargada de aislar la subventana por un nombre, en nuestro caso el nombre del espacio de trabajo. De esta forma la subventana solo comparte sesión, cache y cookies con las ventanas que estén en la misma partición. Cada subventana, al tener una partición diferente, no comparten ninguna de estas características, salvo con los elementos que están incluidos en ella misma, como son las pestañas incluidas en la capa 2 que muestra esta subventana.

4. Resultado y Tour por la aplicación

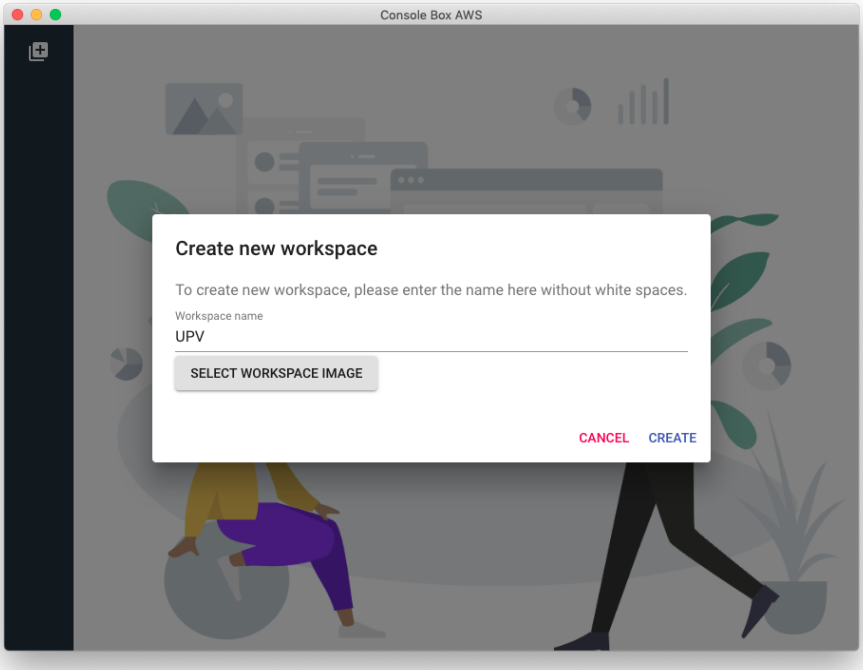
A continuación, se puede ver el flujo normal de la aplicación, así como todas las opciones que presenta, siguiendo el curso de trabajo de un usuario nuevo de la aplicación.

Figure 11

Lo primero que nos encontramos al abrir la aplicación es el preloader. Este se muestra mientras la aplicación carga, permitiendo al usuario saber que la aplicación está funcionando y cargando. El preloader permite mejorar la experiencia del usuario ya que, al percibir algo en vez de nada, mientras la aplicación se carga da una sensación de tiempo de carga menor.

Figure 12

Una vez cargada la aplicación, encontramos la ventana que se muestra en la figura anterior. Al ser el primer uso, no se disponen de espacio de trabajo que seleccionar, por lo que solo podemos crear un espacio de trabajo nuevo.



Al crear el espacio de trabajo aparece una ventana modal pidiendo un nombre de espacio

Figure 13

de trabajo y una imagen. En el tour se crea un espacio de trabajo llamado UPV con la imagen del logo de la universidad.

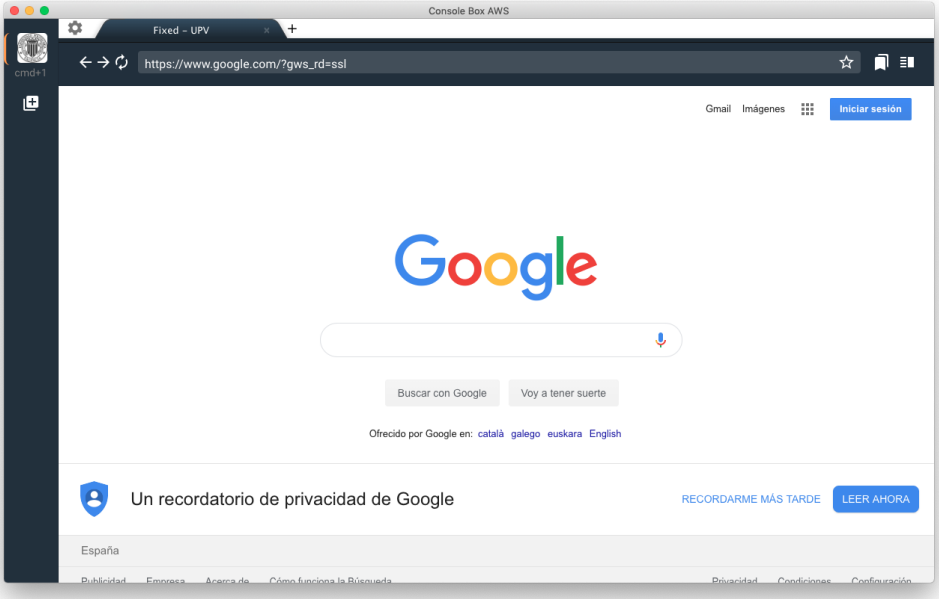


Figure 14

En la imagen se puede ver como se ha creado el espacio de trabajo y se abre la ventana del navegador con una pestaña abierta por defecto.

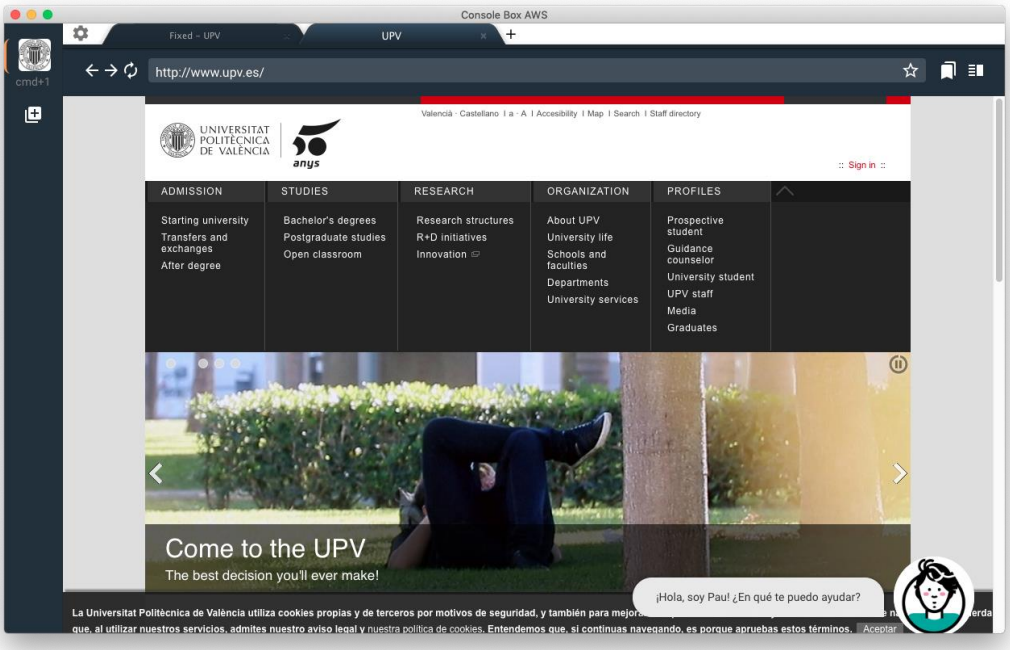


Figure 15

Siguiendo el tour por la aplicacion, se crea una nueva pestaña y se accede a la pagina web de la upv comprobando que el navegador funciona correctamente.

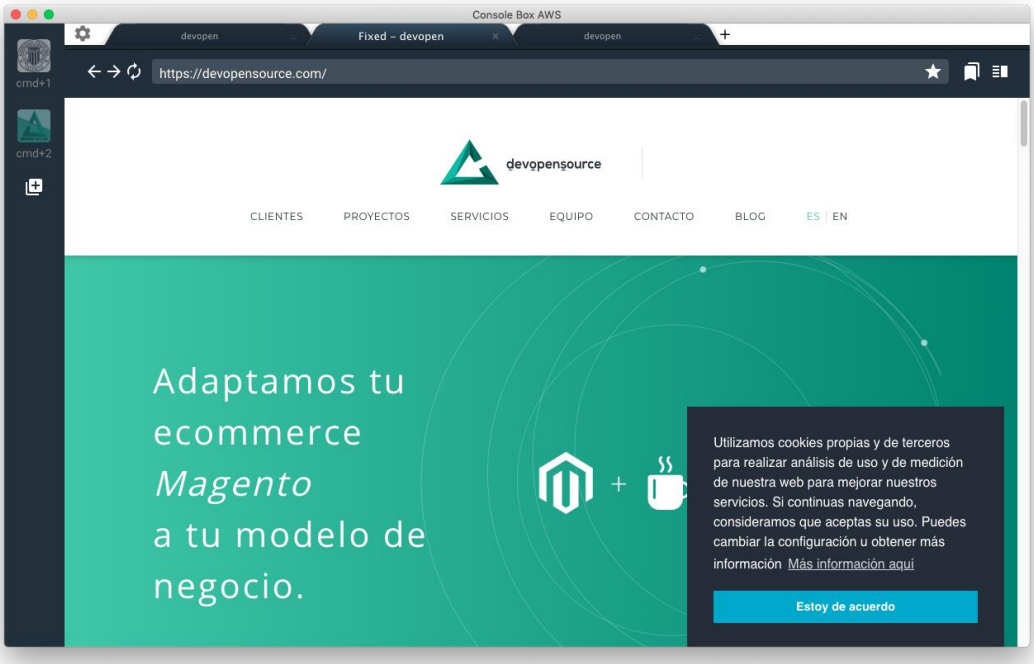


Figure 16

Ahora se crea otro espacio de trabajo y se accede a otra web comprobando que efectivamente tenemos dos veentanas de navegacion, con sus correspondientes pestañas separadas.

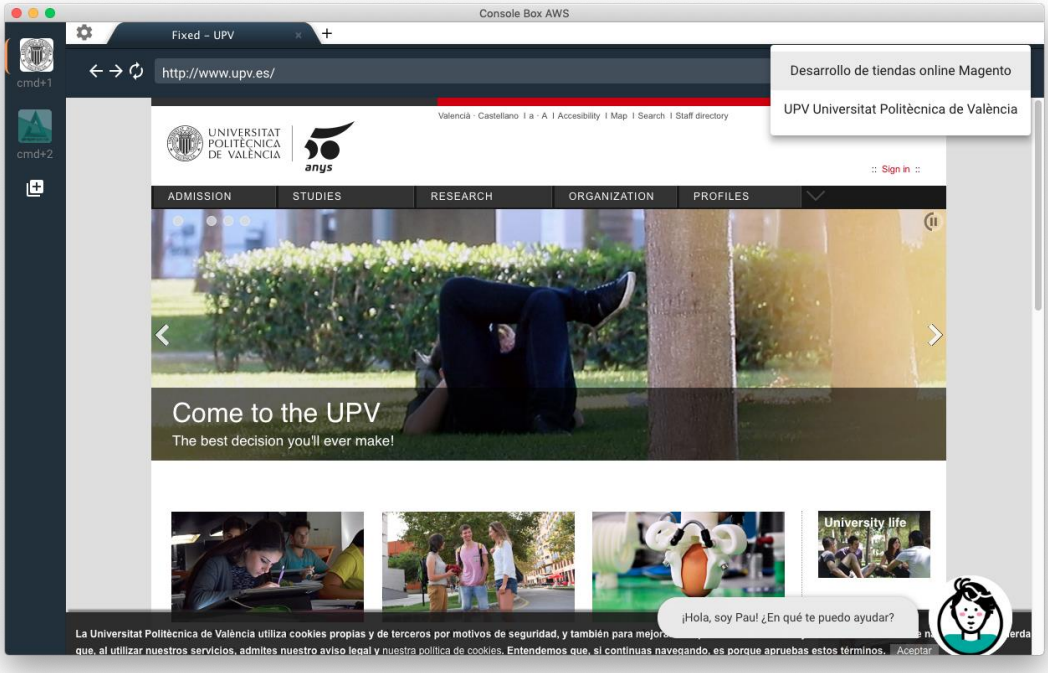


Figure 17

Una vez con los dos espacios de trabajo creados, y habiendo accedido a webs diferentes, se procede a almacenar estas paginas en favoritos, presionando la estrella que aparece al lado de la direccion. Desde cualquiera de los espacios de trabajo, se puede acceder a la lista de urls almacenadas, como se muestra en la imagen anterior.

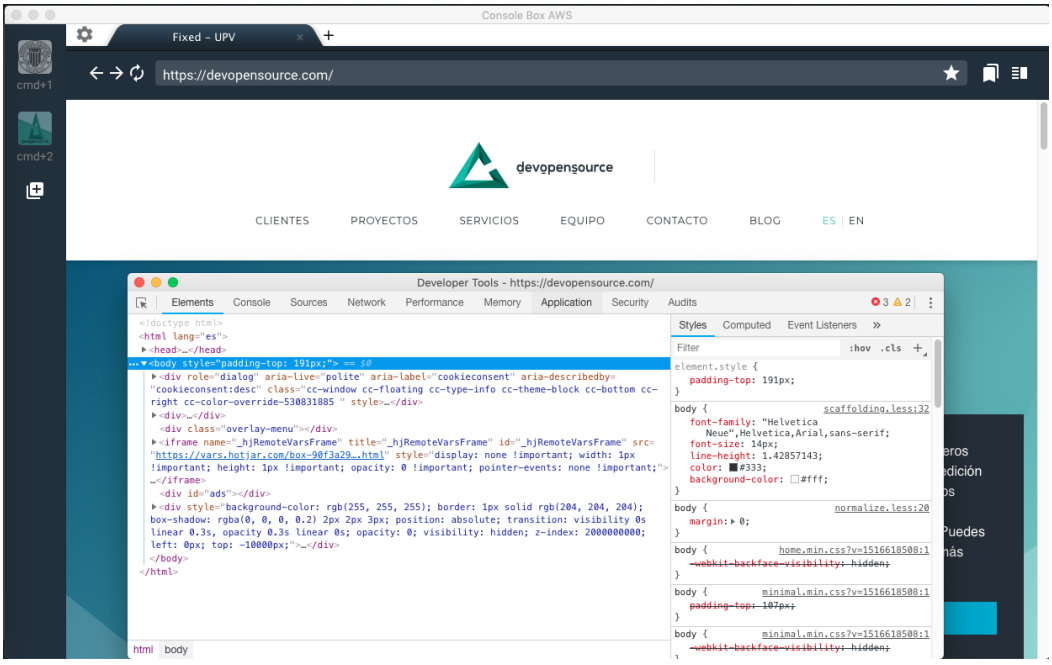


Figure 18

Por ultimo se se muestra la consola de desarrollo donde se puede examinar el codigo de la web que se esta visitando. Esta consola es identica a la del navegador web chrome, pues los dos programas estan basados en chromium.
Como ultimo recurso, se muestra una imagen con seis espacios de trabajo, funcionando todos al mismo tiempo.

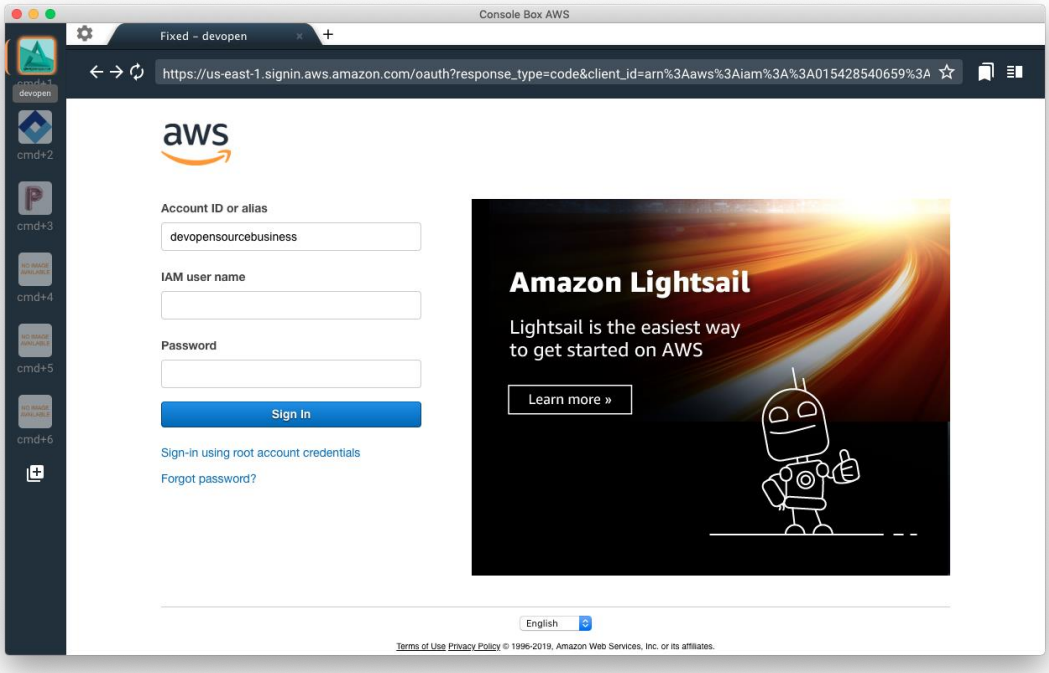


Figure 19

5. Conclusiones y trabajos futuros

Durante la realización de este proyecto se ha intentado conseguir el funcionamiento esencial de la aplicación, sin dejar de lado el diseño y usabilidad de la misma. Si bien, quedaron algunas funciones por implementar la gran mayoría de ellas se han conseguido con éxito, obteniendo resultados mejores a los esperados.

La solución implementada, por el momento, pese a estar en fase de pruebas ya ha conseguido varios de los objetivos propuestos. Varios compañeros de trabajo la utilizan de forma diaria, y aunque la aplicación no está exenta de bugs, sí que han notado una mejora en el flujo de trabajo diario. Además, no solo están contentos con el incremento de rendimiento que les ofrece en la realización de tareas diarias, sino que, además, no paran de pensar y pedir funcionalidades nuevas para mejorar el navegador aún más.

La realización de este trabajo, me ha permitido abrir los ojos a un mundo nuevo, donde con conocimientos básicos sobre tecnologías web, se pueden crear aplicaciones de escritorio muy potentes y de forma muy sencilla, así como de conseguir diseños profesionales. El uso de react, cada vez más conocido y utilizado por los desarrolladores, ofrece una característica esencial para crear y mejorar aplicaciones y es la gran comunidad que hay detrás con guías y soluciones a problemas del día a día, y que, con un poco de esfuerzo permiten el autoaprendizaje. Esto no es solo aplicable a aplicaciones con electrón, pues el conocimiento adquirido sobre react durante la creación de este proyecto es aplicable a futuras aplicaciones y trabajos web.

Como a mejoras y trabajos futuros, cabe resaltar varios de ellos. Entre las mejoras a futuro ya se ha empezado a trabajar en ellas, en las que podemos encontrar: la posibilidad de personalizar los colores del navegador, poder abrir varios espacios de trabajo que se muestren al mismo tiempo, el uso de redux para almacenar datos compartidos en varias capas de la aplicación, la capacidad de aceptar extensiones de Chrome…

Por la parte referente a trabajos futuros, en base a las opiniones de los usuarios se está pensando en adaptar la base de la aplicación para ser compatible con las nuevas tecnologías, así como ofrecer una mejor integración con el sistema operativo. También se está pensando en ofrecer un sistema de idiomas y traducción de la aplicación que funcione por voz y la posibilidad de exportar e importar los favoritos de cualquier navegador a este, lo cual vendría dentro de una redefinición e implementación del sistema actual que por el momento no está visualizado como una prioridad.

6. Bibliografía

React.js - Biblioteca Javascript para crear interfaces de usuario.

https://es.reactjs.org/docs/

Electron.js - Build cross platform desktop apps with JavaScript, HTML, and CSS.

https://electronjs.org/docs

W3schools - The language for building web pages.

https://www.w3schools.com/

Material-ui - React components for faster and easier web development.

https://material-ui.com/

ECMAScript 6 - ES6 new features: Overview & Comparison.

http://es6-features.org

npm - Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

https://www.npmjs.com/

IEEE 830-1984 - IEEE Guide for Software Requirements Specifications.

https://standards.ieee.org/standard/830-1984.html

https://es.wikipedia.org/wiki/Especificaci%C3%B3n_de_requisitos_de_software

Node.js - Entorno de ejecución para JavaScript.

https://nodejs.org/es/

Stateofjs (javascript data) - collected data from over 20,000 developers.

https://stateofjs.com/

7. Anexos

Como parte del anexo se mencionarán diversas cosas que no se han podido añadir en la memoria pero que es interesante hacer mención.

El proyecto está de forma publica en GitHub, y puede ser utilizado por cualquiera. https://github.com/DevopensourceTeam/Console_Box_AWS.

La aplicación se encuentra disponible en el link anterior, todas sus versiones, tanto para MacOS como para Windows. Estas serán descargadas y utilizadas a la hora de realizar la presentación del proyecto.

Por otro lado, actualmente se está realizando un estudio con todos los usuarios de la aplicación a fin de llevar el navegador al siguiente nivel y crear una solución a nivel profesional, opensource y gratuita.