

Equipo Ryan Dahl

Sistemas y Tecnologías Web

12 de Junio del 2020

Descubre Aragón



Eduardo Gimeno - 721615

Gonzalo Berné - 715891

Jorge Fernández - 721529

Joaquín Puyuelo - 604736

Índice

RESUMEN DEL PROYECTO	3
DATOS DEL EQUIPO:	3
LISTADO DE REQUISITOS	4
PROPUESTAS SIMILARES	6
ARQUITECTURA	7
Vista de módulos	7
Vista de componente y conector	8
MODELO DE DATOS	9
API REST	16
IMPLEMENTACIÓN	17
MODELO DE NAVEGACIÓN	20
Pantalla Login	20
Pantalla Registry	21
Pantalla Index-User	22
Pantalla Profile	23
Pantalla Chat-User	24
Pantalla Index-Admin	25
ANALÍTICAS	26
DESPLIEGUE DEL SISTEMA	27
VALIDACIÓN	29
PROBLEMAS ENCONTRADOS DURANTE EL DESARROLLO	30
Problemas con el registro y autenticación de los usuarios	30
Problemas a la hora de importar datos	30
ANÁLISIS DE PROBLEMAS POTENCIALES	33
DISTRIBUCIÓN DE TIEMPO	34
CONCLUSIONES	35
VALORACIÓN PERSONAL DE CADA MIEMBRO	36
ANEXO	37

1. RESUMEN DEL PROYECTO

DATOS DEL EQUIPO:

Nombre: Equipo Ryan Dahl

Integrantes:

Nombre	NIP	Función
Eduardo Gimeno	721615	Back-End
Joaquín Puyuelo	604736	Front-End
Gonzalo Berné	715891	Front-End
Jorge Fernández	721529	Back-End

Repositorio del proyecto: <https://github.com/EduardoGimeno/Sistemas-y-Tecnologias-Web>

Aplicación desplegada: <https://turismoaragon.herokuapp.com/>

Back-End desplegado: <http://back-turismoaragon.herokuapp.com/>

Se ha desarrollado una aplicación orientada al turismo en Aragón, que permita consultar y descargar información sobre agencias de viaje, albergues, refugios, alojamientos de turismo rural, alojamientos hoteleros, apartamentos turísticos, cafeterías y restaurantes, campings turísticos, guías turísticos, oficinas de turismo y puntos de información turística (todos estos datos se denominan entradas). Además, permitirá a los usuarios contactar por e-mail con los dueños de los alojamientos para iniciar un proceso de reserva. Habilitando una zona de chat específica para cada conversación de los usuarios con los dueños.

La información que se va a utilizar en la base de datos de la aplicación pertenece a una página web del gobierno de Aragón denominada Aragón Open Data, la cual es de acceso libre. El cual contiene un banco de datos con una gran cantidad de información sobre temas muy diversos como turismo, deporte, economía, medio ambiente, medio rural, salud, etc. Dicha base de datos es una iniciativa que persigue que los datos y la información, especialmente la que poseen las administraciones públicas, estén disponibles para el conjunto de los ciudadanos. La publicación de sus datos se hace de forma abierta y reutilizable por lo que pueden ser utilizados de forma libre en nuestro proyecto.

El usuario final para el que se planteó la aplicación es cualquier tipo de persona que desee realizar turismo en la comunidad autónoma de Aragón. El funcionamiento básico del sistema será el siguiente: Los usuarios deberán registrarse primero en la aplicación introduciendo algunos datos sobre ellos, que serán su nombre, apellidos, fecha de nacimiento, teléfono de contacto, país de origen, provincia, correo electrónico y una contraseña. Una vez tengan una cuenta propia en el sistema, los usuarios podrán consultar distintos datos de guías, restaurantes, hoteles, etc, que se sitúan en Aragón. Además de poder ponerse en contacto con alojamientos ubicados en las cercanías de los lugares que decidan visitar.

LISTADO DE REQUISITOS

A continuación se puede observar la listado de requisitos iniciales que se planteó previamente al desarrollo del proyecto:

Requisitos funcionales	
Usuario	
ID	Descripción
RFU-1	El sistema permitirá al usuario registrarse
RFU-2	El sistema requerirá al usuario nombre, apellidos, fecha de nacimiento, e-mail, contraseña, teléfono, país y provincia para poder registrarse.
RFU-3	El sistema permitirá al usuario acceder autenticándose mediante e-mail y contraseña
RFU-4	El sistema permitirá al usuario acceder autenticándose utilizando la cuenta de Gmail (autenticación contra Google).
RFU-5	El sistema permitirá al usuario consultar información sobre las entradas.
RFU-6	El sistema permitirá al usuario filtrar las entradas por varios parámetros.
RFU-7	El sistema permitirá al usuario contactar con una entrada de tipo alojamiento mediante un chat.
RFU-8	El sistema permitirá al usuario consultar la ubicación de una entrada en un mapa.
RFU-9	El sistema permitirá al usuario descargar datos de búsquedas en formato CSV y PDF
RFU-10	El sistema permitirá al usuario descargar datos de entradas en formato CSV y PDF
RFU-11	El sistema permitirá al usuario modificar sus datos personales
RFU-12	El sistema permitirá al usuario darse de baja

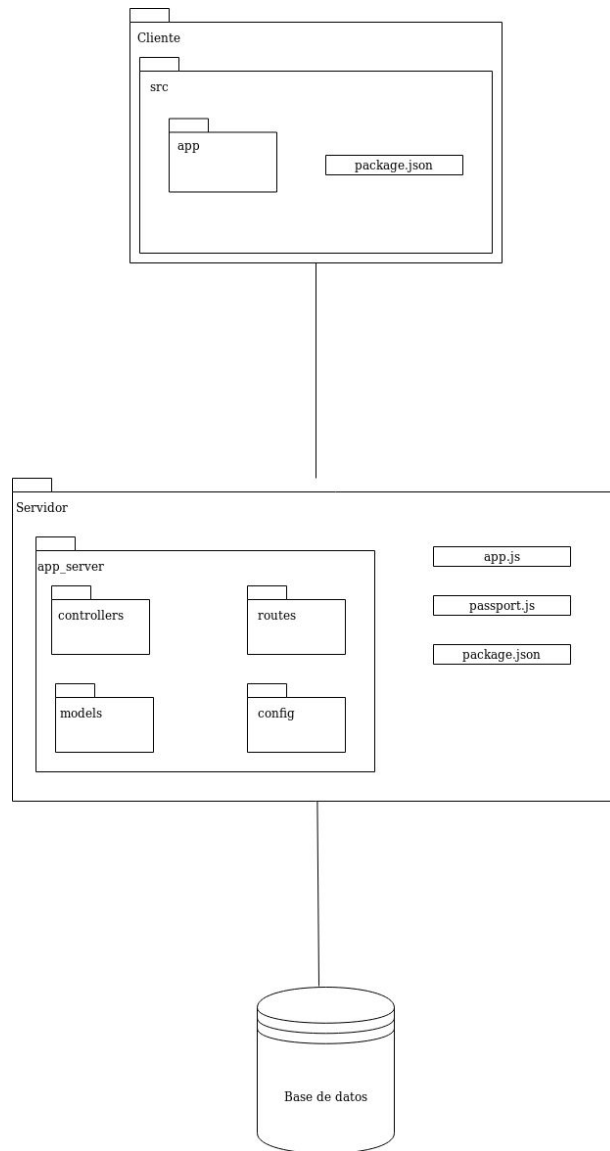
RFU-13	El sistema permitirá al usuario consultar estadísticas sobre municipios y entradas de tipo alojamiento
Administrador	
RFA-1	El sistema permitirá al administrador acceder autenticándose mediante nombre de usuario y contraseña
RFA-2	El sistema permitirá al administrador gestionar a los usuarios (eliminar, banear, enviar e-mail)
RFA-3	El sistema permitirá al administrador consultar estadísticas sobre los usuarios y sobre los datos importados
Sistema	
RFS-1	El sistema permitirá a la gerencia de las entradas de tipo alojamiento responder a los mensajes de los usuarios en un chat
Requisitos no funcionales	
RNF-1	La contraseña tanto del usuario como del administrador deberá tener como mínimo 8 caracteres alfanuméricos
RNF-2	El sistema guardará las contraseñas encriptadas utilizando MD-5
RNF-3	El sistema calculará diariamente las estadísticas mediante procesos en segundo plano
RNF-4	El sistema generará un enlace que enviará por e-mail a la gerencia de una entrada de tipo alojamiento para permitir responder a un cliente
RNF-5	El enlace que permite a la gerencia de una entrada de tipo alojamiento acceder al chat con un cliente tendrá una validez de 72 horas prorrogables
RNF-6	El sistema mostrará las estadísticas al administrador mediante gráficas
RNF-7	El sistema actualizará las entradas mediante procesos en segundo plano

2. PROPUESTAS SIMILARES

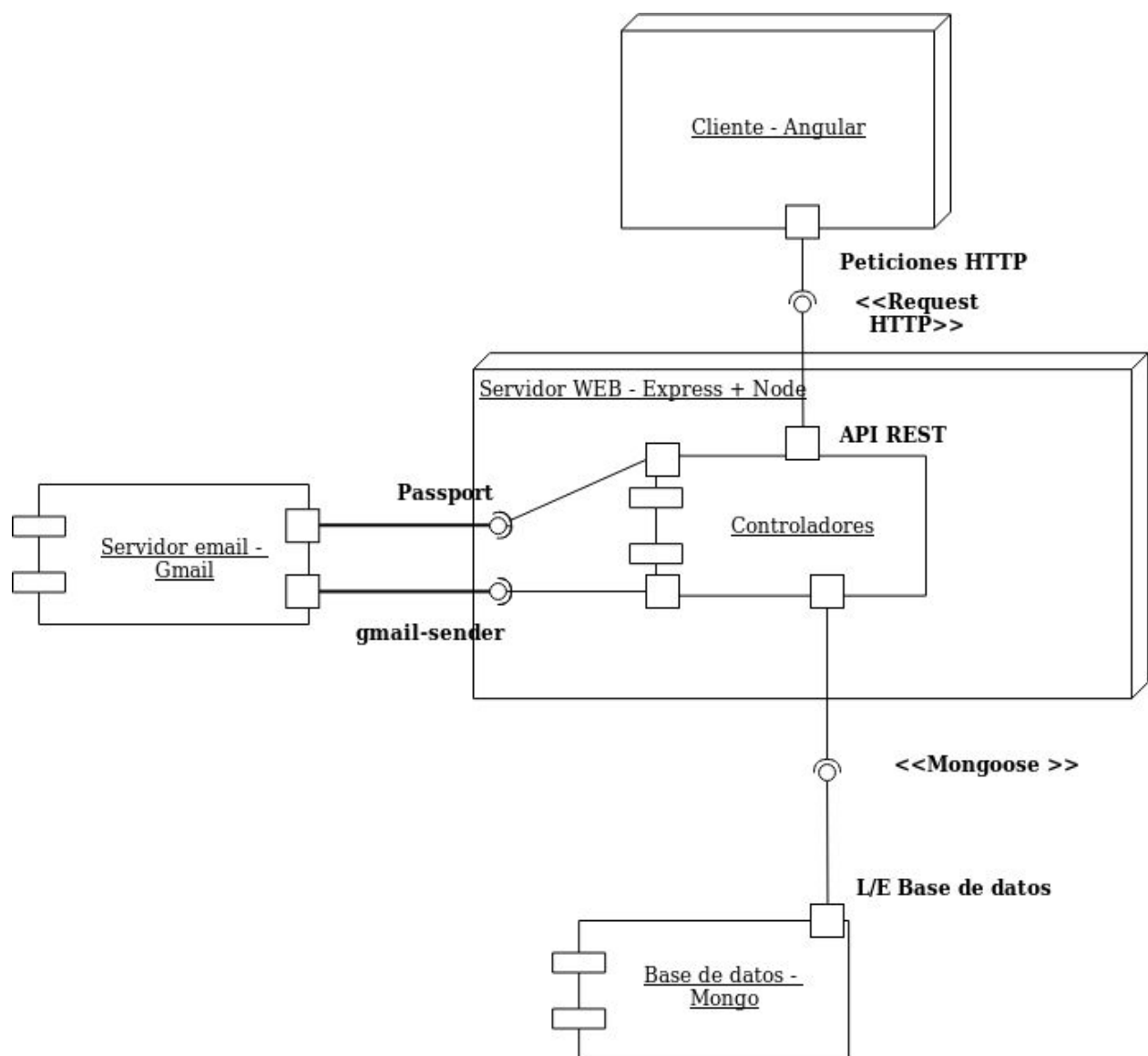
Se buscó información sobre cualquier tipo de aplicación similar para poder realizar una comparativa sobre posibles implementaciones que mejorarían dicha versión, pero no hemos encontrado una aplicación que permita la planificación completa de una visita o unas vacaciones en la comunidad de Aragón, desde el alojamiento hasta las diferentes actividades y sitios de interés de la zona. De esta forma, se considera que la aplicación puede tener éxito ya que rellenará un hueco que actualmente se encuentra libre en la temática del turismo en nuestra comunidad autónoma. Una página web que presentaba una temática parecida que se encontró fue www.turismodearagon.com, aunque su temática es más bien informativa, y presenta una amenaza competitiva para nuestra aplicación. Se conocen también algunas aplicaciones orientadas al turismo de Aragón, como por ejemplo una desarrollada por la empresa Quelinka (<https://www.quelinka.com/app-turismo-de-aragon/>) pero dado que están orientadas de forma total a la oferta de esquí tampoco se consideran posibles alternativas viables al proyecto que se ha desarrollado.

3. ARQUITECTURA

Vista de módulos



Vista de componente y conector



4. MODELO DE DATOS

Se han creado un total de trece esquemas para poder modelar los datos necesarios para la aplicación. Los ficheros creados para los modelos residen en la carpeta models.

El esquema creado para la información de los usuarios, contenido en el fichero usuario.js, contiene trece campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
nombre	String	si	Nombre del usuario
apellidos	String	si	Apellidos del usuario
email	String	si	Email con el que el usuario se ha registrado y accede a la aplicación
fechaNacimiento	Date	no	Fecha de nacimiento del usuario
contrasena	String	no	Contraseña con la que el usuario accede a la aplicación. en caso de que se haya registrado a través de la propia aplicación
telefono	String	no	Teléfono del usuario
país	String	no	País del usuario
provincia	String	no	Provincia del usuario
activo	Boolean	no	Indica si la cuenta está habilitada o no
inicioBan	Date	no	Fecha a partir de la cual el usuario no podrá acceder a su cuenta en caso de ser baneado temporalmente
finBan	Date	no	Fecha a partir de la cual el usuario podrá acceder de nuevo a su cuenta en caso de ser baneado temporalmente
admin	Boolean	no	Indica si el usuario es un usuario normal o administrador

Como se puede observar, hay algunos campos importantes respecto a la información del usuario que no están marcados como requeridos. En el apartado Problemas encontrados durante el desarrollo se explica a que es debido (ver [Problemas con el registro y autenticación de los usuarios](#)).

El esquema creado para la información de los mensajes de los chat, contenido en el fichero mensaje.js, contiene tres campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
emisor	String	si	Emisor del mensaje
texto	String	si	Contenido del mensaje
hora	Date	si	Hora en la que el mensaje ha sido enviado

El esquema creado para la información de los chat, contenido en el fichero chat.js, contiene cinco campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
nomEntrada	String	si	Nombre del alojamiento
nomUsuario	String	si	Nombre del usuario
emailEntrada	String	si	Email del alojamiento
emailUsuario	String	si	Email del usuario
mensajes	Array de esquema de mensajes	no	Mensajes del chat

Para la entradas que son alojamientos, es decir, refugios, alojamientos de turismo rural, hoteles, campings y apartamentos, se ha creado un esquema con los datos que comparten, el cual luego es utilizado por cada uno de ellos en su esquema específico.

Este esquema contiene la información común a todos los alojamientos, contenido en alojamiento.js, contiene diez campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
signatura	String	no	Identificador utilizado por el Gobierno de Aragón
nombre	String	no	Nombre del alojamiento
dirección	String	no	Dirección del alojamiento
codigoPostal	Number	no	Código postal del alojamiento
provincia	String	no	Provincia donde se encuentra el alojamiento
comarca	String	no	Comarca donde se encuentra el alojamiento
municipio	String	no	Municipio donde se encuentra el alojamiento
capacidad	Number	no	Capacidad del alojamiento
email	String	no	Email de contacto del alojamiento
teléfono	String	no	Teléfono del alojamiento

Al igual que ocurre en el esquema de usuario, hay campos que no son requeridos, pese a que son importantes, también ocurre en los esquemas del resto de entradas. En el apartado Problemas encontrados durante el desarrollo se explica a que es debido (ver [Problemas a la hora de importar datos](#)).

El esquema creado para la información de los alojamientos de turismo rural, contenido en el fichero alojamientoTurismoRural.js, contiene tres campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
común	Esquema de alojamiento	no	Datos comunes de todos los alojamientos
espigas	Number	no	Categoría del alojamiento de turismo rural

tipo	String	no	Tipo de alojamiento de turismo rural (casa, apartamento, ...)
------	--------	----	---

El esquema creado para la información de los apartamentos, contenido en el fichero apartamento.js, contiene un campo, el cual se describen a continuación:

Nombre	Tipo	Requerido	Descripción
común	Esquema de alojamiento	no	Datos comunes de todos los alojamientos

El esquema creado para la información de los campings, contenido en el fichero apartamento.js, contiene un campo, el cual se describen a continuación:

Nombre	Tipo	Requerido	Descripción
común	Esquema de alojamiento	no	Datos comunes de todos los alojamientos

El esquema creado para la información de los hoteles, contenido en el fichero hotel.js, contiene tres campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
común	Esquema de alojamiento	no	Datos comunes de todos los alojamientos
grupo	String	no	Empresa a la que pertenece el hotel
estrellas	Number	no	Categoría del hotel

El esquema creado para la información de los refugios, contenido en el fichero apartamento.js, contiene un campo, el cual se describen a continuación:

Nombre	Tipo	Requerido	Descripción
común	Esquema de alojamiento	no	Datos comunes de todos los alojamientos

El esquema creado para la información de las oficinas de turismo, contenido en el fichero oficinaTurismo.js, contiene siete campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
signatura	String	no	Identificador utilizado por el Gobierno de Aragón
nombre	String	no	Nombre de la oficina de turismo
dirección	String	no	Dirección de la oficina de turismo
provincia	String	no	Provincia donde se encuentra la oficina de turismo
municipio	String	no	Municipio donde se encuentra la oficina de turismo
teléfono	String	no	Teléfono de la oficina de turismo
horario	String	no	Horario de la oficina de turismo

El esquema creado para la información de los puntos de información, contenido en el fichero puntoInformacion.js, contiene cinco campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
signatura	String	no	Identificador utilizado por el Gobierno de Aragón
nombre	String	no	Nombre del punto de información

dirección	String	no	Dirección donde se encuentra el punto de información
provincia	String	no	Provincia donde se encuentra el punto de información
municipio	String	no	Municipio donde se encuentra el punto de información

El esquema creado para la información de los restaurantes, contenido en el fichero restaurante.js, contiene diez campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
signatura	String	no	Identificador utilizado por el Gobierno de Aragón
nombre	String	no	Nombre del restaurante
dirección	String	no	Dirección donde se encuentra el restaurante
codigoPostal	Number	no	Código postal del restaurante
provincia	String	no	Provincia donde se encuentra el restaurante
comarca	String	no	Comarca donde se encuentra el restaurante
municipio	String	no	Municipio donde se encuentra el restaurante
capacidad	Number	no	Capacidad del restaurante
teléfono	String	no	Teléfono del restaurante
categoría	Number	no	Categoría del restaurante

El esquema creado para la información de los guías, contenido en el fichero guia.js, contiene diez campos, los cuales se describen a continuación:

Nombre	Tipo	Requerido	Descripción
signatura	String	no	Identificador utilizado por el Gobierno de Aragón
nombre	String	no	Nombre del guía
apellidos	String	no	Apellidos del guía
teléfono	String	no	Teléfono del guía
español	Boolean	no	El guía puede hablar español
inglés	Boolean	no	El guía puede hablar inglés
francés	Boolean	no	El guía puede hablar francés
alemán	Boolean	no	El guía puede hablar alemán
italiano	Boolean	no	El guía puede hablar italiano
Otros	Boolean	no	El guía puede hablar otros idiomas diferentes a los anteriores

5. API REST

No se ha podido documentar la API con Swagger por falta de tiempo.

6. IMPLEMENTACIÓN

Estado actual de la aplicación

Debido a la alta carga de trabajo de las últimas semanas, nos vimos obligados a pedir una prórroga para poder entregar algo presentable. Por ello, los últimos días hemos trabajado en equipo una gran cantidad de horas con el objetivo de acabar la aplicación, aún así, somos conscientes de que no ha sido suficiente ya que aspectos que consideramos importantes no han sido implementados o al menos no totalmente.

El punto más importante que no se ha podido realizar ha sido el tema de las estadísticas. No hemos tenido tiempo para mirar los módulos necesarios ya que hemos considerado que era un añadido extra a la aplicación y había otros puntos más relevantes.

La autenticación contra Gmail también nos ha dado varios problemas y en la versión final no hemos sido capaces de integrar completamente, también, las acciones que en un principio podían realizar los administradores sobre los usuarios se han reducido.

El mayor punto de fallo ha sido la interacción con MongoDB, hemos tenido muchos problemas que nos han consumido mucho tiempo y algunos de ellos no hemos sido capaces de solucionarlos, el ejemplo más claro es al realizar búsquedas con filtros en las “entradas” ya que estas búsquedas funcionan perfectamente con los hoteles y sin embargo con el resto no (habiendo utilizado la misma implementación para todos).

Hay dos aspectos que consideramos importantes y que se han implementado parcialmente ya que están preparados en el BackEnd pero no se han llegado a conectar desde el FrontEnd. Se ha automatizado la actualización de los datos de las “entradas” en la base de datos (parserData) descargando los json que se utilizan en la aplicación desde la web de open data Aragón y también está habilitada la descarga de datos en CSV desde el BackEnd a partir de un JSON.

Sin embargo, sí que se ha llegado a implementar totalmente varios aspectos que se consideran importantes como las conversaciones entre los usuarios y los administradores de las “entradas”, la utilización de Maps de Google y el uso de jwt para la autenticación.

A continuación, se van a resumir algunos aspectos que se consideran de importancia en la implementación tanto del Front-End como del Back-End.

Front-End

En líneas generales, las distintas pantallas del front-end fueron implementadas generando componentes con el comando: `ng generate component`. Una vez creadas se enlazaban creando su enrutamiento en el archivo `app-routing.module`. Se utilizaron las librerías Geocoding API y Maps

JavaScript API en el componente del front entry. El cual está destinado a mostrar toda la información de una de las entradas que aparecen como resultado de una búsqueda de un usuario. Para poder hacer uso de ellas fue necesario realizar un largo proceso en el cual primero se creó una cuenta en Google Cloud Platform. En la creación de dicha cuenta, fue necesario aportar la tarjeta de crédito de uno de los componentes del equipo para así recibir acceso a una prueba gratuita con una duración de 1 año y 300\$ de créditos promocionales. Una vez creada dicha cuenta, se pudieron activar dichos servicios y haciendo uso de una API-Key suministrada se pudo hacer uso de estas librerías. La Maps API se utilizó para poder mostrar mapas en las entradas. A esta se le unió la Geocoding API la cual permite introducir la dirección de una calle en concreto y a partir de esta sacar las coordenadas de latitud y longitud de la misma. Todo ello junto, hizo posible que al introducir la dirección de uno de los lugares almacenado en nuestra base de datos, se muestre su ubicación exacta en un mapa de Google Maps. En el componente profile y en las pantallas de registro se puede elegir entre distintos países de la unión europea y sus respectivas provincias. Esta funcionalidad se implementa utilizando la librería npm: country-state-city. La cual realiza una búsqueda en una base de datos con todas las ciudades, provincias y países.

Back-End :

En relación al backend, se ha seguido una arquitectura basada en modelos y controladores de los diferentes tipos de documentos almacenados por nuestra base de datos. De esta manera, los diversos modelos contenían las definiciones de los diversos documentos existentes, mientras los controladores contenían diversas funciones que permiten trabajar con los datos. Además, aparece un nuevo elemento, que son los routers, que ejercen como middlewares para encaminar las peticiones hacia las funciones que cubren las mismas, de esta manera, las diversas rutas tienen asignada una función que será ejecutada en caso de acceder a esa ruta. Por otro lado, las funciones definidas en los diversos controladores, al contar con consultas a la base de datos MongoDB, debido a que se debe esperar a la respuesta del nodo correspondiente a la base de datos, se debe incluir la cláusula await en todas las peticiones. Esta cláusula await implica que la función que lo utiliza pasa a ser asíncrona, por lo cual también se debe definir la función como async.

Por otro lado, se han utilizado diversos módulos externos para apoyar la realización de una serie de tareas, como son jsonwebtoken, que permite firmar, verificar y descifrar los tokens que se intercambian con la aplicación web. Estas funciones se encuentran en un fichero a parte de los modelos, controladores y rutas.

Otro de los módulos utilizados ha sido passport, del cual se usa el módulo passport-google-oauth para la autenticación con Google, en concreto utilizando el método OAuth2Strategy para tal fin, este método se compone de una función principal adherida a la variable passport, a la cual indicamos las claves de Google asociadas, y la dirección del callback, que ejecutará la función siguiente, en la cual se procederá, si procede, a almacenar al usuario.

De manera similar ocurre con el módulo passport-json, que permite autenticar usuarios mediante un fichero JSON, en este caso, la configuración básica se realiza indicando cuales van a ser los nombres

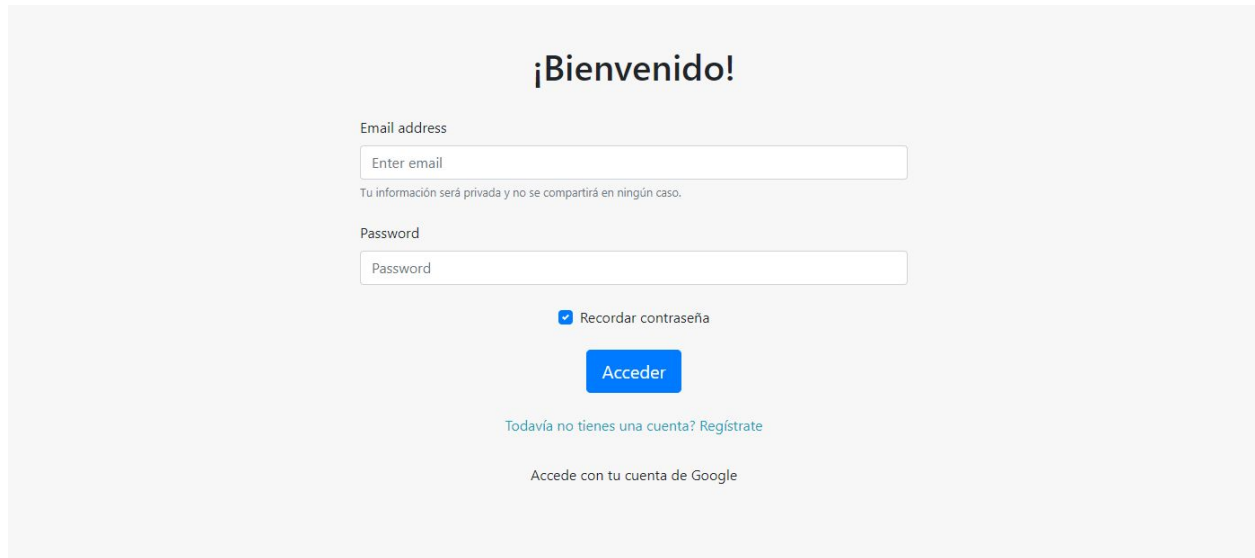
de los campos correspondientes al usuario y a la contraseña respectivamente. En este caso, únicamente existe un fichero de router que permite autenticar de determinada manera dependiendo la petición recibida.

Por último, otro modo que se ha utilizado es jsonexport, que es utilizado para generar un fichero csv a partir de un JSON.

7. MODELO DE NAVEGACIÓN

En esta sección se mostrarán las pantallas del cliente de la aplicación, explicando el funcionamiento de cada una de ellas, su utilidad y la forma de navegar entre ellas. La primera de ellas será el Login.

Pantalla Login



A mockup of a login screen with a light gray background. At the top center, the text '¡Bienvenido!' is displayed in a bold, black, sans-serif font. Below this, there are two input fields. The first is labeled 'Email address' and contains the placeholder text 'Enter email'. Below the email field, a small line of text reads 'Tu información será privada y no se compartirá en ningún caso.' The second input field is labeled 'Password' and contains the placeholder text 'Password'. Below the password field, there is a checkbox with a blue checkmark and the text 'Recordar contraseña'. Centered below the checkbox is a blue button with the white text 'Acceder'. Below the button, there is a link that reads 'Todavía no tienes una cuenta? Regístrate' in a teal color. At the bottom center, there is a text link that reads 'Accede con tu cuenta de Google'.

Esta es la primera pantalla que aparece al acceder a la aplicación. Desde ella los usuarios accederán a su sección personal de la aplicación, y en caso de que no tengan una cuenta podrán registrarse haciendo click en el enlace que pregunta si aún no posees una cuenta. Además se ha implementado la posibilidad de que los usuarios accedan con su cuenta de Google.

Pantalla Registry

Regístrate en Descubre Aragón

Nombre	Apellidos
<input type="text"/>	<input type="text"/>
Fecha de nacimiento	Teléfono
<input type="text"/>	<input type="text"/>
País	Provincia
<input type="text"/>	<input type="text"/>
Correo Electrónico	Tu información será privada y no se compartirá en ningún caso.
<input type="text" value="Introduzca su e-mail"/>	
Contraseña	Confirmar contraseña
<input type="password"/>	<input type="password"/>

Confirmar Registro

En esta sección los usuarios podrán registrarse en la aplicación introduciendo los datos solicitados. Y una vez se complete el registro (en caso de que todos los datos se introduzcan correctamente, se redirigirá al nuevo usuario a su sección personal de la aplicación, al igual que ocurre cuando este accede a la aplicación desde el login del sistema.

Completa tu registro en Descubre Aragón

Faltan los siguientes campos de añadir para completar su registro:

Fecha de nacimiento	Teléfono
<input type="text"/>	<input type="text"/>
País	Provincia
<input type="text" value="España"/>	<input type="text" value="A Coruna"/>

Confirmar Registro

En el caso de que el acceso a la aplicación se realice mediante un login con una cuenta de gmail del usuario, se pedirán algunos datos extra que faltan, para así completar con éxito el registro del usuario. Para ello se ha implementado una pantalla alternativa de registro, en la cual se dispone tan sólo de 4 campos que rellenar.

Pantalla Index-User

Nombre Apellidos
España, Zaragoza

Inicio

Buscar

Hoteles

[1]

Provincia: Todas

Comarca: Todas

Municipio: Todos

Estrellas
Minimo: 1 (Una) Máximo: 5 (Cinco)

Idioma (Guías turísticas): Todos

Buscar

Esta es la pantalla a la que se accede al hacer tanto login en la aplicación como registrarse en ella. Como se puede apreciar está compuesta por tres secciones principales. A la izquierda se dispone de una serie de botones con los que poder navegar entre las distintas secciones de la aplicación. El botón que se encuentre resaltado en azul indica en todo momento la sección en la que uno se encuentra. Los usuarios disponen de una pantalla principal, un perfil en el que pueden realizar cambios, una sección de conversaciones para hablar con los propietarios de los alojamientos que visiten y una sección de estadísticas. En la columna de enmedio se mostrarán los resultados de las búsquedas que se realicen. En este caso aún no se ha llevado a cabo ninguna búsqueda. Estas búsquedas se realizan en la sección de más a la derecha de la pantalla, donde se puede observar los distintos filtros disponibles para hacer las búsquedas (provincia, comarca, municipio, estrellas del lugar e idioma de los guías turísticos disponibles). Por último, cabe destacar que en todas las pantalla se dispone del menú de la parte superior, que indica la ventana en la que uno se encuentra actualmente, además de una breve descripción con algunos datos del usuario en la esquina superior izquierda.

Pantalla Profile

Nombre Apellidos
España, Zaragoza

Perfil Personal

Bienvenido, Nombre Apellidos

País: Espana Provincia: A Coruna

Email: nombre@apellidos.com

[Guardar cambios](#) [Cambiar contraseña](#)

Esta es la sección del perfil de cada usuario, donde pueden realizar algunos cambios a su información personal. Estos serían el país al que pertenece, su provincia correspondiente y su correo electrónico. La búsqueda de los países y sus provincias se ha implementado mediante el uso de una librería externa que se encontró en npm: country-state-city

Mediante el uso de dicha librería se pudo implementar la posibilidad de elección de los países de la unión europea junto con sus provincias correspondientes. Además, se cuenta con la posibilidad de cambiar la contraseña pulsando en el botón “Cambiar contraseña” mediante el cual se accede a una pantalla diseñada a tal efecto. Se muestra a continuación una captura de esta última (su estructura es similar a la de profile pero cambiando los contenidos de la columna central):

Nombre Apellidos
España, Zaragoza

Cambiar Contraseña

Bienvenido, Nombre Apellidos

Introduzca su contraseña actual

Contraseña actual:

Introduzca su nueva contraseña

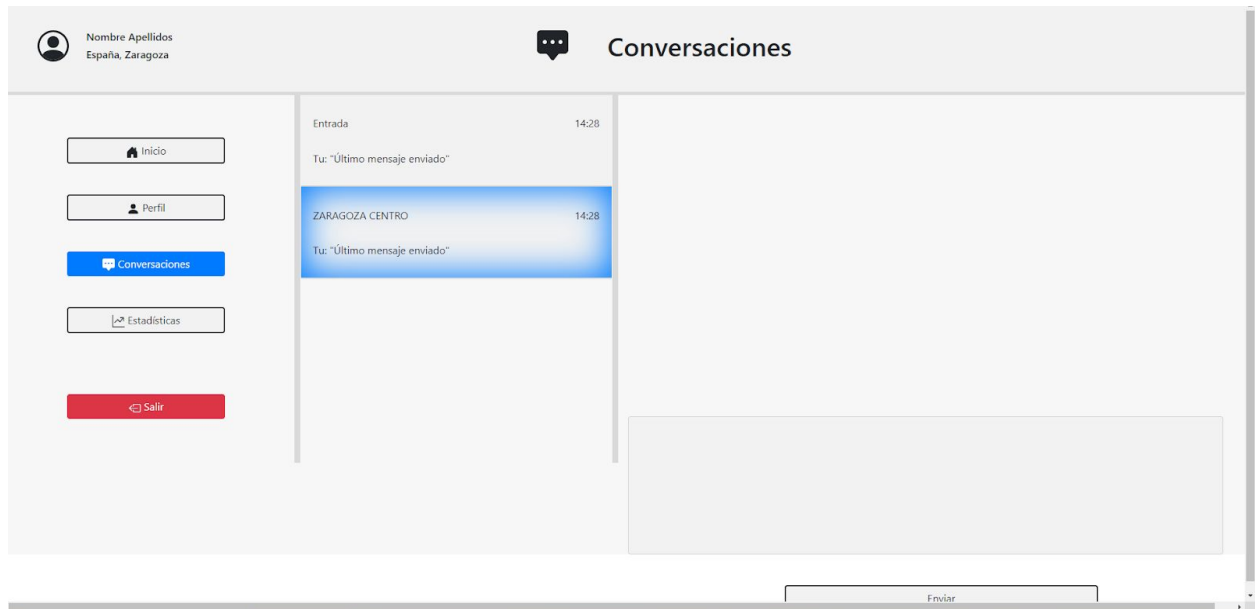
Nueva contraseña:

Confirme su nueva contraseña

Repetir la contraseña nueva:


[Guardar cambios](#)


Pantalla Chat-User




Esta es la ventana que aparece cuando los usuarios pulsan el botón de conversaciones. Desplegándose en la columna central las distintas conversaciones activas que tienen. Además cuando se selecciona una de ellas se abre un chat en la tercera columna con la conversación seleccionada.

Pantalla Index-Admin

 Administrador

 Inicio

 Buscar

Inicio

Estadísticas

Salir

Nombre Apellidos

E-mailProvinciaPaís→

Nombre Apellidos

E-mailProvinciaPaís→

Nombre Apellidos

E-mailProvinciaPaís→

Nombre Apellidos

E-mailProvinciaPaís→

Nombre

Apellidos

E-mail

Buscar

Esta pantalla posee de toda la funcionalidad de la que disponen los administradores de la aplicación. A la cual, los administradores acceden a través de la misma ventana de login que los usuarios.

8. ANALÍTICAS

No se ha podido realizar este apartado por falta de tiempo.

9. DESPLIEGUE DEL SISTEMA

Para realizar el despliegue, se ha decidido realizar el despliegue en 3 máquinas diferentes, una para cada nodo necesario para nuestro sistema (aplicación, servidor web y base de datos).

En el nodo que almacena la base de datos de nuestro sistema, para poder realizar el despliegue de una base de datos MongoDB, se ha necesitado incluir en la máquina el add-on mLab-MongoDB, una plataforma ofrecida como Database-as-a-Service que permite desplegar esta base de datos y utilizarla como servicio para nuestra aplicación. Gracias a la facilidad que proporciona mongoose para realizar la conexión con la base de datos, únicamente nos es necesario conocer la URI de la misma para poder utilizarla.



The screenshot shows the 'Application Logs' for a Heroku application. The logs detail the deployment of a MongoDB add-on. The process starts with creating release v1, followed by an initial release, then release v2. It then shows the enablement of Logplex, attaching the MONGODB add-on (ref: mongolab-parallel-99697), running release v3 commands, and finally creating release v4. The last log entry indicates that the provisioning for the add-on is completed and the MONGODB_URI is set.

```
2020-06-10T08:38:09.536149+00:00 app[api]: Release v1 created by user 721615@unizar.es
2020-06-10T08:38:09.536149+00:00 app[api]: Initial release by user 721615@unizar.es
2020-06-10T08:38:09.666453+00:00 app[api]: Release v2 created by user 721615@unizar.es
2020-06-10T08:38:09.666453+00:00 app[api]: Enable Logplex by user 721615@unizar.es
2020-06-10T08:40:54.551416+00:00 app[api]: Attach MONGODB (@ref:mongolab-parallel-99697) by user 721615@unizar.es
2020-06-10T08:40:54.551416+00:00 app[api]: Running release v3 commands by user 721615@unizar.es
2020-06-10T08:40:54.562022+00:00 app[api]: Release v4 created by user 721615@unizar.es
2020-06-10T08:40:54.562022+00:00 app[api]: @ref:mongolab-parallel-99697 completed provisioning, setting MONGODB_URI. by user 721615@unizar.es
```

Autoscroll with output [Save](#)

Como se puede observar, los pasos seguidos para el despliegue de este nodo del sistema fueron muy simples: primero se realizó la creación de la máquina para después incluir en la misma el add-on comentado anteriormente que proveía a la misma de una base de datos MongoDB.

Para los dos nodos restantes de nuestro sistema, se ha utilizado la misma estrategia en ambos, la cual se basa en que primero Heroku trata de encontrar un fichero Procfile y, en caso de no encontrarlo, como en este caso en el cual ese manejo se realiza mediante el fichero package.json, trata de generar un script de nombre web en función del script de start definido en el package.json. Es importante también la presencia en el package.json del dato engines, en el cual se va a especificar la versión de node que va a ser utilizada para el despliegue del repositorio en la máquina.



The screenshot shows the 'web' process view in Heroku. The command 'npm start' is displayed. On the right, there is a toggle switch, the price '\$0.00', and an edit icon.

```
web npm start
```

En la imagen podemos ver la existencia del proceso web, en base a lo definido en el package.json.

Otra función ofrecida por Heroku, es la de definición de variables de entorno, `config_var` para Heroku, que en nuestro caso se han utilizado para definir las variables relacionadas con las claves necesarias tanto para la autenticación con Passport y la integración de Google con el mismo, como para la clave secreta utilizada para la firma del JWT. De esta manera, podemos referenciar en el código fuente el uso de estas variables de la siguiente manera:

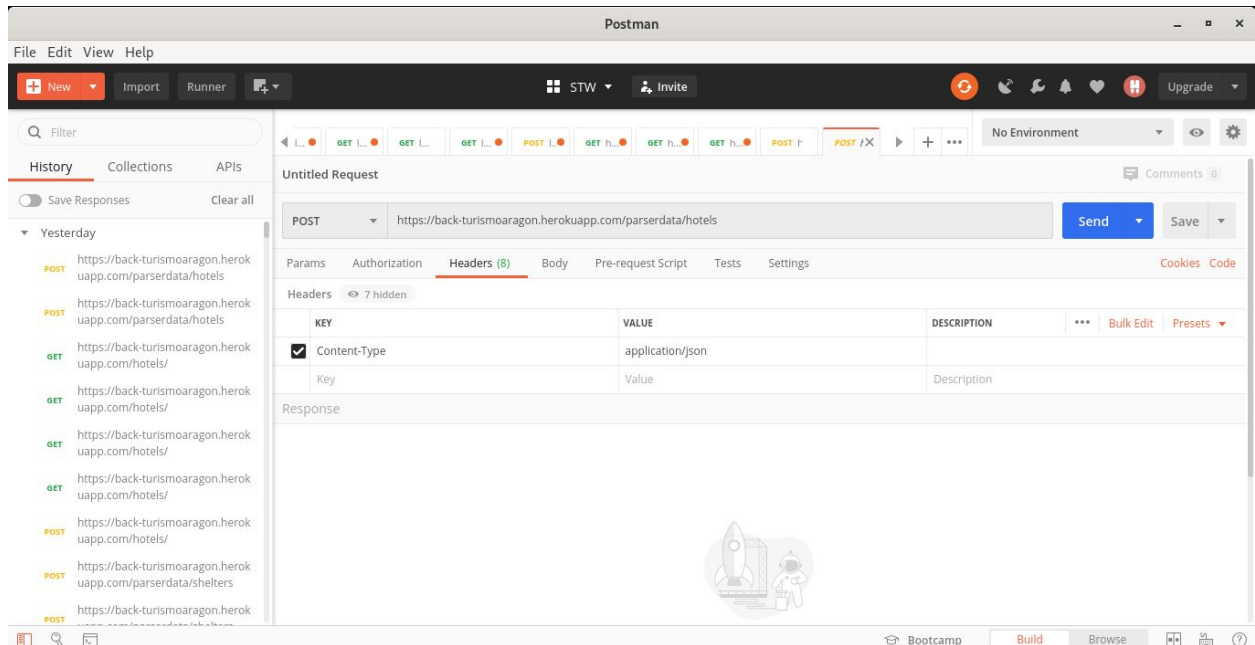
```
var clienteID = process.env.CLIENTE_ID  
  
var secretID = process.env.SECRET_ID  
  
var jwtClave = process.env.JWT_CLAVE
```

En cuanto al método utilizado para el despliegue en Heroku, se ha optado por el uso de Heroku CLI, sin embargo debido a que ya existía previamente un repositorio en el cual se almacenaban los códigos fuente, en vez de realizar el clonado del repositorio de git local generado por Heroku, se ha añadido el mismo como repositorio remoto del ya existente, de manera que al ejecutar `git push heroku master`, se realice el push al repositorio de Heroku y se realice un nuevo deployment de la aplicación.

10. VALIDACIÓN

Por parte del front-end, no se ha podido llevar a cabo la validación E2E por falta de tiempo.

Por parte del back-end, para probar las distintas operaciones de los controladores, se ha utilizado Postman, tanto cuando se estaba desplegando en local el servidor y la base de datos, como una vez desplegados en Heroku.



A la hora de realizar integración entre el cliente y el servidor se ha utilizado principalmente la consola de desarrollador del navegador para ver el estado de cada petición desde la vista del cliente.

11. PROBLEMAS ENCONTRADOS DURANTE EL DESARROLLO

Problemas con el registro y autenticación de los usuarios

En el desarrollo del registro y autenticación de usuarios mediante Passport y en concreto la posibilidad de realizar este procedimiento utilizando una cuenta de Google ha generado diversos problemas.

El primero de ellos consistió en que dado que para poder realizar esta autenticación lo primero que se debe realizar es generar una serie de credenciales para la aplicación en Google, en la creación de estas llega una parte en la cual se pregunta por cuáles van a ser las direcciones que van a poder realizar este tipo de peticiones con esas credenciales, dando a un fallo en caso de que la petición recibida no estuviera incluida dentro del rango de direcciones. Situación similar ocurría con el callback de la misma.

Por otro lado, una vez solucionados esos problemas básicos y consiguiendo la funcionalidad completa del módulo Passport, nos dimos cuenta que la información a la cual se permite acceder utilizando este método en comparación con los modelos definidos previamente en nuestra aplicación que contenían una serie de parámetros obligatorios, era inferior, por lo cual se tuvieron que modificar los modelos de modo que los únicos campos obligatorios fueran aquellos comunes a los dos métodos posibles de autenticación, con Google y mediante Formulario de registro. Esto resultó en la necesidad desde la aplicación web, de realizar un formulario complementario en el caso del registro mediante Google que permitiera recolectar la información necesaria para complementar la información básica.

Problemas a la hora de importar datos

A la hora de realizar la importación de los datos disponibles en Aragón Open Data, se tenía como idea inicial realizar inicialmente una actualización de los datos que ya existieran en nuestro sistema, y la introducción de los nuevos datos. Sin embargo, debido a que surgieron problemas con la búsqueda de los datos actuales con el fin de actualizarlos posteriormente, se tomó la decisión de realizar un borrado completo de los documentos y un nuevo poblado del modelo. Además, apareció un problema extra, dado que un pequeño número de datos necesarios en nuestra aplicación como datos de tipo entero, aparecían en la fuente de datos como dato null, por lo tanto se debió realizar una modificación en los modelos, restringiendo el número de datos requeridos de manera obligatoria. Además de este pequeño problema, la fuente de datos también tenía añadido otro problema, y es que el formato del fichero JSON que proporcionaba no coincide con el formato JSON que se llevaba trabajando en la aplicación durante todo el desarrollo del proyecto, y hacía mucho más complicado el acceso a los datos, debido a esto se tuvo que crear una función auxiliar que permitiera parsear los datos fuente a el mismo formato utilizado en nuestro servidor web:

```
/*  
* Función utilizada para obtener un JSON bien estructurado
```

```
* con el que poder extraer los datos más fácilmente
*/

test = function(datos) {

    let ejemplo = datos;

    let claves = ejemplo[0];

    let retVal = [];

    ejemplo.shift();

    ejemplo.forEach(function(item, index) {

        let aux = {};

        claves.forEach(function(jsonitem, jsonindex) {

            if (item[jsonindex] == null) {

                aux[jsonitem] = "";

            } else {

                aux[jsonitem] = item[jsonindex];

            }

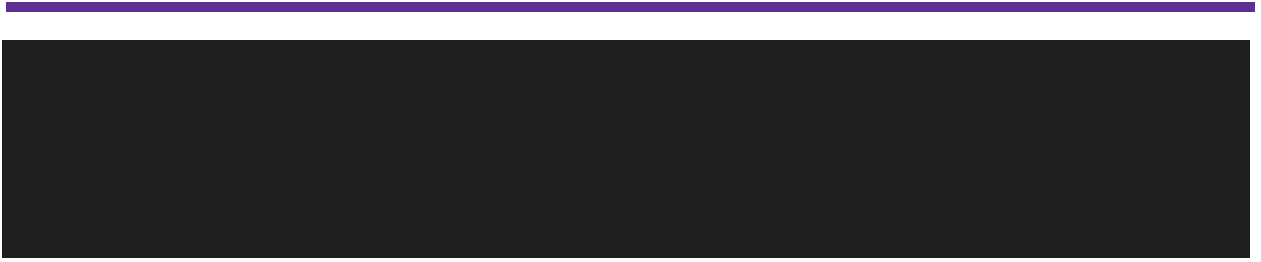
        });

        retVal.push(aux);

    });

    return retVal;

}
```



12. ANÁLISIS DE PROBLEMAS POTENCIALES

En relación al servidor web de nuestra aplicación, el primero problema potencial que se nos ocurre es, debido al uso de tokens, que ocurre si estos tokens acaban siendo interceptados, dado que existen un pequeño número de peticiones en las cuales el token se utiliza como parámetro para poder obtener determinada información. Ante este hecho, se han definido todos aquellos puntos de entrada que no son los estrictamente necesarios para el acceso al sistema, para que antes de realizar ninguna tarea comprueben que se ha enviado el token identificativo en la petición y que este sea válido, una vez realizada esta comprobación se procederá a ejecutar la operación, en caso contrario se informará de que el token enviado no es correcto. De esta manera se cubre uno de los principales problemas potenciales.

Un problema potencial derivado del anterior, es referente a los tiempos de expiración que se incluyen en los tokens

Otro gran problema en torno al servidor web, son las NoSQL Injections que puede sufrir nuestra base de datos, este tipo de problemas si existe actualmente en nuestra implementación, dado que por ejemplo procedemos a realizar consultas en la base de datos con los mismos datos que recibe la petición, sin realizar ninguna comprobación de los mismos.

13. DISTRIBUCIÓN DE TIEMPO

Se ha llevado a cabo un control de esfuerzos, junto con las tareas dedicadas en una hoja excel, debido a que no cabe bien en el documento, se encuentra disponible en <https://github.com/EduardoGimeno/Sistemas-y-Tecnologias-Web/blob/master/Documentacion/Control/Esfuerzos.xlsx>.

Se ha utilizado la página <https://monday.com/lang/es/> para crear un tablero al que agregar las tareas de cada parte del sistema, asignándole un miembro del equipo encargado de llevarla a cabo, así como una fecha límite y una prioridad. Se ha podido exportar el tablero a una hoja excel, la cual se encuentra disponible en <https://github.com/EduardoGimeno/Sistemas-y-Tecnologias-Web/blob/master/Documentacion/Control/Tablero.xlsx>.

También se han utilizado las issues de GitHub para definir las tareas de cada parte del sistema, creando dos tableros, uno para el front-end y otro para el back-end. Se ha utilizado como una vista global del sistema.

14. CONCLUSIONES

El desarrollo de este proyecto ha sido interesante a la hora de conocer una serie de nuevas tecnologías que se utilizan mucho actualmente. Node y Express permiten crear un servidor que soporte peticiones REST en muy poco tiempo y de forma muy rápida, es uno de los aspectos más positivos que tienen, la rapidez y sencillez que presentan a la hora del desarrollo. Integrar una base de datos como Mongo con Node y Express resulta sencillo con Mongoose, aunque Mongo presenta algunos inconvenientes como actualizar los documentos, resulta más sencillo borrarlo y volver a guardarlo que aplicar una actualización en sí sobre el mismo. El uso de angular en el cliente de la aplicación ha sido sin duda una experiencia agri dulce, debido a que como se ha podido apreciar es un framework para aplicaciones web con una gran cantidad de posibilidades, potente y que permite realizar trabajos muy interesantes. Pero por otro lado, como se estudió en la asignatura, tiene una curva de dificultad muy pronunciada, además de que puede resultar algo complejo. De los dos miembros del equipo que desarrollaron la aplicación, uno había trabajado con angular antes, lo que facilitó la toma de contacto, mientras que el otro miembro tuvo que estar estudiando el funcionamiento del entorno durante un tiempo al comienzo del desarrollo. El despliegue a través de Heroku no ha requerido demasiado tiempo, la configuración de las máquinas resulta sencilla y la integración con GitHub lo facilita mucho.

Por otra parte, respecto a la gestión del proyecto debido a lo acontecido los meses de marzo y abril, nos ha resultado complicado y en este último tramo final del curso ha habido mucho trabajo acumulado de todas las asignaturas, lo cual nos ha complicado el desarrollo del trabajo y poder profundizar más en algunos aspectos.

15. VALORACIÓN PERSONAL DE CADA MIEMBRO

Gonzalo Berné:

La carga de trabajo que hemos llevado todos estas últimas semanas y la dificultad de trabajar en la situación que nos encontramos han sido puntos en contra en la realización de este trabajo y ha hecho que no se haya sobrellevado como mejor hubiéramos querido. No se ha conseguido un sistema final como nos habría gustado y como se planteó en un principio pero considero que a raíz del trabajo realizado hemos aprendido bastantes cosas sobre tecnologías que no habíamos utilizado mucho antes.

Eduardo Gimeno:

Aunque el proyecto esté lejos de ser lo que esperábamos cuando lo planteamos, el hecho de aprender estas nuevas tecnologías, me ha servido para aprender a desarrollar en la parte del back-end de forma rápida y diferente a lo que había hecho hasta ahora. Me ha sorprendido la rapidez de Mongo a la hora de trabajar con documentos, aunque resulte complicado en otros aspectos, como la actualización de los mismos.

Jorge Fernández:

Si bien el proyecto no ha concluido como planteamos, la valoración personal sobre el proyecto y en concreto sobre la parte de back-end que me toca es muy positiva, dado que me ha permitido interactuar con distintos módulos de muy variadas características, lo cual se ha trasladado en una importante búsqueda de información y solución de problemas, tanto con Nodejs como con MongoDB y todo ello me ha permitido desarrollar unos conocimientos que resultan muy útiles para la vida profesional.

Joaquín Puyuelo:

Considero que el trabajo desarrollado en este proyecto me ha ayudado a aprender en gran medida los conceptos del desarrollo en Front-End. A pesar de que al comienzo del trabajo se hizo algo duro el aprender a utilizar Angular, valoro muy positivamente los conocimientos adquiridos, ya que estoy seguro de que me serán de gran utilidad en el desarrollo de aplicaciones en un entorno de la vida real.

16. ANEXO

Email creado para utilizar en las cuentas de administrador:

- descubrearagonstw@gmail.com
- STW-1920

Email utilizado para las entradas, para comprobar el funcionamiento del chat:

- entradaexample@gmail.com
- STW-STACK-MEAN

Administrador:

- descubrearagonstw@gmail.com
- STW-1920