

E.T.S. de Ingeniería Industrial, Informática
y de Telecomunicación

Desarrollo de APP para el proyecto DEMOLA



Grado en Ingeniería
en Tecnologías de Telecomunicación

Nahia Rey Izco

Carlos del Río Bocio

Pamplona, 27-10-2020

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Resumen

Trabajo realizado en colaboración con la Asociación Española de Ayuda al Refugiado (CEAR) englobado en el proyecto DEMOLA. Dicho proyecto está enfocado a la colaboración entre diferentes empresas y la universidad, centrándose en la resolución de problemas planteados por estas.

El problema planteado por CEAR ha sido estudiado previamente en el trabajo “Nuevas soluciones de vivienda para migrantes y otros colectivos vulnerables” en el que se llega a la conclusión de que una de las soluciones para la búsqueda de vivienda asequible es el traslado a poblaciones pequeñas, planteando que es necesario solucionar la falta de transporte colectivo en estas.

En nuestro caso nos centramos en aplicar una de las soluciones a esto. Hemos creado una aplicación para facilitar a los habitantes de estas pequeñas poblaciones a compartir vehículos particulares, haciendo más sencillo a aquellas personas sin recursos a acceder a un medio de transporte.

Índice

| | |
|-------------------------------------------------------|----|
| 1. Introducción | 5 |
| 1.1 Situación..... | 5 |
| 1.2 Estudio de la solución | 6 |
| 1.3 Propuesta de la aplicación | 7 |
| 2. Herramientas utilizadas | 8 |
| 2.1 Criterios de selección | 8 |
| 2.2 Backend..... | 9 |
| 2.3 Base de datos | 10 |
| 2.4 Servidores | 10 |
| 2.5 Servicios externos. | 10 |
| 3. Estructura de código | 11 |
| 3.1 Frontend..... | 11 |
| 3.1.1 Carpeta SRC | 12 |
| 3.1.2 Código desarrollado (Carpeta src/app) | 12 |
| 3.2 Backend..... | 14 |
| 3.2.1 Modelos..... | 14 |
| 3.2.2 Rutas | 16 |
| 3.2.3 Clases | 18 |
| 3.2.4 Middleware..... | 19 |
| 4. Producto final | 20 |
| 4.1 Dónde se encuentra disponible la aplicación. | 20 |
| 4.2 Uso de la aplicación | 20 |
| 4.2.1 Inscripción: | 20 |
| 4.2.2 Edición de perfil..... | 21 |
| 4.2.3 Inicio de sesión. | 22 |
| 4.2.4 Página principal. | 23 |
| 4.2.5 Vista de viaje..... | 24 |
| 4.2.6 Chat | 25 |
| 4.2.7 Crear un viaje..... | 26 |

| | | |
|-------|----------------------------------|----|
| 4.2.8 | Notificaciones | 28 |
| 5. | Visión a futuro | 29 |
| 5.1 | Utilización..... | 29 |
| 5.2 | Marketing de la aplicación | 29 |
| 5.3 | Soporte y desarrollo..... | 30 |
| 6. | Bibliografía | 32 |

1. Introducción

1.1 Situación

Mediante el proyecto DÉMOLA se ha realizado una colaboración entre la asociación CEAR (Comisión Española de Ayuda al Refugiado) y las diferentes universidades que comprenden el Campus Iberus, entre las cuales se encuentra la UPNA.

Gracias a esta colaboración entre ambas entidades se detectó un problema creciente a la hora de realizar la tarea de acogimiento de Refugiados por parte de la asociación. Debido a la subida progresiva del precio de alquiler las ciudades grandes (20% en los últimos 5 años en Pamplona)[1] en cada vez es más complicado la obtención de vivienda para los usuarios de la asociación.

Por otra parte, se podría solucionar el problema de la subida de alquiler en grandes ciudades trasladando la búsqueda de vivienda a poblaciones más pequeñas, ya que los precios bajan en lugares con menos habitantes.

Con esta solución nos encontramos con otro problema por parte de los usuarios de la asociación CEAR, el transporte en poblaciones pequeñas está muy limitado y es necesario el disponer de un vehículo propio para poder trasladarse.

Por una parte, es necesario ya que CEAR, aparte de ofrecer servicios de búsqueda de vivienda también ofrece otros servicios como pueden ser la ayuda en la búsqueda de empleo, diferentes cursos.... Debido a esto es necesario que aquellas personas atendidas por la asociación tengan una forma sencilla de llegar a los centros de CEAR, los cuales se encuentran en grandes ciudades.

Por otra parte, la oferta de empleo en poblaciones pequeñas suele ser escasa y difícil de encontrar, mientras que en poblaciones más grandes se concentran la mayoría de estos.

Por lo tanto, si conseguimos mejorar la accesibilidad en poblaciones pequeñas a un transporte frecuente, permitiríamos que la búsqueda de vivienda en poblaciones pequeñas por parte de los refugiados sea una alternativa válida.

1.2 Estudio de la solución

Este problema ya se encuentra en estudio por diferentes instituciones de las poblaciones navarras, estos municipios cada vez reducen más su población y están intentando revertir esta situación.

Servicio de Taxi

Una de las soluciones ofertadas actualmente se trata de la implementación de un servicio de taxi a demanda subvencionado tanto por la población como por el gobierno de Navarra [2].

Este servicio supone un gran avance, pero los horarios siguen siendo muy limitados y no supone una solución demasiado óptima. Además, solo está implementado en ciertas poblaciones.

Colaboración entre vecinos

Otra de las soluciones que actualmente se utilizan es la de compartir viajes entre los vecinos. Muchos de los habitantes de pequeñas poblaciones que ya viven ahí se transportan periódicamente con vehículo propio a grandes ciudades. Estos vehículos se encuentran completamente desaprovechados ya que existe unas plazas no ocupadas que podrían ser utilizadas por otros habitantes del pueblo.

Esta solución también se ha empezado a implementar de forma independiente mediante la colaboración de los vecinos creando grupos con la plataforma de mensajería “WhatsApp”.

La principal limitación que encontramos actualmente con esta mecánica es la necesidad de conocer a los demás vecinos para poder introducirse en la mecánica de compartir coche. Aquellos habitantes que se trasladen nuevos al pueblo tendrían que esperar a trazar una red de contactos antes de poder utilizar esta solución.

Debido a que existe esta disposición de forma propia por parte de los habitantes de diferentes pueblos, queremos proporcionar una aplicación que les permita realizar la actividad de una forma mucho óptima y organizada. Que por otra parte con simplemente inscribirte en la aplicación te permita pedir el servicio sin necesidad de tener previamente ningún vínculo.

Esto también puede permitir que vecinos de poblaciones cercanas colaboren sin conocerse.

1.3 Propuesta de la aplicación

Esta propuesta ha sido desarrollada con anterioridad mediante un estudio[3] realizado dentro del proyecto DEMOLA con el nombre: “Nuevas soluciones de vivienda para migrantes y otros colectivos vulnerables”.

Este estudio llega a la conclusión de que la solución óptima para poder ayudar a estos colectivos es la colaboración con los vecinos. El tener una herramienta que ayude a los nuevos habitantes a obtener esa ayuda que sin conocer a nadie es complicado solicitar.

Extrapolando estas conclusiones del estudio nos hemos dedicado a hacerlo tangible en una aplicación.

Esta aplicación se ha desarrollado con el nombre “UNE-APP” como ha sido propuesto en el estudio realizado[3]. Este nombre viene referenciando la unión entre los vecinos para realizar esta colaboración.

En la aplicación se ofrece la posibilidad a todos los usuarios de ofertar servicios de transporte. Principalmente está pensado para que incluyan en la aplicación aquellos viajes que realizan con cierta frecuencia y que indiquen cuantas plazas les quedan libres para que las puedan compartir con aquellos que las necesitan.

Por otro lado, aquellos usuarios que se conecten a la aplicación con la intencionalidad de solicitar un vehículo se encontrarán con una interfaz en la que se le ofertará los diferentes trayectos disponibles.

Una vez que el usuario solicita un viaje se abre un chat en común entre todos los usuarios que se encuentran suscritos al viaje, pudiendo así comunicarse y concretar la forma en la que se realizará el desplazamiento.

2. Herramientas utilizadas

2.1 Criterios de selección

Uno de los criterios más importantes que utilizamos para seleccionar las herramientas a utilizar fue la capacidad de desarrollo multiplataforma. Esto nos permite una mayor flexibilidad de uso y reducir las limitaciones de entrada de usuarios al uso de esta.

Dentro del mundo de las aplicaciones multiplataforma existen varios criterios que consideramos prioritarios:

1. Comunidad grande de desarrolladores: Esto nos facilita por una parte encontrar información del uso del framework como la seguridad de que el framework se actualizará con rapidez en el caso de contener fallos.
2. Utilización de plugins en varias plataformas.
3. Utilización de lenguaje conocido: Esto nos permitiría dedicar más tiempo al desarrollo y no tanto tiempo a la curva de aprendizaje del lenguaje.
4. Visualización de sus componentes sencilla y estética: Esto supondrá un uso agradable por parte de los usuarios.

Al realizar la búsqueda encontramos que existían 4 tecnologías que se mencionaban en la mayoría de los sitios como frameworks óptimos para su utilización:

PhoneGap:

Se trata de un framework multiplataforma creado en el año 2011. Su desarrollo se realiza mediante lenguajes de programación Web como pueden ser: JS, HTML y CSS. Esta desarrollada por Adobe. Comprende una gran cantidad de plataformas en el desarrollo, pero esto puede dar algunos problemas de rendimiento.

IONIC

Framework de código libre que te permite en el mismo desarrollo crear múltiples aplicaciones de diferentes plataformas. Estaba sado en el desarrollo mediante Angular. Permite realizar la simulación mediante web ya que está basado en desarrollo web.

Xamarin

Framework de código libre que te permite el desarrollo mediante C# de aplicaciones multiplataforma. Está desarrollado por Microsoft. Esta también fue lanzada en 2011

ReactNative

Este es creado por Facebook y lanzado a mediados de 2015 lo que le convierte en uno de los frameworks más novedosos de los analizados. Te permite la programación mediante código web.

Finalmente nos decidimos por la utilización de **IONIC** ya que se basaba en un lenguaje que ya manejábamos (JavaScript) con una gran comunidad que podría solventar los diferentes problemas que fueran surgiendo. Así como la implementación de una gran cantidad de plugins que nos harían más sencillo la implementación de todas las funcionalidades necesarias.

Una de las mayores ventajas que nos proporciona IONIC es esa facilidad de programar mediante lenguaje web, ejecutando la mayoría de las funcionalidades mientras se va desarrollando la aplicación en nuestro propio navegador. El no tener que ejecutar continuamente la aplicación en un dispositivo físico o simulado simplifica ampliamente su desarrollo.

Por otra parte, nos proporciona gran cantidad de componentes prediseñados que se adaptan a la apariencia de cada Sistema Operativo. Esto hace que la visualización de la aplicación y su usabilidad resulten muy naturales y profesionales.

Por último y no por ello menos importante, con un solo desarrollo de software nos permite desarrollar una aplicación multiplataforma adaptable a diferentes Sistemas Operativos sin la necesidad de crear dos códigos independientes.

2.2 Backend

A la hora de realizar le backend también existen diferentes lenguajes de programación que podemos seleccionar para ello.

Al estar desarrollando simultáneamente tanto frontend como backend creímos conveniente el utilizar el mismo lenguaje de programación, por lo tanto, se ha utilizado también **JavaScript**.

Dentro de haber utilizado este lenguaje introducimos el framework NODE.JS, el framework más utilizado mundialmente para el desarrollo backend con JavaScript.

2.3 Base de datos

Para realizar la base de datos decidimos utilizar una no relacional frente a las bases de datos relacionales. Esto nos permitía tener una mayor flexibilidad en su manejo, así como una mayor eficiencia a la hora de la consulta de datos.

Finalmente se optó por la utilización del gestor de bases de datos “MongoDB”.

2.4 Servidores

Otro de los aspectos a tener en cuenta en el desarrollo *backend* es la selección de servidores que nos permitan acceder a los datos en cualquier dispositivo, en cualquier sitio.

Se han seleccionado el servicio proporcionado por “DigitalOcean” ya que nos ofrecen un servidor con SO Linux el cual nos proporciona una gran flexibilidad a la hora de instalar todos los programas necesarios para que nuestro código funcione correctamente.

También nos permite configurar el Cortafuegos de forma que la información se encuentre en un lugar seguro.

2.5 Servicios externos.

Para el correcto desarrollo de las notificaciones ha sido necesario la utilización de servicios externos.

FireBase: Plataforma desarrollada por Google enfocada al desarrollo de aplicaciones web. De todos los servicios que nos puede ofrecer esta tecnología, nosotros utilizamos la “mensajería en la nube”.

Habiendo creado un proyecto asociado a nuestra aplicación nos ofrece dos claves, la clave del servidor y el ID de remitente.

OneSignal: Es un servicio que se encarga de enviar las notificaciones Push entre dispositivos. Por sí sola no podría trabajar, necesita esas claves que hemos obtenido anteriormente en FireBase.

Una vez introducidas las claves y creado el proyecto ya se encuentra disponible para ser utilizado.

3. Estructura de código

3.1 Frontend

En el caso de Frontend, como hemos explicado anteriormente, utilizamos un framework llamado IONIC. Este Framework compone diferentes directorios:

- **Node_modules:** en dicha carpeta se instalan todos los módulos necesarios para que la aplicación pueda ejecutarse como una aplicación móvil. Estos archivos se crean de forma automática mediante el framework
- **Platforms:** Al simular la aplicación en un dispositivo físico o simulado se crea esta carpeta en la que contiene los diferentes archivos específicos de cada sistema operativo para que funcione en este.
- **Plugins:** Conforme queremos utilizar recursos ofrecidos por el dispositivo físico se necesita instalar ciertos plugins al proyecto. Estos se almacenan en dicha carpeta.
- **Resources:** Esta carpeta también se crea automáticamente. Contiene elementos multimedia necesarios para cada plataforma en la que se ha desarrollado.
- **Src:** Esta es la carpeta en la que más vamos a ahondar ya que contiene el código que nosotros hemos creado mediante los lenguajes de TypeScript, HTML y CSS.
- WWW.

Dentro del proyecto de IONIC también existen algunos archivos que nos interesan como pueden ser:

- **Config.xml:** Nos permite configurar ciertos comportamientos en la aplicación final, como pueden ser los permisos de acceso a internet, permisos de plugins...
- **Package.json:** Proporciona la información general de la aplicación como puede ser: Nombre de la aplicación, autores, página web...
- **Otros archivos:** Los demás archivos se crean de forma automática y manejan principalmente características de TypeScript.

3.1.1 Carpeta SRC

Como hemos dicho anteriormente esta carpeta es la que contienen los archivos enfocados al desarrollo de la aplicación:

- **App:** Contiene todo el código que desarrollamos. Nos centraremos en ella más adelante.
- **Assets:** Contiene contenido multimedia predefinido dentro de la aplicación.
- **Environments:** Esta carpeta contiene dos archivos, uno enfocado a variables que se utilizarán mientras se produce el desarrollo de la aplicación, mientras que el otro archivo contiene variables que se utilizarán en el momento de lanzar la aplicación a producción:
 - **Desarrollo:** environments/environment.ts
 - **Producción:** environments/environment.prod.ts
- **Theme:** Carpeta en la que se encuentran las variables globales de SCSS. Principalmente contiene los colores utilizados en el global de la aplicación.
- **Index.html:** Archivo en el cual comienza la aplicación a funcionar, como si está fuese una página web. En dicho archivo se define el nombre de la aplicación, el icono de esta...
- **Main.ts:** Se ejecutan las funciones para que la aplicación comience.

3.1.2 Código desarrollado (Carpeta src/app)

Como hemos explicado con anterioridad esta carpeta incluye la mayoría del código frontend desarrollado para realizar la aplicación.

El código está también diferenciado por diferentes bloques de archivos: Los principales son:

- Páginas:

Son bloques de código directamente relacionados con la apariencia física de la aplicación. Por cada página que puedes visualizar en la aplicación existe un bloque de código dentro de la carpeta src/app/paginas.

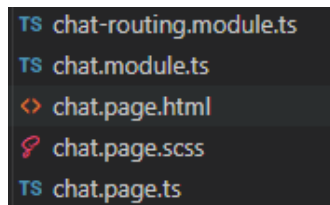
Cada página se encuentra en una carpeta que contiene los documentos:

- **Módulo de rutas de la página (TS):** En la mayoría de las páginas no tiene introducido más que su propia definición, a excepción de la página TAB. Esto se debe a que la página de TABS incluye otras tres, como son: "inicio", " viajesPropios" y recomendación.
- **Módulo de la página (TS):** En este elemento se importan los módulos externos que se van a utilizar dentro de la página.

- **Código HTML:** El código HTML se encarga de dar estructura a la página.
- **Código SCSS:** El código SCSS permite dar una visualización más personalizada a la página.
- **Lógica de la página (TS):** En este archivo se definen todas las funciones que se van a utilizar para que se pueda visualizar toda la información en la página. Es el punto en el que se piden los datos necesarios y si es preciso se transforman para su disposición en pantalla.

También incluye las funciones que dan respuesta a la interacción del usuario con el terminal.

Por ejemplo, la página que compone el chat tiene el siguiente esquema:



```
TS chat-routing.module.ts
TS chat.module.ts
<> chat.page.html
🔗 chat.page.scss
TS chat.page.ts
```

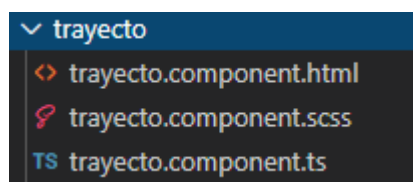
- Componentes:

Hay elementos en la aplicación que son necesarios que aparezcan en más de una página a la vez, al ser el mismo código nos gustaría tener que reutilizarlo sin tener duplicidades.

Este problema lo solucionan los componentes. Estos elementos de código se pueden insertar dentro de las páginas permitiendo que el mismo código se encuentre en dos páginas a la vez sin escribirlo dos veces.

Su esquema de archivos es bastante parecido al de una página:

- Código HTML
- Código SCSS
- Lógica del componente



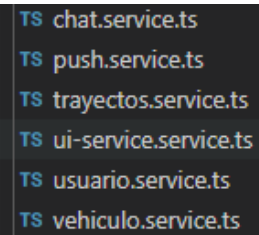
```
▼ trayecto
  <> trayecto.component.html
  🔗 trayecto.component.scss
  TS trayecto.component.ts
```

También existe un archivo *“module.ts”* que es compartido por todos los componentes.

- Servicios:

Estos archivos se encuentran en la carpeta *“src/app/services”*. Es la lógica de la aplicación que se encarga de la comunicación con el servidor.

Estos archivos responden más a la estructura que se ha creado de datos en el servidor que a la estructura de visualización de la aplicación. Desde cualquier página se llama a las funciones que nos ofrecen los archivos de servicio tanto para consultar información como para actualizarla o incluso eliminarla.



```
TS chat.service.ts
TS push.service.ts
TS trayectos.service.ts
TS ui-service.service.ts
TS usuario.service.ts
TS vehiculo.service.ts
```

3.2 Backend

3.2.1 Modelos

Uno de los elementos más básicos del servidor son los modelos. Están directamente relacionados con cómo se guarda la información en la base de datos.

Estos están diferenciados en cuatro grupos:

Chat:

- chat.model.ts
- mensaje.model.ts

Fechas:

- frecuencia.model.ts

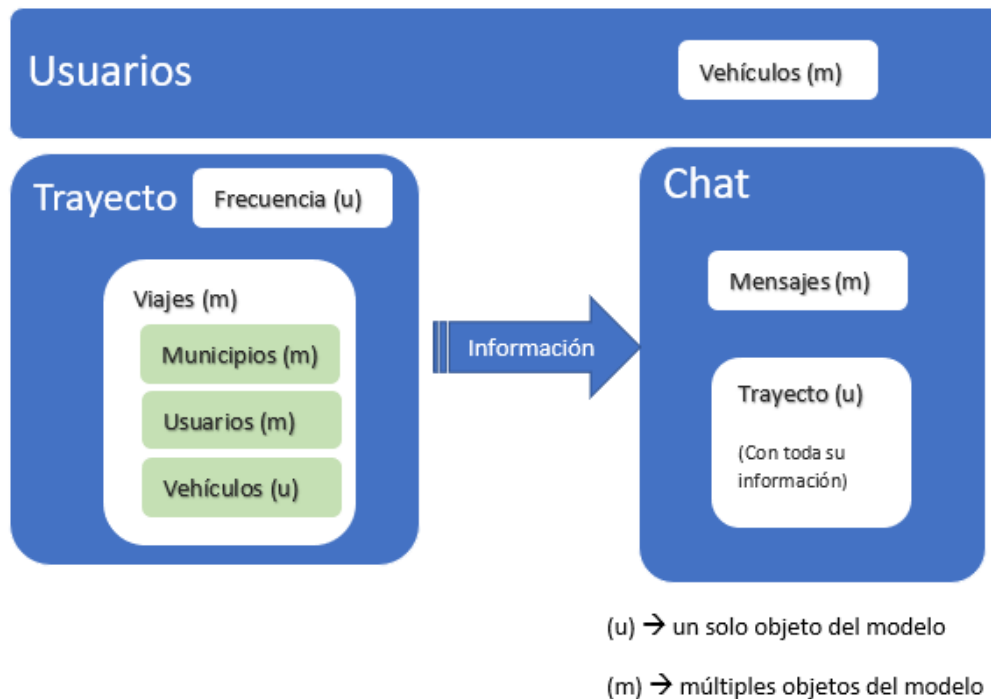
Usuarios:

- usuario.model.ts

Viajes:

- trayecto.model.ts
- viaje.model.ts
- vehiculo.model.ts
- municipio.model.ts

Relaciones entre ellos:



Usuarios ↔ Vehículos:

Cada usuario puede contener un número múltiple de vehículos. Estos se registran como propios del usuario para que posteriormente los pueda introducir en un trayecto.

Trayecto ↔ Frecuencia

El trayecto tiene una única frecuencia.

Trayecto ↔ Viajes

El trayecto puede contener uno o dos viajes. Esto dependiendo de si se trata de un viaje de ida o un viaje de ida y vuelta.

Esto nos permite que en el funcionamiento de la aplicación te puedas apuntar solo al viaje de ida y no ocupes la plaza del viaje de vuelta, permitiendo así que la plaza que quede libre pueda ser ocupada por otro usuario.

En un futuro esto permitirá un trayecto con múltiples paradas.

Viajes ↔ Municipios

El viaje tiene dos campos relacionados con municipios. Origen y destino, ambos campos son de valor único.

Viajes ↔ Usuarios

En el objeto Viajes existen dos atributos relacionados con el objeto "Usuario".

- Conductor: Campo único.
- Pasajeros: Campo múltiple.

Chat ↔ Mensajes.

El chat contiene un array de mensajes. Por lo tanto, contiene múltiples mensajes.

Chat ↔ Trayecto

El chat tiene un campo único de trayecto. Este indica a que trayecto corresponde.

3.2.2 Rutas

En esta se definen las diferentes peticiones que se pueden hacer al servidor de forma externa.

En la definición se incluye toda la lógica que es necesaria realizar para esto. (Peticiones de información a la base de datos, manejo de información...).

Como URL para comunicarse con el servidor es *178.62.33.148:3000*.

La ruta está formada por: `ruta_servidor + /{{nombre_archivo}}+/{{elemento}}`

Los archivos de rutas son:

- Users:
 - **Get /:** Obtener lista de todos los usuarios.
 - **Get /yo:** Información de usuario.
 - **Get /:id:** Obtener información de usuario mediante id.
 - **Post /login** Iniciar sesión mediante correo y contraseña
 - **Post /créate:** Crear un usuario.
 - **Post /update:** Actualizar información de un usuario
 - **Post /checkPassword:** Comprobar si la contraseña es correcta
 - **Post /fotoPerfil:** Actualizar foto de perfil en el usuario

- Viaje:

/* Obtener información del viaje*/

- **Get /listado:** Obtener viajes paginados
- **Get /conductor:** Obtener los viajes en los que eres conductor
- **Get /pasajero:** Obtener los viajes en los que eres pasajero
- **Get /:viajeld:** Obtener el viaje por id

/* Editar información del viaje*/

- **Post /:** Crear un nuevo viaje
- **Post /editar:** Editar un viaje.
- **Post /eliminar:** Eliminando un viaje.
- **Post /nuevoPasajero:** Introducimos un nuevo pasajero en el viaje.

/* Sección de municipios*/

- **Get /municipios:** Obtenemos todos los municipios de España guardados en la base de datos.

- Trayecto

- **GET /** Listado de trayecto
- **GET /pasajero** Obtener los trayectos en los que eres pasajero
- **GET /conductor** Obtener los trayectos en los que eres conductor
- **GET /frecuencia** Obtener el listado de frecuencias
- **GET /frecuencia/acotado** Obtener el listado de trayectos entre dos fechas
- **GET /frecuencia/:id** Obtener una frecuencia en concreto con su id

- **POST /** Crear un nuevo trayecto
- **POST /elemento** Obtener la información de un trayecto por id en el body.
- **POST /editar** Editar un trayecto
- **POST /frecuencia** Creamos una nueva frecuencia
- **POST /eliminar** Eliminamos un trayecto

- Chat

- **GET /:** Listado de chats
- **GET /mensajes/:id** Listado de mensajes de un chat con id

- **GET /:id** Obtención de un chat por id
- **POST /mensajes** Crear un nuevo mensaje
- **POST /mensajes/eliminar** Eliminar un mensaje

- **POST /** Crear un nuevo chat
- **POST /nuevoUsuario** Añadir un usuario al chat
- **POST /eliminarUsuario** Eliminar un usuario del chat
- **POST /información** Añadir información al chat

- Vehículo
 - **GET /** Obtener el listado de todos los vehículos
 - **GET /misVehiculos:** Obtener los vehículos del usuario
 - **GET /:id** Obtener los datos de un vehículo en concreto

 - **POST /:** Crear un nuevo vehículo
 - **POST /update:** Actualizar datos de un vehículo
 - **POST /delete:** Eliminar un vehículo

- pushUpNotification
 - **POST /enviarNotificacionUsuarios** Enviar notificación a usuarios en concreto.
 - **POST /enviarNotificacionGeneral** Enviar notificación a todos los usuarios de la aplicación

3.2.3 Clases

Funciones específicas realizadas de forma generalizada en el servidor. En nuestro caso tenemos tres:

- File-system.ts: Se encarga de manejar los ficheros que se suben al servidor. En nuestro caso de las imágenes que se suben como fotos de perfil
- Server.ts: Funciones principales que ponen en marcha el servidor.

- Token.ts: Se encarga de manejar el token de usuario que verifica si los usuarios que contactan con el servidor son válidos.

3.2.4 Middleware

Función que se utiliza antes de realizar la lógica de la ruta especificada. En nuestro caso tenemos el middleware de autenticación. En la mayoría de las peticiones se pide que se envíe en la cabecera el token de autenticación para así, antes de contestar a cualquier notificación se compruebe que esta está hecha por algún usuario de la aplicación y que tiene autorización a acceder a dicha información.

4. Producto final

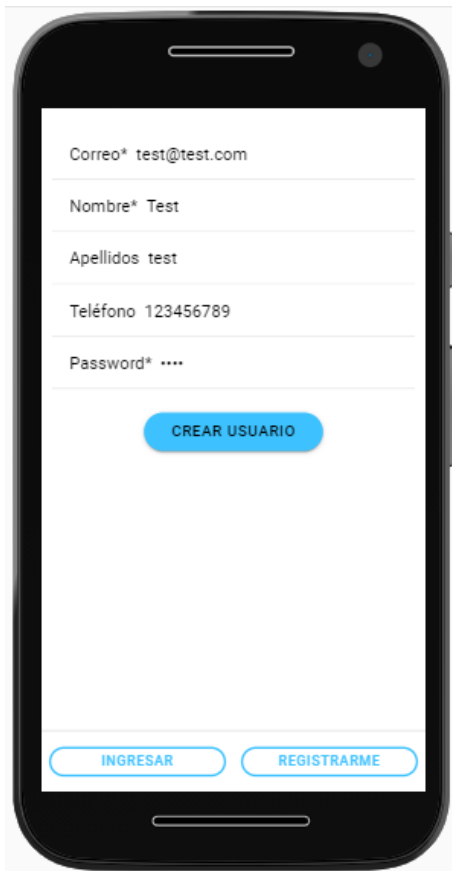
4.1 Dónde se encuentra disponible la aplicación.

Actualmente la aplicación se encuentra disponible en Android store .

4.2 Uso de la aplicación

El primer paso que es necesario realizar en la aplicación es Inscribirte en ella.

4.2.1 Inscripción:



Como podemos ver, la pantalla de inscripción es bastante sencilla.

Permite al usuario introducir elementos básicos como pueden ser:

Correo, Nombre, Apellidos, Teléfono y contraseña.

Aquellos campos que acaben en un * son los campos obligatorios para poder crear el usuario.

Una vez introducidos estos campos, pincharemos en “CREAR USUARIO”. Si todo ha salido bien la aplicación redireccionará al nuevo perfil creado.

4.2.2 Edición de perfil

En esta página existen varias funcionalidades. Por un lado, tenemos la sección en la que podemos visualizar como editar nuestros datos de perfil.

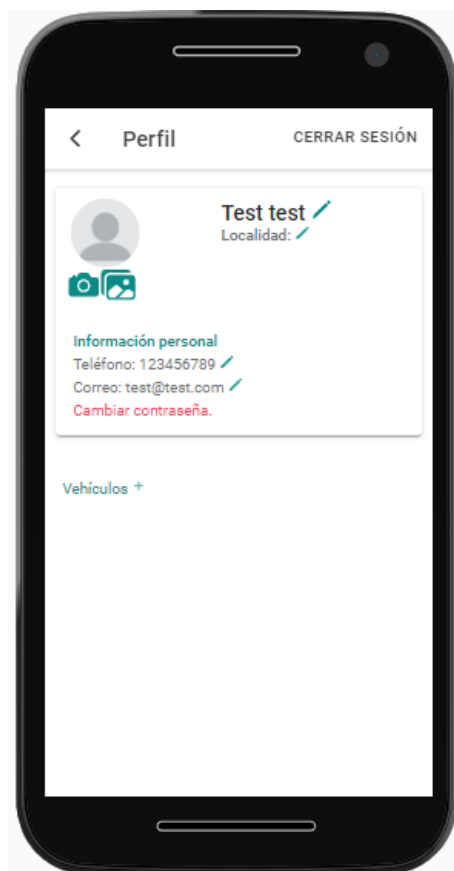
- Datos de perfil

Estos datos se pueden cambiar clicando en los lápices que aparecen al lateral de su valor.

Como se puede observar la **imagen de perfil** por defecto aparece vacía. Puede ser seleccionada mediante una foto actual mediante la cámara o seleccionar una foto que tengamos en nuestra librería.

Una vez se haya realizado algún cambio dentro de los **datos del perfil**, incluida la imagen, aparecerá un botón **GUARDAR** que nos permitirá guardar los datos de forma definitiva

El **cambio de contraseña** obliga a reintroducir la contraseña antigua antes de cambiarla por seguridad.



- Vehículos

Esta sección aparece completamente vacía la primera vez que creamos nuestro perfil. Esta sección está enfocada para aquellos usuarios que deseen compartir los viajes que realicen en su propio vehículo con los demás.

Permite guardar los datos de los vehículos que se utilicen sin necesidad de introducir los datos cada vez que se cree un viaje.

Citroen - Verde  

Plazas: 7

Descripción:

Coche familiar

Como podemos ver los datos que se piden son la marca, el color, el número de plazas que dispone y una descripción.

Esta información aparecerá en la descripción de los viajes para que sea más sencillo a los pasajeros identificarlos. En el caso de que no se desee dar mucha información con introducir el número de plazas es suficiente

4.2.3 Inicio de sesión.



En el caso de que se te haya cerrado la sesión o estés entrando a la aplicación por primera vez, esta será la pantalla que te aparecerá por defecto

Es una pantalla simple en la que te permite introducir tanto correo como contraseña.

En el caso de que se te haya olvidado la contraseña aparecerá una opción de recuperar contraseña con la cual se te enviará un correo para reestablecerla.

4.2.4 Página principal.

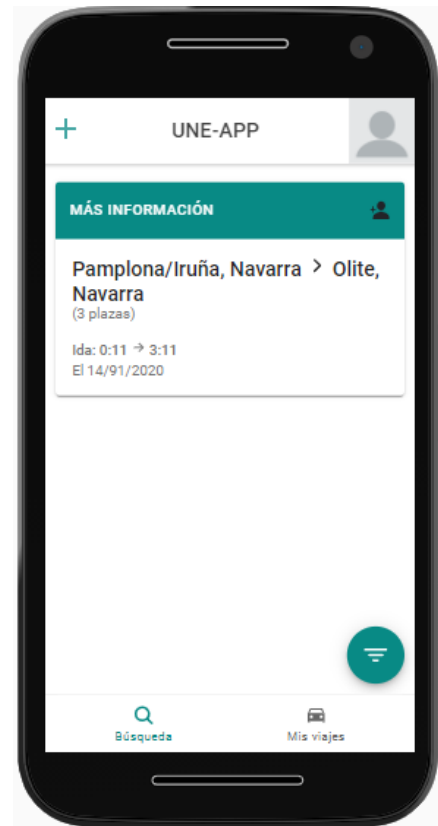
Una vez iniciada sesión, si volvemos a abrir la aplicación nos lleva directamente a la página principal.

Esta está compuesta por dos secciones principales:

- **Búsqueda:**

Elemento de la página en la que se publica los viajes introducidos por los demás usuarios.

Para buscar, se puede utilizar el botón colocado abajo a la derecha. Este botón desplegará la página de **filtros**, permitiendo filtrar los viajes interesantes.



- **Mis viajes:**

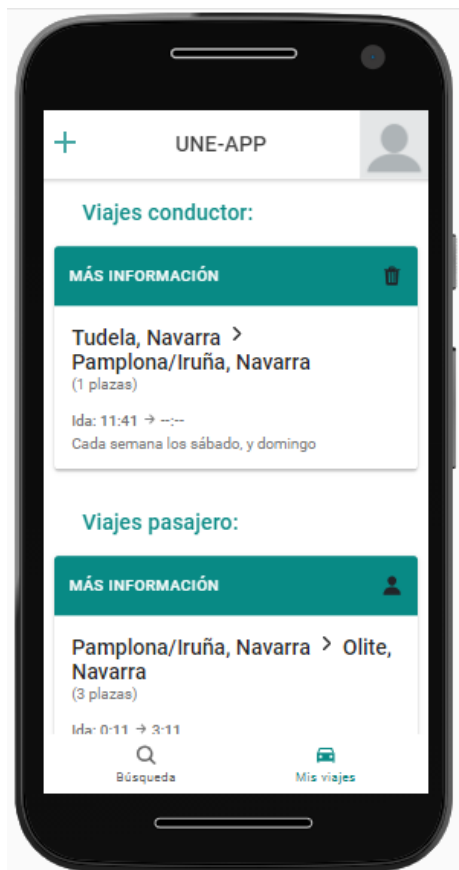
Si seleccionamos en las pestañas situadas abajo aquella que indica "Mis Viajes". Estos comprenden los viajes en los que participas, tanto como conductor como pasajero. Apareceremos en una pantalla en la que nos aparecen dos secciones:

- **Viajes Conductor:**

Listado de viajes creados por uno mismo.

- **Viajes Pasajero:**

Viajes en los que te has apuntado como pasajero.



4.2.5 Vista de viaje.




Clicando en uno de los viajes en “Más información” la aplicación te lleva directamente a esta vista.

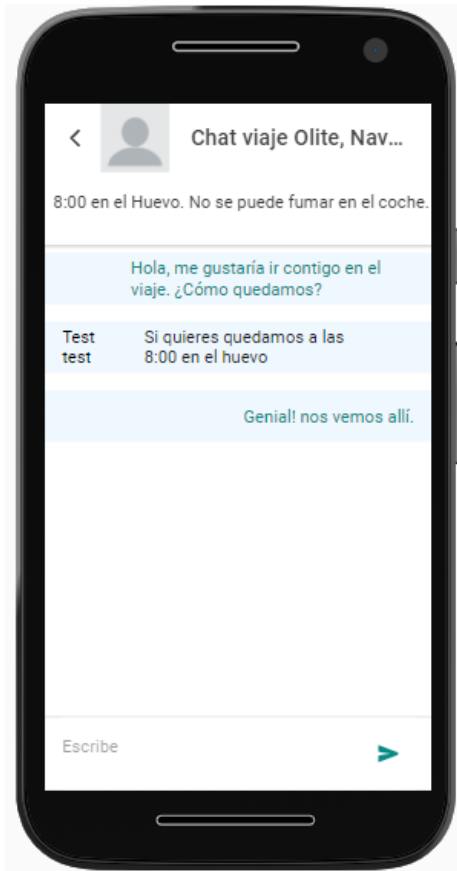
En ella puedes encontrar toda la información relacionada con el viaje como es:

- Conductor del trayecto.
- Vehículo.
- Detalle viajes.

Ya que el trayecto puede ser de ida o de vuelta el trayecto se desglosa en viajes. Uno si es únicamente de ida y dos si es de ida y vuelta. Cada viaje tiene sus propios pasajeros.

En el caso de que entres a los detalles de un viaje creado por ti aparecerá el botón de editar viaje abajo a la derecha. 

4.2.6 Chat



Una vez que existe un viaje se crea un chat asociado a él.

Dicho chat está diseñado para poder comunicarse entre todos los pasajeros y conductor del viaje.

Tiene tres partes diferentes:

Cabecera:

Indica el chat de qué viaje te encuentras

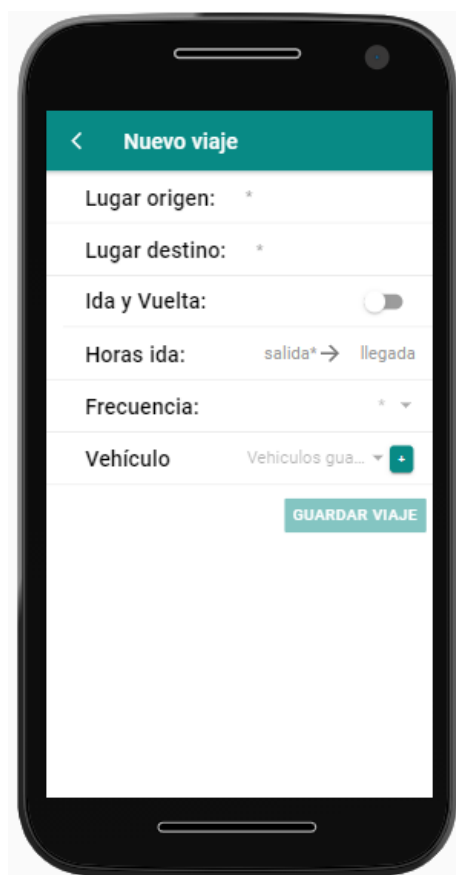
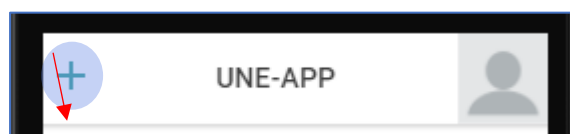
Información fijada:

Información escrita por el conductor. Está pensada para poner información importante que necesite quedarse a la vista para todo el mundo

Chat:

Intercambio de información entre los pasajeros y conductor del viaje. Permite discutir los diferentes detalles del viaje.

4.2.7 Crear un viaje



Al seleccionar el botón que aparece arriba a la izquierda en la cabecera de las páginas principales podremos crear un nuevo viaje.

Como podemos ver en la imagen de la derecha hay diferentes campos a rellenar:

Lugar Origen*: Selección entre todos los municipios de España.

Lugar destino*: Selección entre todos los municipios de España.

(Tanto el origen como el destino se seleccionan con una vista explicada más adelante)

Ida y vuelta: Si se activa el trayecto será de ida y vuelta.

Horas ida: Aparece siempre

- **Salida*:** Introducir la hora obligatoriamente
- **Llegada:** No es obligatorio introducirlo

Horas vuelta: Aparecerá siempre que se seleccione “ida y vuelta”

- **Salida:** Es obligatorio introducirlo si es el viaje de ida y vuelta
- **Llegada:** No es obligatorio introducirlo.

Frecuencia*: Existen diferentes opciones a la hora de seleccionar la frecuencia de un viaje.

Sin repetición: Te permite seleccionar un único día de viaje.

| | |
|--------------------|------------------|
| Frecuencia: | Sin repetición ▾ |
| Día: | 18/12/2020 |

Cada x días: Te permite repetir el viaje los días que indiques. Empezando desde el día dado

| | |
|----------------|---------------|
| Frecuencia: | Cada x días ▾ |
| Cada: | días. |
| Día de inicio: | 18/12/2020 |

Semanalmente: Repetir semanalmente los días de la semana a elegir.

| | |
|-------------|-------------------|
| Frecuencia: | Semanalmente ▾ |
| Días: | Lunes, Mierc... ▾ |

Mensualmente: Se repite mensualmente el día elegido.

| | |
|-------------|----------------|
| Frecuencia: | Mensualmente ▾ |
| Día: | 18/12/2020 |

Vehículo:

La selección de vehículo te permite seleccionar de entre los vehículos que ya tienes guardado en tu perfil. En el caso de que el vehículo que quieras introducir no lo tengas previamente guardado en el perfil puedes añadirlo mediante el botón (+) y se te seleccionará directamente.

Este vehículo quedará disponible para próximos viajes.

Horas ida: salida* -> llegada

Vehículo

Renault - Blanco 2020

CANCEL OK

GUARDAR VIAJE

Selección municipios:

Para seleccionar los municipios se utiliza una nueva lista en la que se puede seleccionar entre todos los municipios de España.



4.2.8 Notificaciones

Existen diferentes notificaciones en la aplicación para ayudar a los usuarios a enterarse de los cambios que aparecen en la aplicación:

Notificaciones en el chat:

Cuando alguien escribe en el chat, a todos los usuarios que pertenecen a dicho trayecto se les envía una notificación.

 MyApp • 2 min

UNE-APP: Nuevo mensaje en chat.
Genial! nos vemos allí.

Notificaciones de cambio de información:

En el caso de que uno de los viajes en los que te encuentres registrado sufra un cambio de información se te notificará.

 MyApp • ahora

UNE-APP: Cambio en su viaje
Se ha editado su viaje con trayecto de Estella/
Lizarra, Navarra a Lerín, Navarra

Eliminación de un viaje:

También se notificará en el caso de que se elimine un viaje en el que estés suscrito

 MyApp • ahora

UNE-APP: Cambio en su viaje
Se ha eliminado su viaje con trayecto de Estella/
Lizarra, Navarra a Lerín, Navarra

5. Visión a futuro

5.1 Utilización

Es una aplicación que para que tenga una utilidad es necesario que cierta cantidad de personas la utilicen.

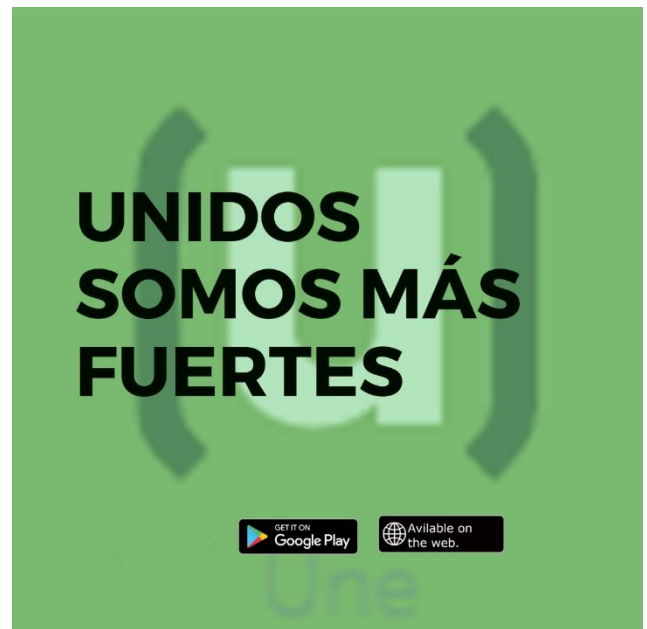
Esta aplicación está pensada para que en colaboración con CEAR y diferentes instituciones públicas se promueva su uso.

5.2 Marketing de la aplicación

Debido a la necesidad de promoción de la aplicación se han realizado dos propuestas de anuncios de marketing.

Unidos somos más fuertes:

Apelando a la unidad de los ciudadanos lanzamos este mensaje. Enfocado a causar una curiosidad al lector, haciendo que este busque información en internet



¿En qué podemos ayudar?

En este caso damos un poco más de información sobre la aplicación y su finalidad.

Enfocamos la información a el principal objetivo de la aplicación. Compartir coche de forma colaborativa



¿En qué te podemos ayudar?

Te conectamos con personas de poblaciones cercanas que hagan el mismo recorrido que tú.

Compartir coche, la nueva forma de movilidad.

Somos UNE APP ¡Búscanos!

5.3 Soporte y desarrollo

Para que la aplicación siga funcionando a un futuro lejano es necesario realizar un mantenimiento.

Mantenimiento económico

El servidor que se encuentra contratado actualmente en red supone un gasto mensual de 5€, pero en el caso de que el uso de la aplicación crezca con el tiempo este servicio no será suficiente por lo que se tendrá que revisar la capacidad de los servidores contratados.

Problemas con la aplicación

Al darse uso a la aplicación de una forma real es inevitable que aparezcan errores no previstos en esta. Esto requiere que haya alguien pendiente de los errores para poder solucionarlos cuanto antes.

Desarrollo de la aplicación

La aplicación actualmente está desarrollada con la capacidad de introducir muchas novedades en ella que podrían hacer la aplicación mucho más útil.

- Búsqueda automatizada de viajes: en el caso de que los usuarios no encuentren el viaje que buscan podrían guardar los filtros que les interesen y se les avisará de cuando encuentren un viaje disponible.

- Introducción de la aplicación en el sistema operativo IOS.

- Adaptar la aplicación a las peticiones de los usuarios.

6. Bibliografía

[1] J. A. Monreal, “La rentabilidad del alquiler de vivienda crece hasta el 5% en Pamplona y Comarca,” *Not. Navarra*, p. 1, 2020, [Online]. Available: <https://www.noticiasdenavarra.com/economia/2020/08/06/rentabilidad-alquiler-vivienda-crece-5/1068355.html>.

[2] Gobierno de Navarra, “prestacion del servicio de taxi a la demanda,” 2013. http://www.navarra.es/home_es/Actualidad/Sala+de+prensa/Noticias/2013/02/18/pr+estacion+del+servicio+de+taxi+a+la+demanda.htm.

[3] “Nuevas soluciones de vivienda para migrantes y otros colectivos vulnerables.”