Manual de uso y referencia
EcoCarga

Cliente : Ministerio de Energía

~~15~~22-05-2019

Mauricio Zúñiga G.
+56 9 94204180
mzuniga@gmail.com

Contenido

1. Introducción 4
2. Arquitectura: descripción general 5
3. Aplicación Móvil 6
   3.2 La aplicación 6
      3.2.1 Inicio de aplicación (Logo) 6
      3.2.2 Inicio aplicación (Splash Screen) 7
      3.2.3 Selección de modelo de auto 7
      3.2.4 Menú 8
      3.2.5 Mapa 9
      3.2.6 Electrolineras 11
   3.3 Android 15
      3.3.1 Lenguaje de programación 15
      3.3.2 Arquitectura general 15
      3.3.3 Librerías asociadas 17
      3.3.4 Datos de entrada 17
      3.3.5 Instalación y configuración en Play Store 17
   3.4 iOS 17
      3.4.1 Lenguaje de programación 17
      3.4.2 Arquitectura general 18
      3.4.3 Librerías asociadas 18
      3.4.4 Datos de entrada 19
      3.4.5 Instalación y configuración en App Store 19
4. Servidor 20
   4.2 Administrador 20
      4.2.1 La herramienta 20
      4.2.2 Configuración del administrador 29
      4.2.3 Usuarios 29
   4.3 Datos de entrada 29
   4.4 Modelo de datos 30
   4.5 Descripción del proyecto 31
   4.6 API REST para aplicación móvil 34
   4.7 Instalación 36
   4.8 Configuración actual del servidor 41
5. Lineamientos de diseño 43

5.2 Norma 43
1.2.1 Logo 43
1.2.2 Articulaciones 44
1.2.3 Variaciones de color 45
1.2.4 Proporciones y tamaños 47
1.2.5 Usos correctos e incorrectos 48
1.2.6 Formatos y aplicaciones 49

1. Introducción

Dentro del contexto de electro movilidad el Ministerio de Energía encargó al Instituto de Sistema Complejos de Ingeniería (ISCI), la labor del diseño y desarrollo de una aplicación móvil que lograra disponibilizar e informar a los usuarios de automóviles eléctricos información de las electrolineras dentro de Chile. El desarrollo consistió en la construcción de una aplicación móvil que muestra en un mapa la posición de cada electrolinera junto con variada información asociada, y un sistema de back-end que provee un administrador para que personal del ministerio pueda actualizar dicha información y esta sea transmitida a la aplicación en los teléfonos de los usuarios.

El periodo de desarrollo se llevó a cabo entre los meses de septiembre de 2018 a noviembre de 2018, donde adicionalmente se realizaron trabajos de mantención posterior a ese periodo durante los meses de noviembre y diciembre de 2018.

Este manual es un documento que describe en detalle la arquitectura de todo el sistema: aplicación móvil, sistema de back-end y lineamientos de diseño. El objetivo principal es capacitar al Ministerio de Energía para que sea capaz de dar soporte a la plataforma, mantenerla y mejorarla si lo estiman conveniente.

2. Arquitectura: descripción general

Para que este tipo de aplicación móvil pueda operar es necesario construir un sistema o plataforma compuesto por dos módulos, por una parte, la aplicación móvil a la cual accede directo el usuario final y por otra una plataforma web que permita administrar los contenidos entregados por la aplicación móvil.

La comunicación entre la plataforma web y la aplicación móvil se realiza cada vez que esta última es abierta, ante lo cual se comunica con la plataforma web para verificar si los datos cambiaron, si ocurrió una modificación los nuevos datos son descargados, estos corresponden a: electrolineras, modelos de autos y tipos de conectores. Si no se reporta modificación en los datos, la aplicación utiliza los almacenados localmente.

El sistema web contiene una base de datos con todos los registros requeridos por la aplicación móvil, estos pueden ser actualizados a través de una interfaz de administración, cualquier modificación hecha a través de esta interfaz es puesta a disposición de todos los usuarios de la aplicación móvil, es decir, la próxima vez que abran la aplicación y se realice la acción descrita en el párrafo anterior.

En la se puede observar un esquema global del sistema a la fecha, en el centro se observa el servidor web, que está instalado en los servicios AWS de Amazon, este servidor contiene una base de datos con la información del sistema y provee de una interfaz web (en la figura se llama “administrador web”) que permite modificar cualquiera de estos datos en forma manual. A la fecha esta información es entregada en forma escrita por la SEC (datos de electrolineras) y por el Ministerio de Energía (datos de modelos de autos y tipos de conectores). Por otro lado, se aprecian las aplicaciones móviles (iOS y Android), que se comunican con el servidor para poder actualizar la información. Es importante destacar que ambas aplicaciones móviles fueron desarrolladas de manera nativa por lo que fueron escritas en lenguajes de programación propios de cada plataforma.

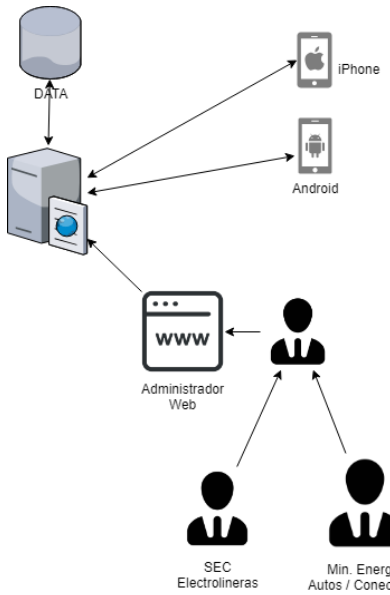


Ilustración 1: Diagrama de principales actores y módulos

3. Aplicación Móvil

La aplicación móvil fue diseñada para que los usuarios puedan informarse principalmente de la posición de las electrolineras, pero durante el desarrollo se detectó que para realizar esto con mayor eficacia era necesario tener la información del modelo de auto del usuario, de tal manera de especificar a través de la aplicación móvil las electrolineras que le son útiles, junto con esto también se integraron datos específicos de las electrolineras rescatados de sitios web asociados al tema.

Toda esta información finalmente es entregada a través de la aplicación móvil la cual fue desarrollada para dos sistemas operativos: Android y iOS, elegidos por concentrar gran parte del mercado de *smartphones*. La aplicación es idéntica en ambos sistemas operativos, pero por detrás sus desarrollos difieren en lenguaje y estructura, es por esto que cuando se presenta la interfaz de la aplicación no se hace distinción, pero si al describir el desarrollo del software.

3.2 La aplicación

En esta sección se realizará una descripción de la interfaz de la aplicación detallando cada funcionalidad para el usuario, para hacer presentaremos cada una de las vistas que contiene la aplicación usando el flujo ideal de uso.

3.2.1 Inicio de aplicación (Logo)

Toda aplicación utiliza un acceso representado por un logo, esta aplicación se presenta con un logo que hace referencia a una electrolinera de color verde, tal como muestra la Ilustración 2.

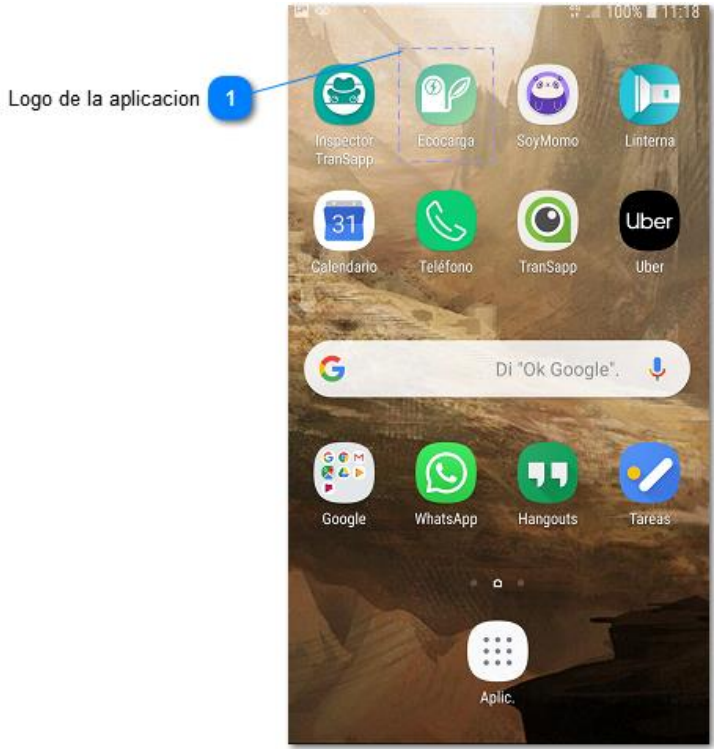


Ilustración 2: Logo y acceso a la aplicación.

3.2.2 Inicio aplicación (Splash Screen)

Al comenzar, la aplicación muestra una imagen mientras se cargan los datos, por una parte, es en esta parte del flujo donde se llevan a memoria los datos guardados en el teléfono y además es en esta etapa donde la aplicación se comunica con el servidor central para actualizar la última versión de los datos de electrolineras, automóviles y tipos de cargadores. La Ilustración 3 muestra como se ve esta vista en el teléfono.

Ilustración 3: Splash Screen, o vista de carga al iniciar la aplicación.

3.2.3 Selección de modelo de auto

Una vez cargada la información necesaria durante la *Splash Screen* y siendo la primera vez que el usuario abre la aplicación se dará paso a la vista de selección del modelo de automóvil del usuario, esto con el objetivo de poder identificar las electrolineras compatibles y también para estimar el tiempo de carga en cada una de ellas. En caso de no ser la primera vez que se abre la aplicación se salta esta sección, pero puede ser siempre modificada desde el menú de la aplicación, menú que se detallara más adelante.

Ilustración 4: Vista de selección de modelo de auto

3.2.4 Menú

La aplicación también provee un menú clásico, a través de un botón hamburguesa, en la Ilustración 5 se puede observar en la esquina superior izquierda (1), al presionarlo se despliega el menú de izquierda a derecha, en la Ilustración 5 la imagen de la derecha, la cual presenta a la fecha dos opciones, la primera corresponde a la vista para seleccionar el modelo de automóvil, esto siempre está disponible para los casos cuando el usuario cambie su vehículo o tenga más de uno, por otro lado, también en este menú se presenta un acceso a los Términos y Condiciones de la aplicación.

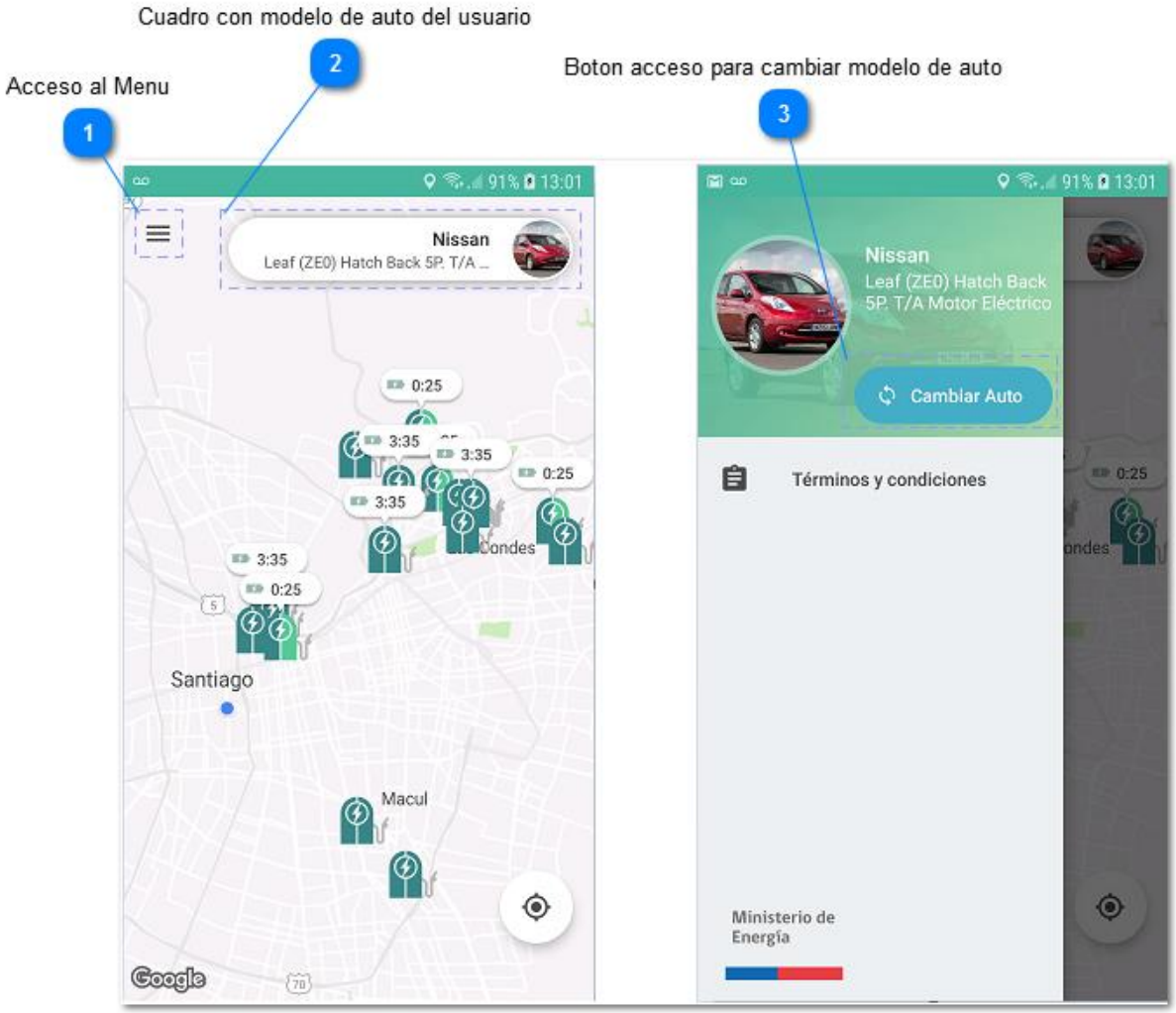


Ilustración 5: Vista inicial de la aplicación (Izquierda) se aprecia el acceso al menú, vista del menú (derecha) donde se aprecia el acceso para ir a la vista de cambio de modelo de auto.

3.2.5 Mapa

La vista principal de la aplicación corresponde al mapa (Ilustración 6) centrado en la ubicación del usuario y que contiene un icono en la posición de cada electrolinera (2), cada icono además muestra: el tiempo estimado de carga con el vehículo del usuario, en caso de no ser compatible no hay un tiempo asociado (4), el icono de divide en dos partes indicando con el color si es de tipo AC o DC, de esta manera cuando se ven dos tipos de colores verdes indica que tiene ambos tipos de corriente.

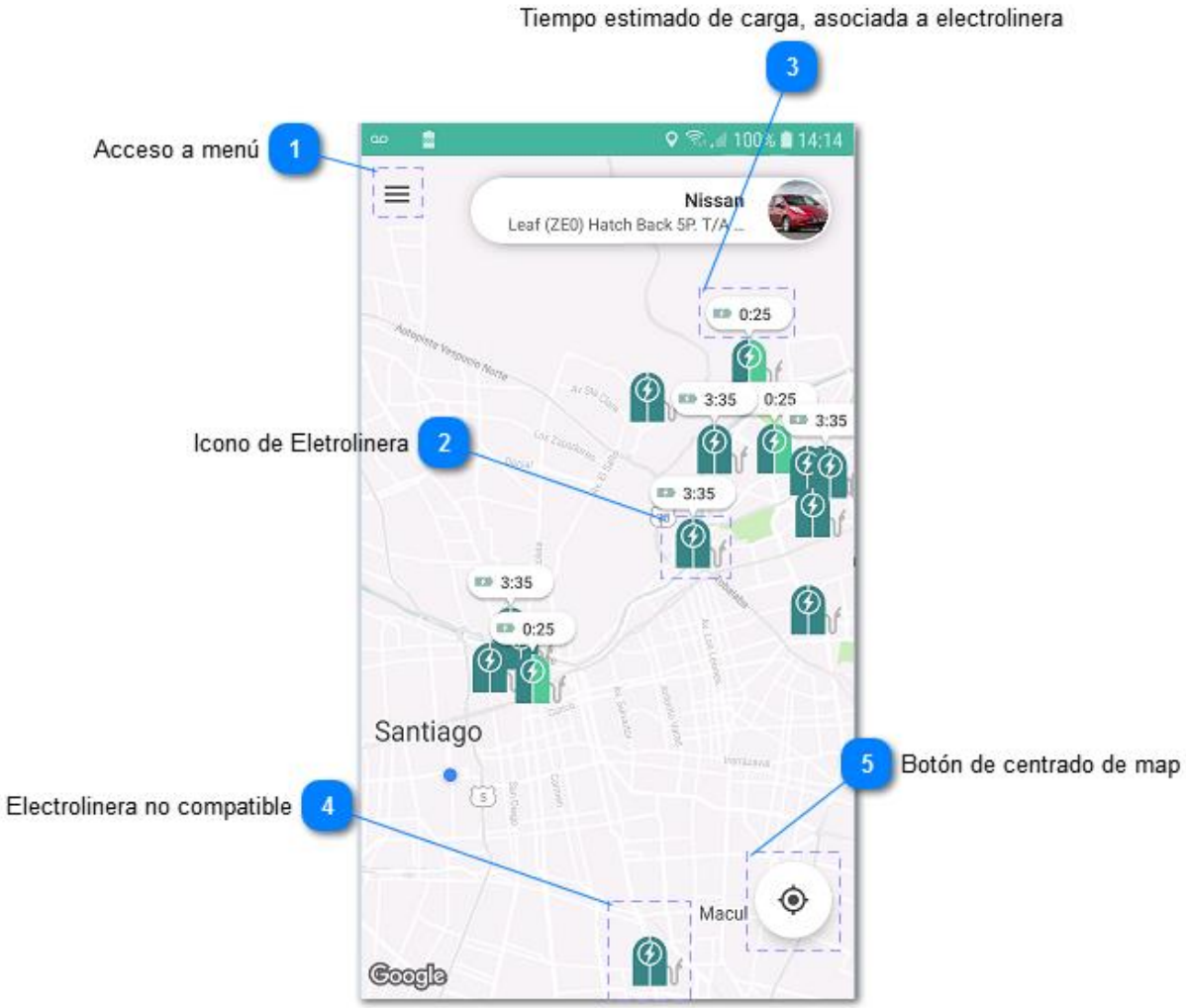


Ilustración 6: Vista de Mapa

La vista de mapa permite al usuario desplazarse arrastrando el dedo en la pantalla, también permite realizar acercamiento y alejamiento del mapa con el objetivo de poder buscar electrolineras muy lejanas a la posición actual del usuario. Como se ve en la Ilustración 6 se provee además de un botón de centrado (5) de tal forma de siempre poder volver a la posición actual del usuario.

Es importante mencionar la metodología del cálculo del tiempo de carga, para esto debemos realizar las siguientes definiciones:

Capacidad [kWh] de la batería: es el estanque del vehículo. Es decir, cuanta energía se puede almacenar.

Capacidad [kW] inversor interno AC: es la potencia máxima a la que puede ser cargada la batería.

Además, se realizan los siguientes supuestos:

1. Al momento de la carga siempre se llega con algo en la batería (nunca es cero) y se asume que es el 20 % porque los VE te pide recargar en ese %.

2. La T° en la batería permanece en el rango que permite la carga sin limitaciones, este supuesto tiene más validez en la carga en DC y con potencias mayores a 40 kW.
3. Para el ejemplo asumiremos la potencia del cargador AC igual a 11 kW.

La metodología se describe a continuación, utilizando un ejemplo:

Marca	Modelo	Capacidad kW inversor interno AC	Capacidad [kWh] de la batería
Renault	Fluence ZE Sedán 4P. T/A Motor Eléctrico	43	22
Hyundai	Ioniq AE Automóvil 4P. T/A Motor Eléctrico	6,6	28

Tabla 1: ejemplo para describir cálculo de tiempo de carga según modelo de auto

Renault

Energía necesaria: 22-22*0.2 = 17,6 kWh (supuesto carga al 20%)
Tiempo de Carga :: 17,6 / 11 ➔ 1 hora 41 minutos

Hyundai

Energía necesaria: 28-28*0.2 = 22,4 kWh (supuesto carga al 20%)
Tiempo de Carga :: 22,4 / 6,6 ➔ 3 hora 24 minutos

3.2.6 Electrolineras

Una vez que el usuario ve las electrolineras en el mapa, puede pedir más información de cada una de ellas, al tocarlas en la pantalla se desplegará una vista con información específica de la electrolinera seleccionada, en la Ilustración 7 se aprecia que al hacer clic sobre una electrolinera aparece un recuadro desde abajo (2) con la información del nombre de la electrolinera (5), dirección, si está en funcionamiento (4) y el tiempo estimado de carga con mayor tamaño, este tiempo corresponde al tiempo del conector más rápido (3). Además, en la parte inferior del recuadro aparecen tres ítems en la horizontal correspondientes a detalles en las categorías de información adicional, tipos de conectores e indicaciones (6, 7 y 8).

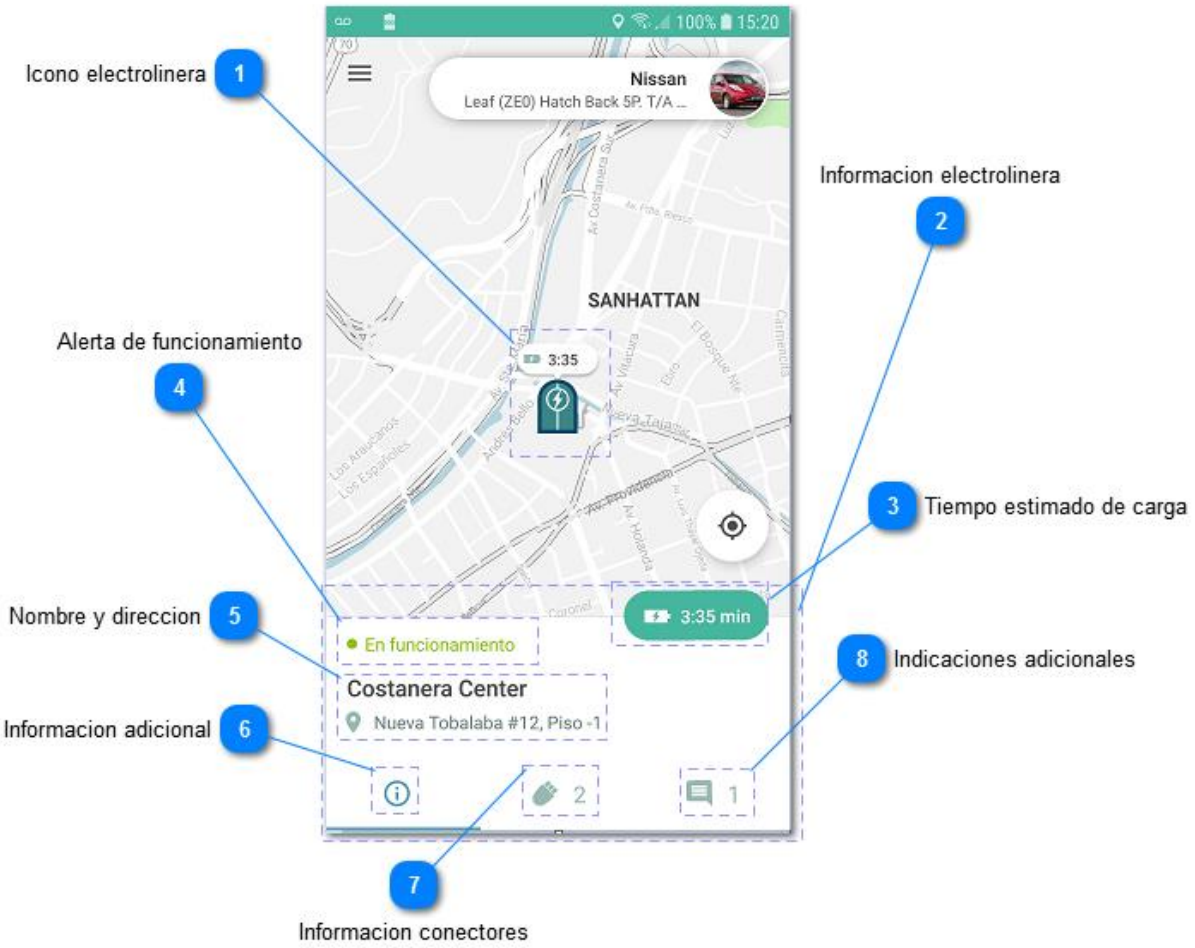


Ilustración 7: Vista al seleccionar una electrolinera

Al presionar la opción de “Información adicional” se despliega nueva información relacionada a la electrolinera. En la Ilustración 8 se muestra la vista que detalla las características de la electrolinera, esta es: tipo de corriente (3), puede ser AC (corriente alterna) o DC (corriente continua), potencia (6) en kilowatts, marca y modelo (2) y horario (5) en caso que esté disponible.

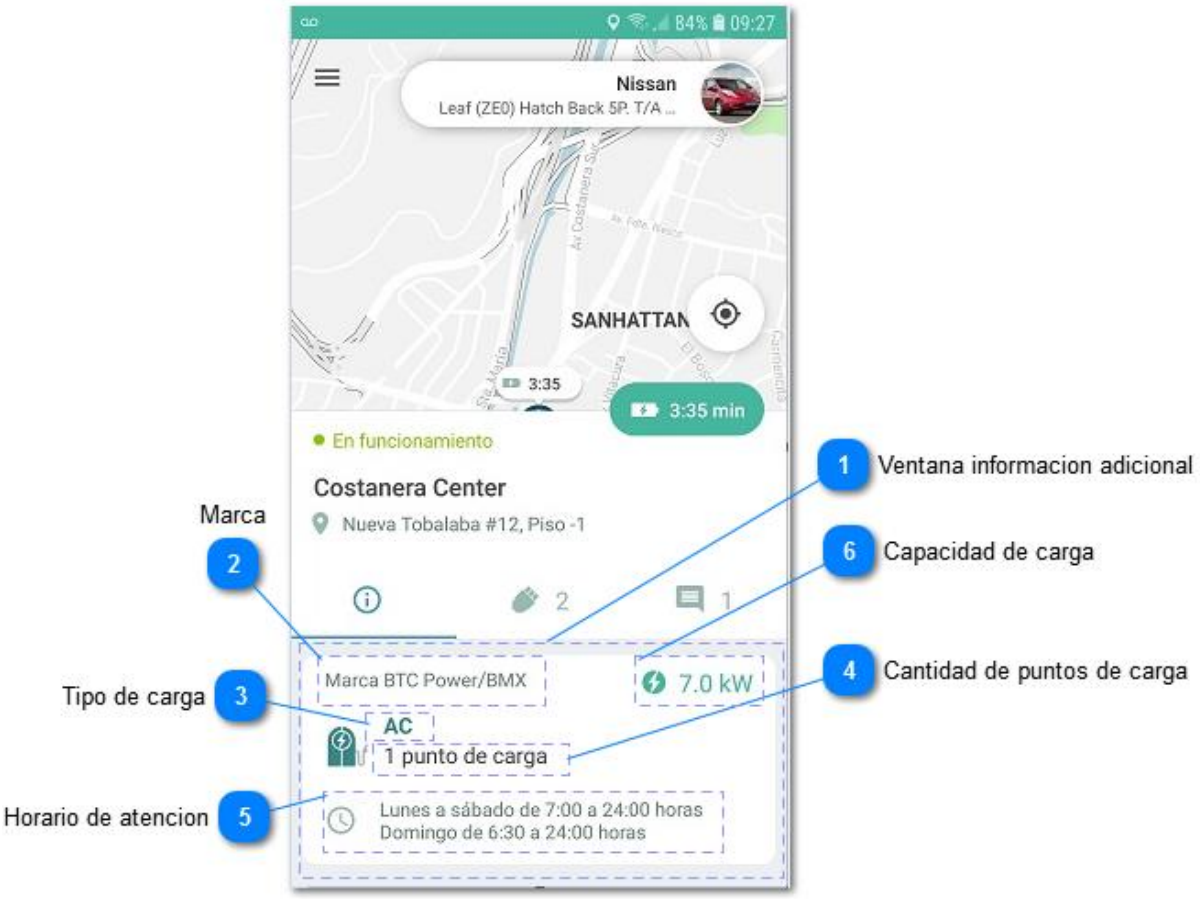


Ilustración 8: Vista de información adicional de la electrolinera.

En la Ilustración 9 se muestra la vista para el detalle de los tipos de conectores disponibles, en ella se provee la siguiente información: tipo de conector (4), si tiene cables disponibles (6), si es compatible con el vehículo del usuario (3), si está habilitado (2) y una imagen del diagrama del conector (4).

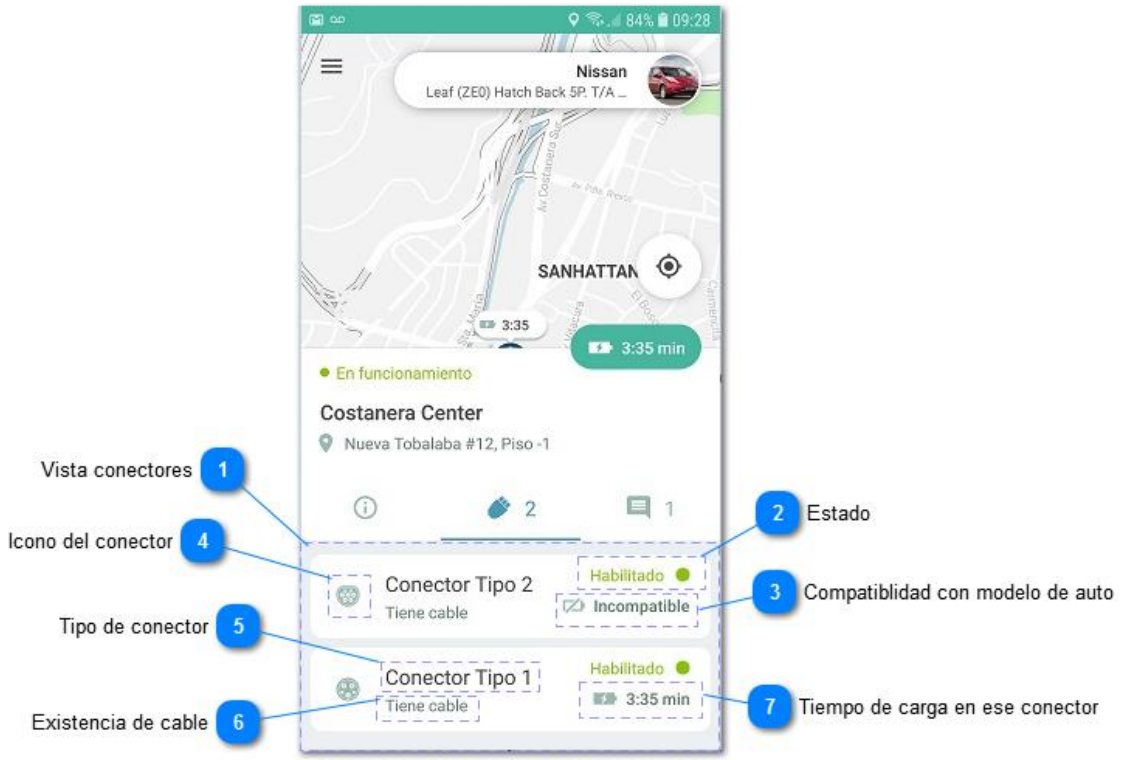


Ilustración 9: Vista de conectores de la electrolinera

Y finalmente en la Ilustración 10 observamos la vista con indicaciones para llegar a la electrolinera o cualquier otro tipo de consejo, por ejemplo, si para llegar a la electrolinera se debe pagar una entrada.

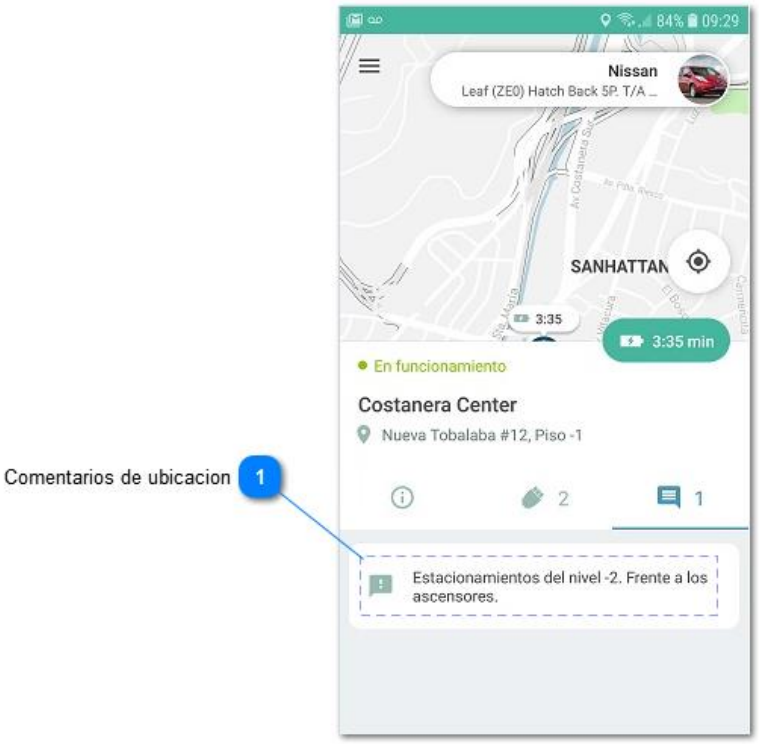


Ilustración 10: Vista de comentarios de la electrolinera.

3.3 Android

Android es un sistema operativo desarrollado por Google, basado en Linux, y fue diseñado para dispositivos móviles, este sistema es el más usado en el mercado chileno con un nivel de penetración 85.3%, lo que implica que una aplicación móvil debe tener una versión para este sistema operativo.

3.3.1 Lenguaje de programación

La versión de la aplicación para dispositivos Android está escrita íntegramente en Kotlin, y fue creada utilizando el IDE oficial de Android, Android Studio. El proyecto está configurado para funcionar con dispositivos Android con un nivel de API superior o igual a 21 (Lollipop), esto representa aproximadamente el 85% de los dispositivos Android.

3.3.2 Arquitectura general

La arquitectura del proyecto sigue los lineamientos de todos los proyectos generados utilizando Android Studio, que separa las clases Java/Kotlin de los recursos gráficos de la aplicación, así como de los archivos de configuración de la misma app.

Los archivos Kotlin están organizados en distintos paquetes, cada uno encargado de encapsular la lógica asociada a uno o más componentes. Los paquetes más relevantes corresponden a los siguientes:

1. ***Database***

Contiene todas las clases asociadas a la base de datos de la aplicación. Esta base de datos está construida sobre la librería Room de Android, que permite definir esquemas de base de datos y genera interfaces para acceder a estos sin tener que preocuparse de la implementación de la base de dato en sí.

Este paquete también contiene la clase DatabaseStartTaskHelper, que tiene la lógica para recuperar y actualizar la base de datos de la aplicación cada vez que el servidor tenga disponible nueva información

2. ***Map***

Contiene la clase MapActivity, encargada de manejar toda la lógica asociada a la vista principal de la aplicación, esto incluye la interacción entre los distintos componentes del mapa (marcadores, burbujas de información), el BottomSheet con información del marcador seleccionado, y el despliegue de la información del auto actual seleccionado por el usuario.

Esta clase obtiene desde la base de datos la información de las electrolineras y las despliega en el mapa.

3. Navigation

Este paquete contiene la Actividad con los términos y condiciones, así como la Actividad NavigationActivity, que contiene la lógica para operar el menú lateral de la aplicación.

4. Network

Contiene todas las clases encargadas de definir y operar la API de conexión con el servidor de la aplicación.

5. SlidingPanel

Toda la lógica de Panel deslizante de la aplicación esta encapsulada en este paquete. Esto incluye los distintos Fragment encargados de cada una de las pestañas del slidingPanel.

6. Splash

Contiene la actividad que es el punto de entrada a la aplicación, la clase SplashScreenActivity. Esta actividad se encarga de obtener la información más actualizada desde el servidor, y redirigir el flujo o a la Actividad principal, o a la actividad de selección de auto según corresponda el caso.

La Ilustración 11 muestra el diagrama de Actividades de la aplicación

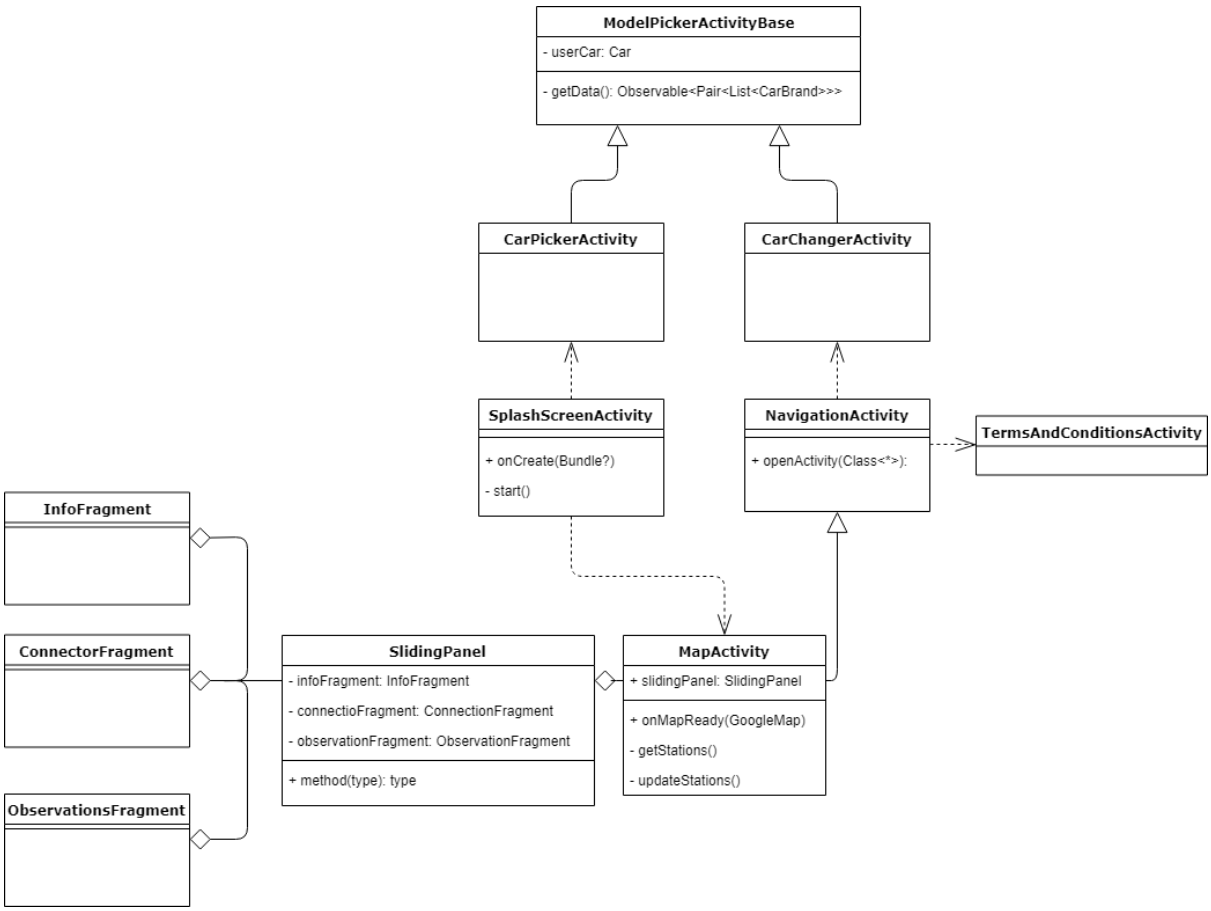


Ilustración 11: Diagrama de Actividades de la aplicación.

3.3.3 Librerías asociadas

Las distintas dependencias utilizadas para compilar la aplicación están definidas en el archivo *build.gradle* del módulo app. Gradle se encarga de descargar y compilar cada una de estas dependencias como parte del proceso de compilación de la aplicación.

Dentro de las librerías utilizadas más importantes están:

- *com.google.android.gms:play-services-maps*: Mapa de Google
- *io.reactivex.rxjava2:rxandroid*: Rx, librería para manejo de eventos
- *com.squareup.retrofit2:retrofit*: Retrofit, librería para la comunicación con el servidor
- *android.arch.persistence.room:runtime*: Room, librería para la base de datos

3.3.4 Datos de entrada

La aplicación no contiene dentro del proyecto ningún archivo con los datos necesarios para poblar la base de datos, esto debido a que esta información es descargada en línea desde el servidor de la aplicación.

Para que la carga de los archivos sea correcta, la aplicación espera que la información enviada por el servidor se encuentre con el formato definido en el archivo ChargingStation, del paquete network/model/stations.

3.3.5 Instalación y configuración en Play Store

La aplicación fue subida a la PlayStore utilizando el proceso estándar de generación de un APK de producción, sin ninguna configuración adicional. El detalle puede ser encontrado en la documentación de google[1]

3.4 iOS

iOS es un sistema operativo desarrollado por Apple, fue diseñado para el iPhone y es el segundo más usado en el mercado con un 15% de penetración, de esta manera sumado con el 85% de Android dominan prácticamente el 100% del mercado de los sistemas operativos para Smartphone. Es importante mencionar que el iPhone es un Smartphone considerado de mayor gamma lo que implica que sus usuarios son en su mayor parte personas con mayor poder adquisitivo, por lo que se toma la decisión de desarrollar EcoCarga también para este sistema entendiendo que en la actualidad los usuarios de autos eléctricos corresponden a este tipo de usuario.

3.4.1 Lenguaje de programación

La versión de la aplicación para dispositivos iOS está escrita íntegramente en Swift 4.0, y fue creada utilizando el IDE oficial de Apple, XCode. El proyecto está configurado para funcionar con dispositivos iOS con un nivel de API superior o igual a 9.0.

[1] https://support.google.com/googleplay/android-developer/answer/113469?hl=es-419

3.4.2 Arquitectura general

La arquitectura del proyecto sigue los lineamientos de todos los proyectos generados utilizando XCode. Los archivos Swift están organizados en distintos grupos, utilizando la misma separación de tareas que se definió en la arquitectura del proyecto Android.

Las mayores diferencias entre ambas arquitecturas están asociadas a las diferencias entre los distintos S.O, así como a las librerías utilizadas, en particular, las más relevantes se detallan a continuación:

- SlidingPanel renombrado a BottomSheet
- Controlador de la base de datos contiene las definiciones de los modelos de la base de datos (Los modelos en si son iguales entre Android e iOS)
- Paquete de navegación es eliminado, dado que la navegación en iOS se explicita por medio de los *interface builders*

El flujo de los ViewControllers de este proyecto es el análogo al flujo de las Actividades del proyecto Android. Este se puede encontrar dentro del archivo Main.storyboard del proyecto, la Ilustración 12, muestra una imagen de como se ve el flujo de la aplicación utilizando la sección Main.storyboard, para más detalles se recomienda entrar a la sección desde el IDE.

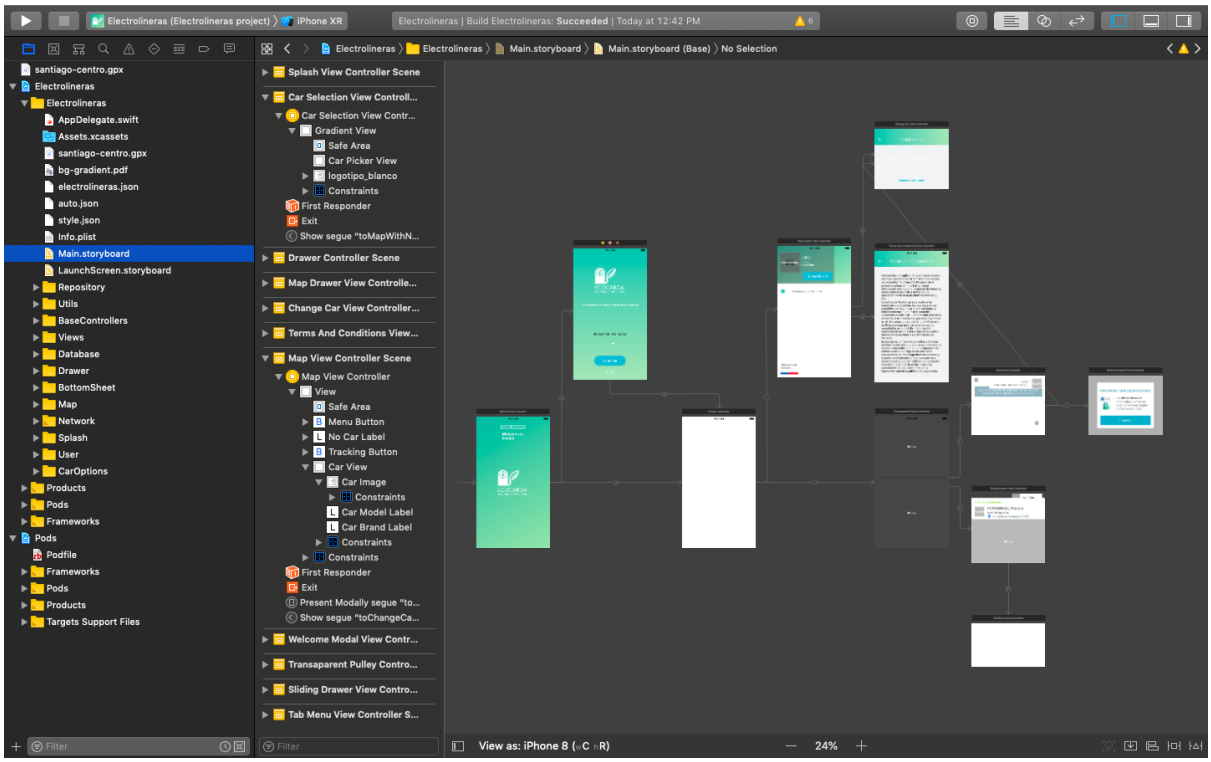


Ilustración 12: Vista del Main.storyboard, donde se detalla el flujo de la aplicación

3.4.3 Librerías asociadas

Las distintas dependencias utilizadas para compilar la aplicación estan definidas en el archivo Podfile. CocoaPods se encarga de descargar y compilar cada una de estas dependencias como parte del proceso de compilación de la aplicación.

Dentro de las librerías utilizadas más importantes están:

- GoogleMaps: Mapa de Google
- RxSwift: librería para manejo de eventos
- Moya: librería para la comunicación con el servidor

- SQLite.swift: librería para la base de datos

3.4.4 Datos de entrada

La aplicación no contiene dentro del proyecto ningún archivo con los datos necesarios para poblar la base de datos, esto debido a que esta información es descargada en línea desde el servidor de la aplicación, al igual que en el proyecto Android.

El formato esperado por la aplicación de la información que envía el servidor es el mismo formato que se especificó en el proyecto Android.

3.4.5 Instalación y configuración en App Store

La aplicación fue subida a la AppStore utilizando el proceso estándar de generación de un Archive Bundle de producción, sin ninguna configuración adicional. Este proceso debe hacerse desde un Mac con las credenciales del proyecto, como es estándar para proyectos iOS[2].

[2] https://developer.apple.com/ios/submit/

4. Servidor

Como se indicó en la sección de arquitectura global del sistema, existe un servidor encargado de centralizar y disponibilizar la información de las electrolineras, modelos de autos y tipos de conectores. La lógica está implementada sobre el *framework* Django (lenguaje de programación Python) y utiliza una base de datos PostgreSQL y el servidor APACHE, todo sobre una plataforma Linux (Ubuntu).

Este servidor a la fecha se compone principalmente de dos módulos, el primero corresponde a un administrador que provee de una interfaz para poder editar, agregar o borrar datos que muestra la aplicación móvil y el segundo módulo es una api de libre acceso (no requiere permiso) que disponibiliza la información utilizada por la aplicación móvil, es un canal solo de lectura.

4.2 Administrador

4.2.1 La herramienta

Como se mencionó en la introducción de este capítulo esta herramienta tiene por objetivo permitir la actualización de los datos del sistema para un usuario no experto del área de computación. Actualmente el administrador puede ser accedido con los siguientes datos:

- URL: https://admin.ecocarga.cl/
- Usuario: ecocarga
- Contraseña: PACELyG4kcFMY8U2

Al entrar, la primera vista corresponde al menú principal del sistema (Ilustración 13), desde donde se puede dirigir hacia las distintas entidades que componen el modelo de datos. La primera entidad corresponde a los modelos de autos (1), la segunda a las compañías dueñas de las electrolineras (2), la tercera son las máquinas electrolineras (3), la cuarta corresponde a todas las marcas de los vehículos eléctricos (4) y finalmente la quinta entidad corresponde a los tipos de conectores soportados por el sistema (5). Por otra parte, el punto seis (6) señala los botones asociados para la función de agregar un nuevo elemento a la entidad correspondiente. Los botones de modificar llevan a la misma vista que al clickear directamente en los links de las entidades donde se muestra la lista de entidades existentes.

Ilustración 13: Vista inicial de herramienta de edición.

A continuación, se realiza una descripción detallada de cada una de las vistas mencionadas en el párrafo anterior.

Autos (Añadir)

La primera información descrita corresponde a los detalles de los modelos de autos que podría tener un usuario, la Ilustración 14 muestra los datos necesarios para añadir un nuevo modelo de automóvil al sistema.

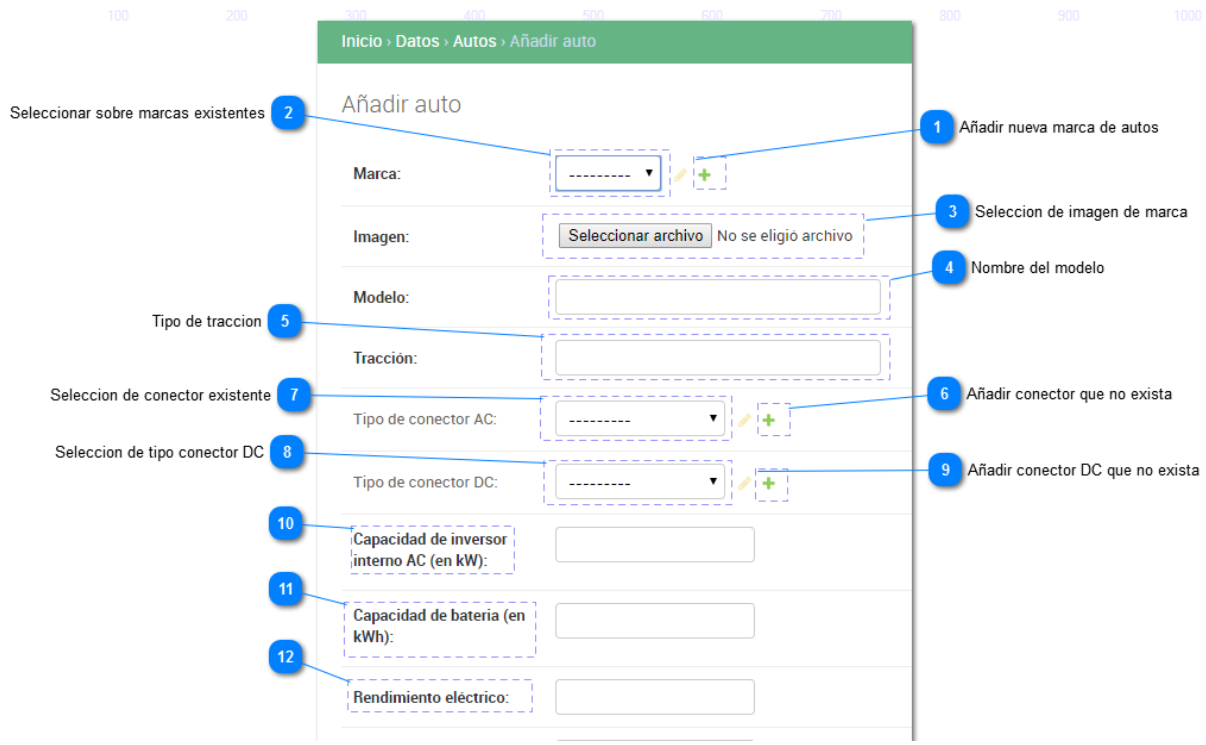


Ilustración 14: Vista para agregar un nuevo modelo de auto al sistema.

La primera información corresponde a la marca del nuevo modelo, se puede seleccionar una ya existente en el sistema (2) o se puede realizar el proceso de incorporación de una nueva (1), le sigue el ingreso de una imagen para el modelo de auto (3), esta imagen será mostrada en la aplicación móvil tal como se muestra en la Ilustración 4. Estas imágenes deben tener un tamaño fijo de 512x512 pixeles.

Autos (Modificar)

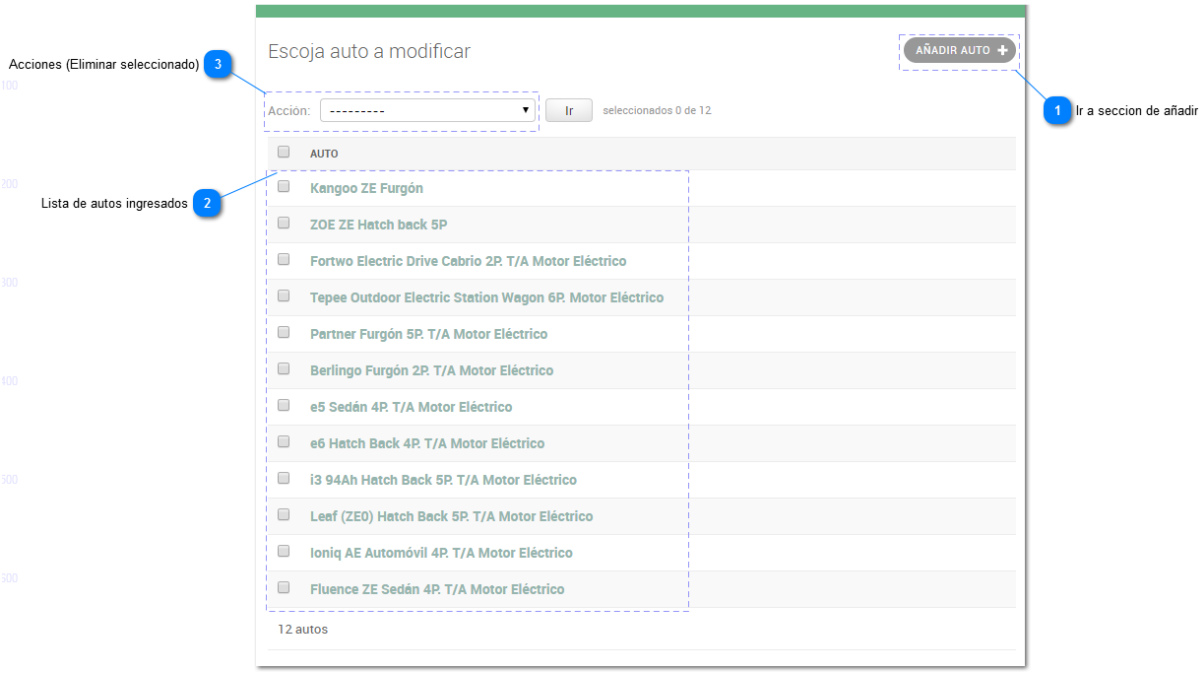


Ilustración 15: Vista para editar autos ya incorporados al sistema.

En esta sección (ver Ilustración 15) se muestra una lista con todos los modelos de autos ya ingresados al sistema (2), pueden ser seleccionados para borrarlos (3), o editar sus características usando el mismo formulario de creación ya descrito en la Ilustración 14. A medida que existan más modelos de autos la lista se irá paginando automáticamente de manera tal de mantener siempre 20 registros por página.

Cada vez que un modelo de auto es editado o añadido se actualiza un registro de versión de datos consultado por la aplicación móvil cada vez que inicia, esto permite que los usuarios accedan a la nueva información en un período pequeño de tiempo (en la siguiente apertura de la aplicación móvil).

Compañías electrolineras (Añadir)

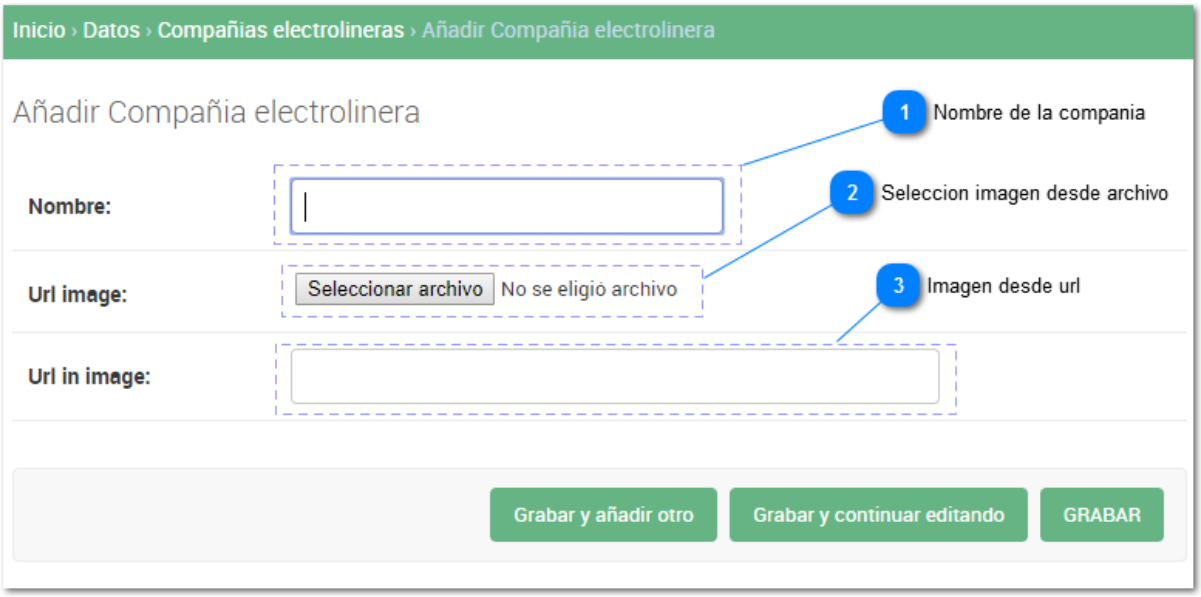


Ilustración 16: Vista de creación de compañía.

Como muestra la Ilustración 16, la compañía debe tener un nombre para identificarla, se recomienda un nombre corto dado que este aparecerá en la aplicación móvil. Adicional al nombre se debe adjuntar una imagen o logo de la compañía con dimensiones de 512x512 pixeles. El tercer campo (3) corresponde a una url asociado a la compañía, como su sitio web corporativo, actualmente este dato no es usado por la aplicación móvil, se creó con el objetivo (a futuro) de disponibilizar información del dueño de la electrolinera en la aplicación.

Compañías electrolineras (Modificar)

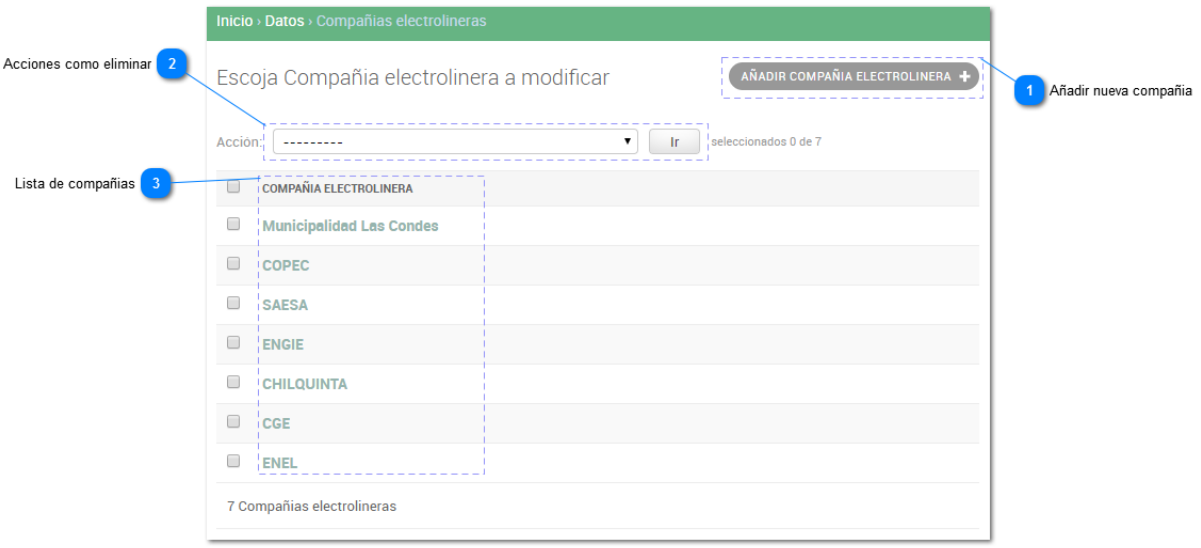


Ilustración 17: Vista de selección de compañía para edición.

La Ilustración 17 muestra la vista de selección de las compañías ya ingresadas y que pueden ser modificadas. La vista también provee de un link directo para añadir una nueva compañía (1), las acciones a la fecha solo proveen la acción de eliminar una compañía seleccionada.

Electrolineras (Añadir)

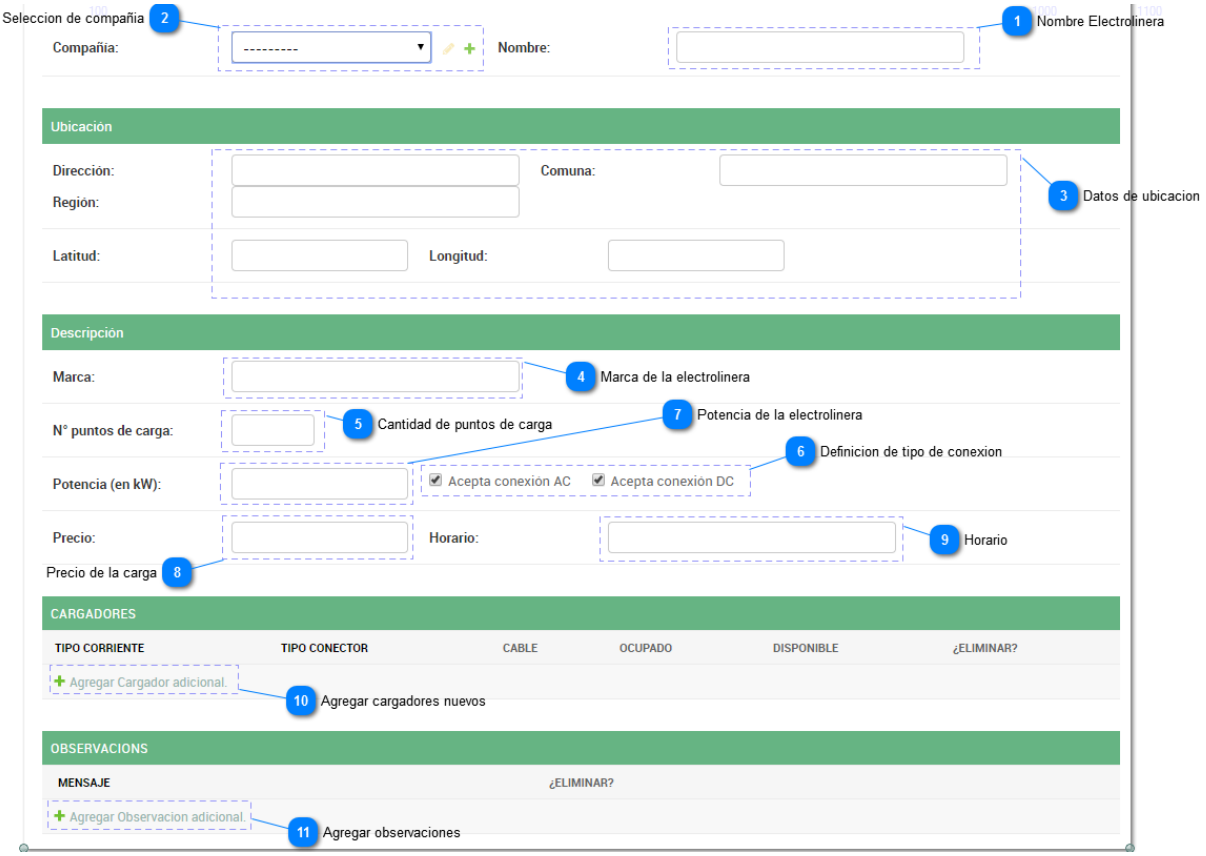


Ilustración 18: Vista de creación de una electrolinera.

La Ilustración 18 muestra la vista para la creación de una electrolinera, esta vista está dividida en cinco secciones: la primera sección corresponde al nombre de la electrolinera y la compañía a la que pertenece, la segunda sección es para ingresar los datos de ubicación de la electrolinera (3): dirección, comuna, región, latitud y longitud. La tercera sección corresponde a los datos técnicos de cada electrolinera: marca, cantidad de puntos de carga, potencia, tipo de conexión AC/DC, horario. También está la opción de agregar el precio, pero hoy en día no es un dato visible en la aplicación móvil. La cuarta sección corresponde a la información de los conectores que existen en la electrolinera, la Ilustración 19 muestra en detalle un ejemplo de una electrolinera que tiene 3 conectores, cada uno de ellos debe definir su tipo de corriente (1), el tipo de conector (2), en caso que no exista el tipo de conector requerido se puede agregar (3), si tiene o no cables disponibles en la electrolinera (4), si está o no disponible para su uso (6) . Finalmente, existe un cuadro de confirmación para poder eliminar conectores (7).

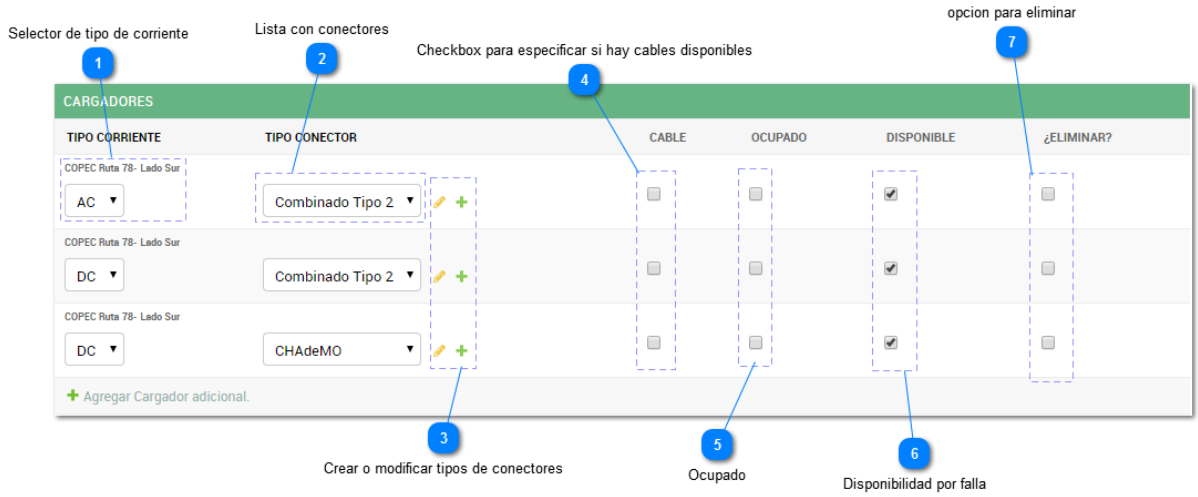


Ilustración 19: Sección para agregar conectores a la electrolinera.

Por último, la quinta sección corresponde al ingreso de observaciones, estas observaciones están pensadas para comentarios que ayuden al usuario de la aplicación móvil a encontrar la electrolinera, puesto que algunas de ellas se encuentran en espacios cerrados, por ejemplo, para indicar el piso en donde se encuentra la electrolinera, si es necesario pagar una entrada o estacionamiento, entre otros.

Electrolineras (Modificar)

Para editar una electrolinera ya creada, se debe acceder primero a la lista de electrolineras, esto se hace haciendo clic en “modificar” en la página principal (Ilustración 13). La página de edición es igual a la de creación, pero con valores ingresados anteriormente.

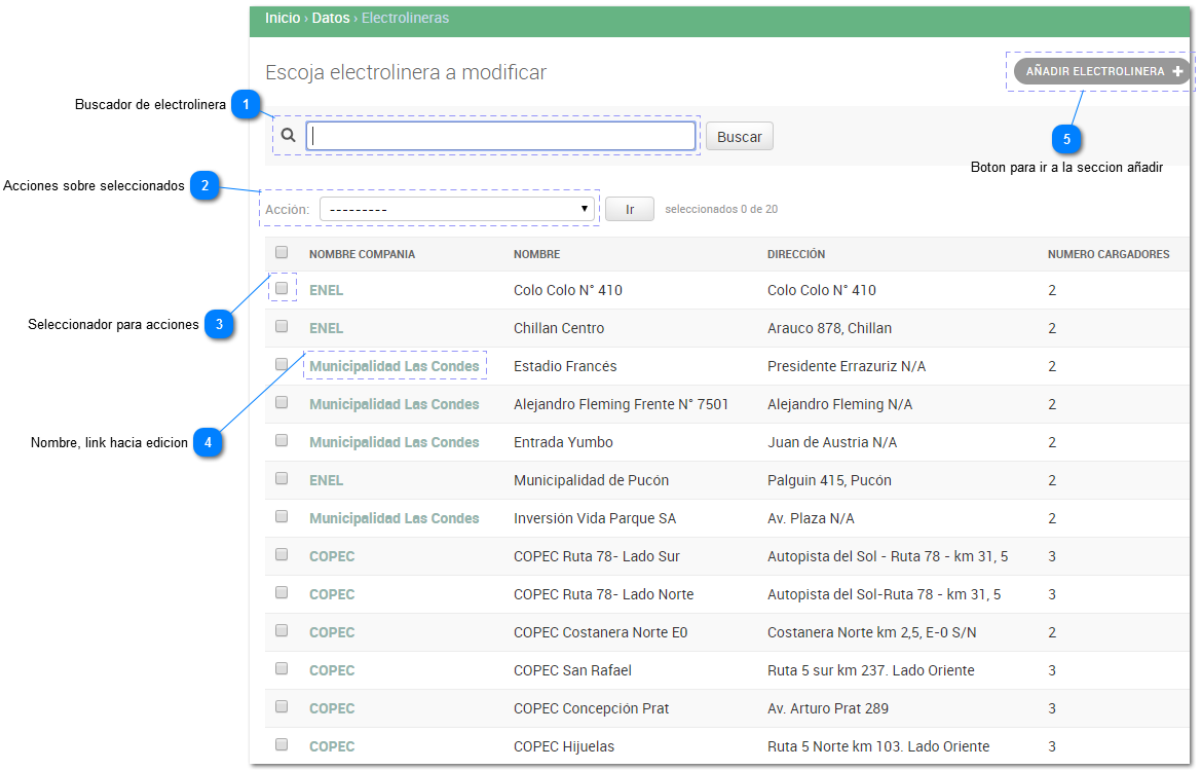


Ilustración 20: Vista para buscar y seleccionar electrolinera para editar.

Como se aprecia en la Ilustración 20, la vista contiene un buscador de electrolineras (1) vía texto, una lista de acciones (2) para realizar sobre las electrolineras seleccionadas, a la fecha la única acción disponible es eliminar una o más electrolineras. La lista de electrolineras trae una breve descripción de cada una (nombre, dirección, numero de cargadores) y un link en el nombre que permite editar sus valores.

Cada vez que una electrolinera es editada o añadida se actualiza un registro de versión de datos consultado por la aplicación móvil cada vez que inicia, esto permite que los usuarios accedan a la nueva información en un período pequeño de tiempo (en la siguiente apertura de la aplicación móvil).

Marca de Autos (Añadir)

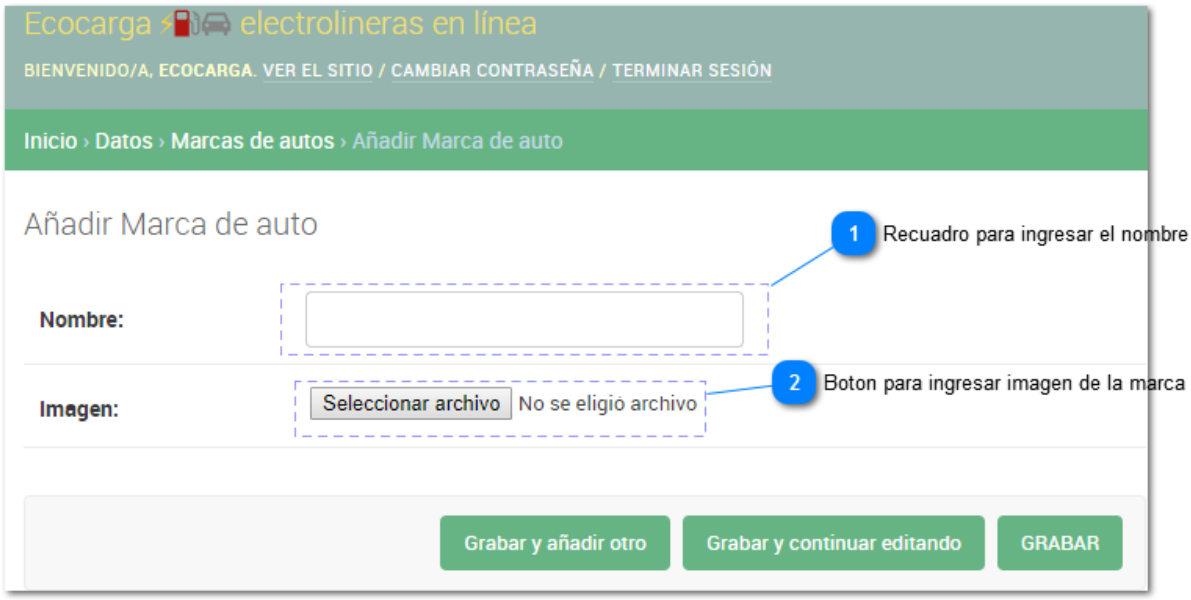


Ilustración 21: Vista para ingresar una nueva marca de automóvil.

La Ilustración 21 muestra la vista para el ingreso de una nueva marca de auto, esto es necesario para el caso de ingresar un nuevo tipo de automóvil de una marca no existente en el sistema, la vista contiene un recuadro para ingresar la marca del auto (1) y un botón para ingresar un logo o la imagen que represente la marca, esta será usada en la aplicación móvil. La imagen debe tener un tamaño de 640x410 pixeles.

Marca de Autos (Modificar)

De la misma manera que el resto de las entidades, para poder editar se debe ingresar a en la sección “marca de autos” a través de la página principal con el botón modificar, esto llevará al usuario a la lista de marcas actuales en el sistema, tal como muestra la Ilustración 22.

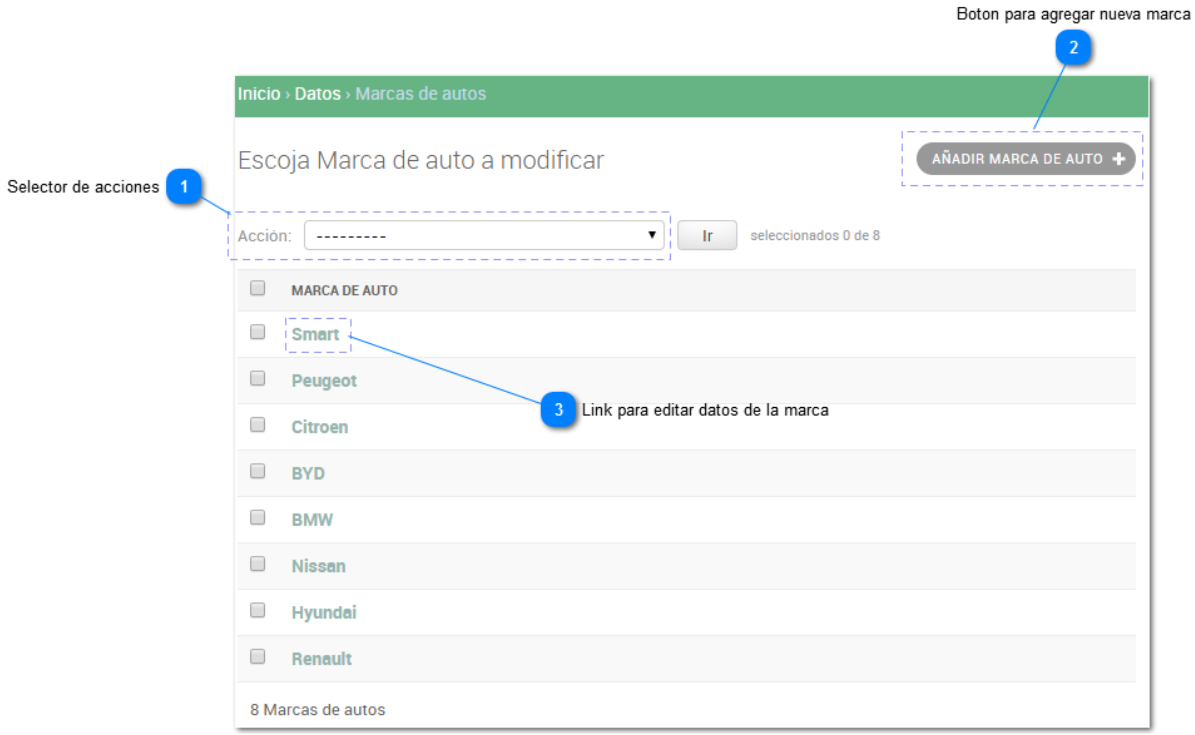


Ilustración 22: Vista con listado de marcas de autos para seleccionar y editar.

Tipo de Conectores (Añadir)

La última entidad necesaria para el sistema corresponde a los tipos de conectores, de la página principal se puede ingresar un nuevo tipo de conector al hacer clic en el botón *Añadir*, este link llevará al usuario a la vista de añadir tal como muestra la Ilustración 23.

Ilustración 23: Vista para añadir un nuevo tipo de conector

Como se muestra en la Ilustración 23, la vista solo provee de un recuadro donde se debe ingresar el nombre del nuevo tipo de conector, es muy importante que este nombre sea representativo del mismo ya que será el nombre que verán los usuarios en la aplicación móvil.

Tipo de Conectores (Modificar)

Siguiendo la misma mecánica de las entidades anteriores, si en la página principal se ingresa al link "*Modificar"* de la sección "tipos de conectores" se llevará al usuario a la lista de tipos de conectores existentes, donde el usuario puede editar.

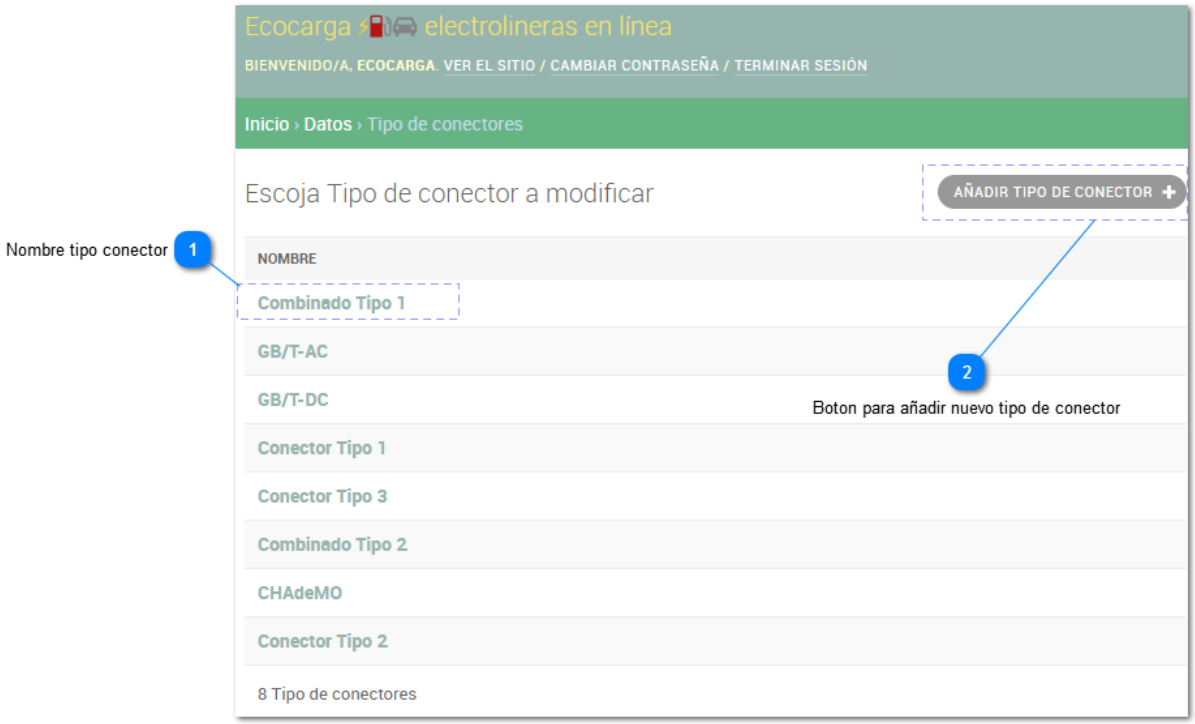


Ilustración 24: Vista de lista para editar tipos de conectores.

La Ilustración 24 muestra el listado de tipos de conectores a la fecha inscritos en la plataforma, cada conector muestra su nombre desde el cual se puede ingresar a la vista de edición, además en caso que no exista un conector existe un botón que llevara al usuario directo a la vista de añadir (2) ya descrita en la sección anterior.

4.2.2 Configuración del administrador

El administrador está construido sobre el sistema de administración que provee el framework Django, así como el sistema de autenticación que permite acceder a él por lo que su construcción se basa en las siguientes configuraciones del framework[3]:

- Archivo data/static/css/adminextra.css: define la paleta de colores y componentes que pintará en cada vista
- Archivo data/admin.py: define los formularios para editar los distintos datos presentes en el modelo
- Archivo templates/admin/base_site.html: define la estructura base del administrador
- Archivo tempaltes/admin/rest_framework/api.html: define la estructura base de las vistas para la API que usa el teléfono

4.2.3 Usuarios

Para acceder al administrador se requiere conocer un nombre de usuario y contraseña por lo que se necesita un proceso de creación de estos en la plataforma. Actualmente es proceso se debe realizar con un comando de consola que provee el framework Django.

Para crear un usuario es necesario estar en la carpeta del proyecto web y ejecutar la siguiente sentencia: “python manage.py createsuperuser”. Luego se solicitarán los siguientes datos:

- Nombre de usuario
- Correo electrónico
- Contraseña

Es importante hacer notar que todos los usuarios son iguales, es decir, no hay permisos asociados a ellos por lo que todos pueden crear, editar o eliminar cualquier registro presente en la base de datos.

4.3 Datos de entrada

Los datos iniciales que requiere la plataforma se encuentran almacenados dentro del proyecto, estos corresponden a los registros de la base de datos y las imágenes usadas por la aplicación móvil.

Los registros de la base de datos se encuentran en el archivo “data/fixtures/initial_data.json” en formato json, mientras que las imágenes están en la carpeta “media”. Existen tres tipos de imágenes: aquellas usadas para el logo de las empresas proveedoras de electrolineras (“media/compania”); las imágenes usadas para las marcas de vehículos (“media/marca”) y, por último, las usadas para identificar visualmente cada modelo de vehículo (“media/modelo”).

[3] Las rutas que aparecen están construidas asumiendo que parten desde la carpeta raíz del proyecto

4.4 Modelo de datos

El modelo de datos que usa el servidor se divide en dos: el primero corresponde a las tablas requeridas por el negocio (administrador y aplicación móvil) y el segundo corresponde a los requeridos por el sistema de administración (sistema de usuarios). En la Ilustración 25 se muestran las ocho (8) tablas requeridas para proveer los datos que necesita la aplicación móvil y la relación entre estás.

Existen dos modelos principales: data_electrolinera y data_auto. Una electrolinera tiene asociada una empresa que actúa como dueña de ella (tabla data_compania), cero o más observaciones (tabla data_observacion) y uno o más cargadores (tabla data_cargador). Cada cargador representa un punto de carga para un vehículo y tiene asociado un tipo de conector. Por otro lado, la tabla data_auto tiene asociada una marca (tabla data_marcaauto), y un tipo de conector (tabla data_tipoconector) para conexiones con corriente AC y otro tipo de conector para conexiones con corriente DC.

La tabla “data_version” contiene solo un registro y es el responsable de almacenar un identificador único que cambia cada vez que los datos son modificados (durante la edición o creación) en el administrador web.

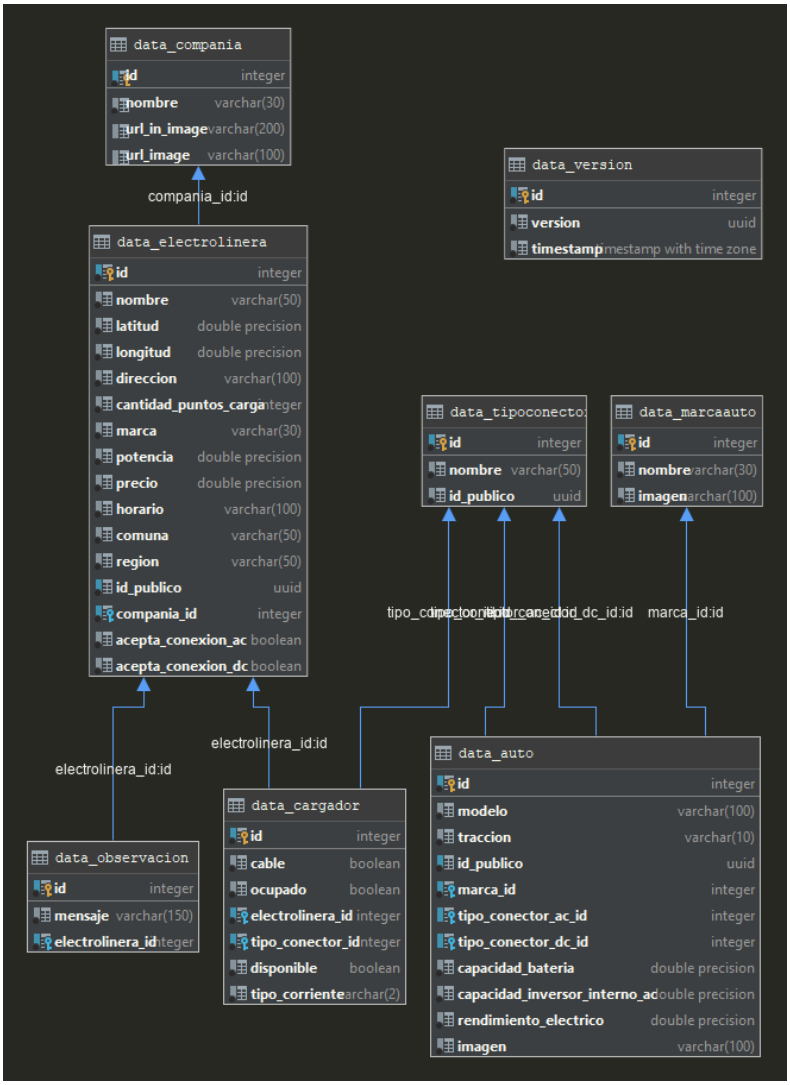


Ilustración 25: modelo de datos requerido por la aplicación móvil

La segunda parte del modelo corresponde a las tablas necesarias para mantener el administrador, usuarios y acceso web. Son generadas por el framework Django y algunas de ellas no son usadas en el proyecto. En la Ilustración 26 muestra las tablas creadas por el framework y las relaciones entre ellas.

Las tablas usadas en el proyecto son las siguientes:

- **auth_user:** almacena los usuarios que pueden acceder al adminstrador web
- **django_admin_log:** almacena las acciones realizadas por los usuarios en el administrador
- **django_content_type** y **django_migrations:** administra la representación (definición y cambios) de las tablas en el código del proyecto[4]
- **django_sesion:** registra las sesiones web que los usuarios inician en el administrador

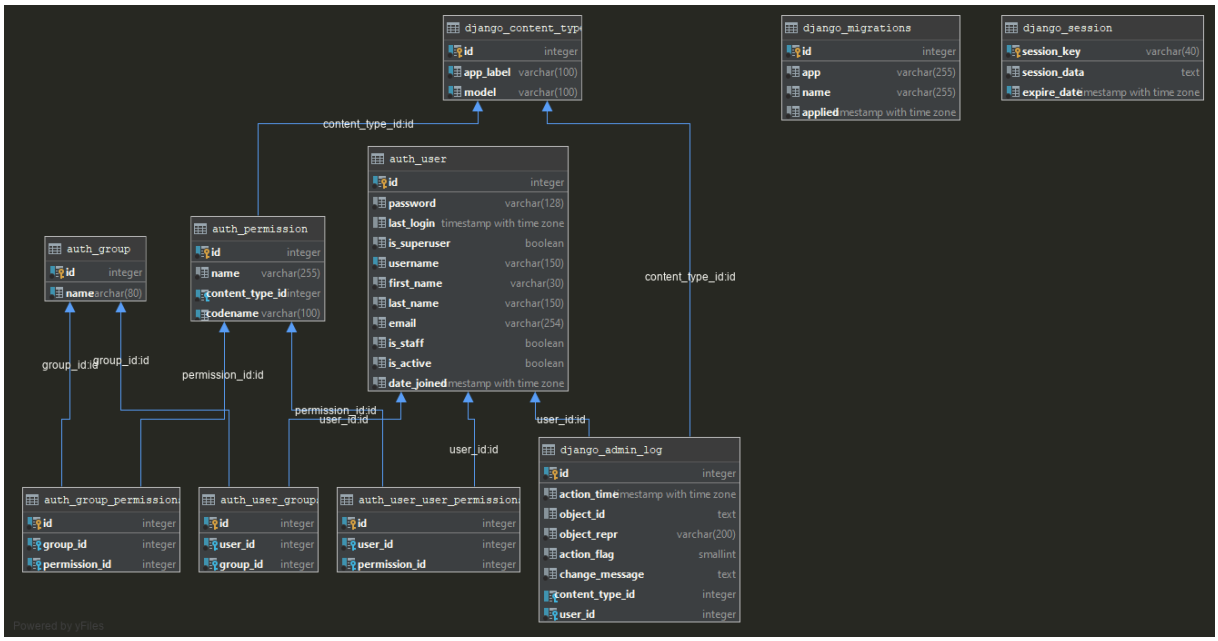


Ilustración 26: modelo de datos requerido para el administrador web

Las tablas no usadas en el proyecto corresponden a un sistema de permisos que regula los accesos (creación, edición o eliminación) que pueden tener los usuarios a cada modelo del proyecto. Para más detalles consultar información del framework Django en el siguiente link: https://docs.djangoproject.com/en/2.2/topics/auth/default/#topic-authorization

4.5 Descripción del proyecto

En la Ilustración 27 se muestra la estructura de archivos usados en el proyecto web, la organización de los archivos es el usado por defecto en proyectos que usan el framework Django. La carpeta “ecocargaWeb”, que tiene el mismo nombre que el proyecto (carpeta raíz), contiene los archivos de configuración centrales, estos son:

[4] Consultar documentación sobre el ORM del framework Django: https://docs.djangoproject.com/en/2.2/topics/db/models/

- settings.py: contiene la configuración de acceso para la base de datos, descripción de dependencias, configuración de logs, configuración de rutas de acceso para imágenes, zona horaria utilizada para las fechas del proyecto
- urls.py: contiene la configuración de las urls
- wsgi.py: punto de entrada al proyecto web[5]

La carpeta “data” es una aplicación Django y su función es contener la definición de los datos (models.py), lógica de administración (admin.py), vistas para acceder a los datos mediante API-REST (views.py), una descripción para la publicación web de los datos (serializers.py) y almacenar los datos iniciales (initial_data.json).

El resto de los archivos y carpetas cumplen las siguientes funciones:

- data/migrations: contiene archivos creados por el framework Django (no deben ser modificados a mano) que usa su ORM (Object Relational Mapping) para sincronizar la definición de las estructuras de datos (modelos) del código con el de la base de datos.
- data/static/css/adminextra.css: contienen estilos que definen los colores del administrador
- data/apps.py: archivo creado por el framework Django, no se usa
- data/tests.py: contiene una batería de pruebas para validar el correcto comportamiento del código, específicamente prueba el código de los archivos data/models.py, data/serializers.py y data/views.py
- logs: la carpeta contiene el archivo file.log que almacena los errores que se generan en el código, en el archivo ecocargaWeb/settings.py se define su uso
- media: almacena todos los recursos visuales, específicamente las imágenes que usa la aplicación móvil y que son subidas a través del administrador web
- static: contiene todos los recursos estáticos como archivos javascript, css o imágenes, es usado en el ambiente de producción
- templates: define la estructura visual de las vistas usadas por el administrador web
- .env: contiene variables de configuración como: si la aplicación está en modo producción o debug (DEBUG), a través que nombres de dominio se puede acceder a él (HOSTS), configuración de la base de datos (DB_ENGINE, DB_NAME, DB_USER, DB_PASSWORD, DB_HOST, DB_PORT) y una llave secreta (SECRET_KEY) usada internamente por el framework como un Salt[6]. Todas son usadas en el archivo ecocargaWeb/settings.py
- requeriments.txt: contiene las dependencias (librerías externas) del proyecto
- manage.py: es generado por el framework Django y es usado para la ejecución de comandos predefinidos o aquellos creados por desarrolladores[7]

[5] https://wsgi.readthedocs.io/en/latest/what.html
[6] https://en.wikipedia.org/wiki/Salt_(cryptography)
[7] https://docs.djangoproject.com/en/2.2/howto/custom-management-commands/

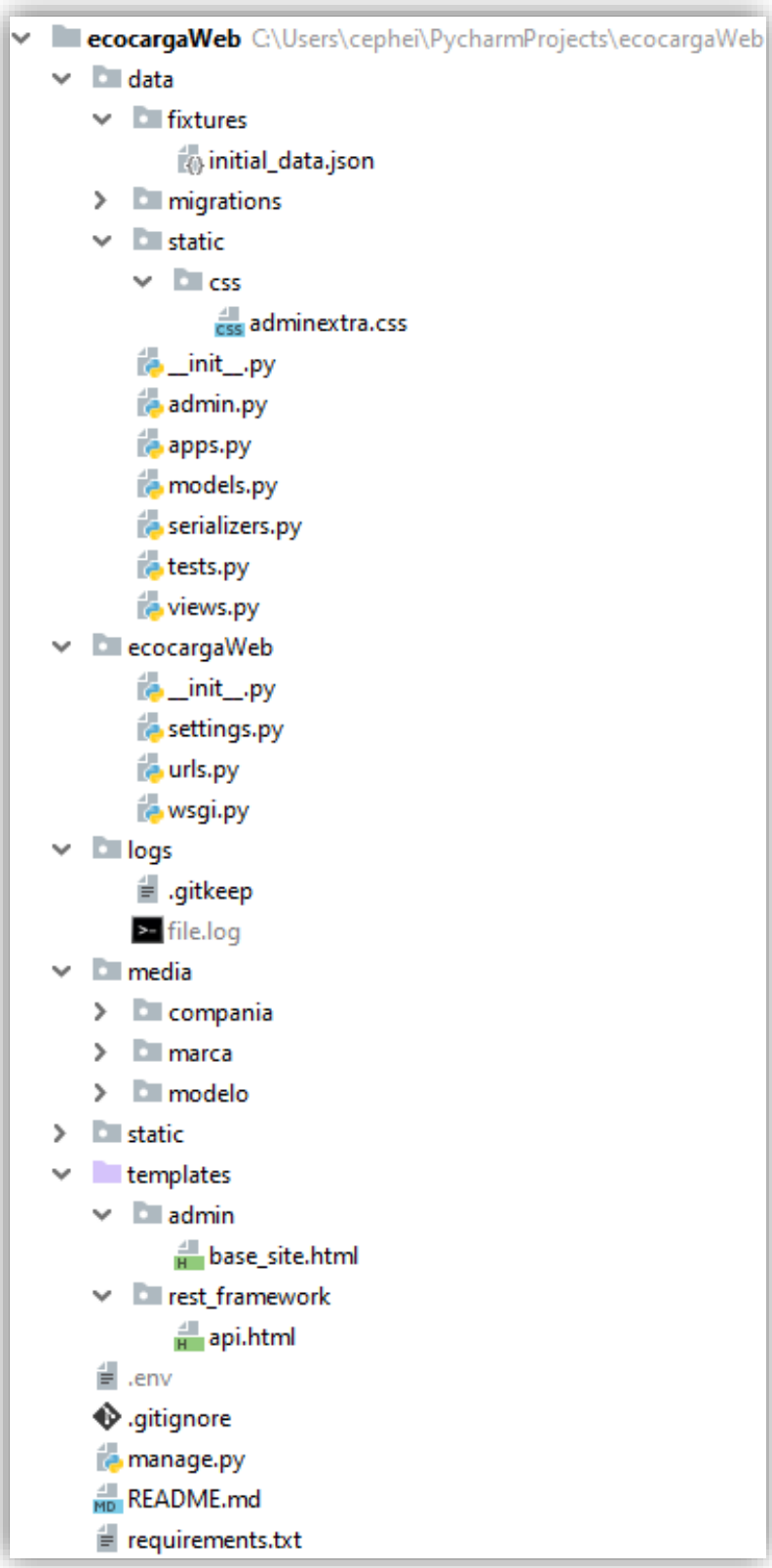


Ilustración 27: estructura de archivos del proyecto web

4.6 API REST para aplicación móvil

El teléfono se comunica con el servidor a través de una API REST que no requiere permiso para su lectura y disponibiliza tres (3) recursos:

- versión: entrega información de la última vez que los datos cambiaron y un código de versión que representa esa marca de tiempo
- electrolineras: entrega una lista de las electrolineras disponibles en el servidor
- autos: entrega una lista de todos los vehículos disponibles en el servidor

Los datos de cada recurso son entregados en formato “json”. En la Tabla 2~~Tabla 2~~ se muestra un ejemplo de una llamada a cada recurso y su respuesta.

Recurso	Respuesta

GET /v1/version/	{
	"version": "b60f76ea-1ef9-46d2-86e7-1add064161e0",
	"timestamp": "2019-05-13T10:47:00.044164-04:00"
	}

GET /v1/autos/	[
	{
	"nombre": "Smart",
	"imagen": "https://api.ecocarga.cl/media/marca/smart.jpg",
	"modelos": [
	… ,
	{
	"modelo": "Fortwo Electric Drive Cabrio 2P.",
	"traccion": "4x2",
	"tipo_conector_ac": {
	"nombre": "Conector Tipo 2",
	"id_publico": "03818088-46cb-409a-b461-b46226995f2c"
	},
	"tipo_conector_dc": null,
	"capacidad_inversor_interno_ac": 5.0,
	"capacidad_bateria": 17.6,
	"rendimiento_electrico": 7.1,
	"imagen": "https://api.ecocarga.cl/media/modelo/ca.jpg",
	"id_publico": "9c41a24c-0b59-4b25-bc9a-328ae6c2100f"
	},
	…
	]
	},
	…
	]

GET /v1/electrolineras/	[
	…,
	{
	"nombre": "CGE Rancagua",
	"latitud": -34.1688448043231,
	"longitud": -70.7258785536954,
	"direccion": "Av. El Trebol #617",
	"cantidad_puntos_carga": 1,
	"marca": "Siemens",
	"potencia": 7.0,
	"precio": 0.0,
	"horario": "-",
	"comuna": "Rancagua",
	"region": "O'higgins",
	"compania": {
	"nombre": "CGE",
	"url_image": null,
	"url_in_image": null
	},
	"observaciones": [
	"subir al 6to piso"
	],
	"cargadores": [
	{
	"tipo_conector": {
	"nombre": "Conector Tipo 1",
	"id_publico": "3b52f342-ac1d-4313-bc6e-7bd86151ba35"
	},
	"tipo_corriente": "ac",
	"cable": true,
	"ocupado": false,
	"disponible": true
	}
	],
	"acepta_conexion_ac": true,
	"acepta_conexion_dc": false
	},
	]

Tabla 2: Ejemplo de respuesta *para cada recurso de la API REST*

4.7 Instalación

A continuación, se describe el proceso de instalación del proyecto web en un servidor con sistema operativo Ubuntu 18 y python3. Si se utiliza otra versión de Linux es importante notar que algunas operaciones o comandos pueden variar.

Lo primero que se debe realizar es una actualización del sistema operativo y luego instalar las dependencias de software del proyecto, que corresponden a Python3, Postgres y APACHE.

```
sudo apt-get update
sudo apt-get upgrade

sudo apt-get –yes –force-yes install apache2 git python-setuptools libapache2-mod-wsgi-py3
python-dev postgresql postgresql-contrib
```

Con Python, Postgres y APACHE instalados se procede a crear la base datos:

```
sudo – postgres psql –c “CREATE DATABASE db”
sudo – postgres psql –c “CREATE USER ecocarga WITH PASSWORD ‘ecocarga’”
sudo – postgres psql –c “GRANT ALL PRIVILEGES ON DATABASE db TO ecocarga”
```

A continuación, se debe elegir una ruta para almacenar el proyecto web, para este ejemplo se usa la ruta “/home/user” (se asume que existe el usuario “user”). Según lo anterior, la raíz del proyecto se encuentra en “/home/user/ecocargaWeb”.

El siguiente paso es crear un entorno virtual en python para aislar las dependencias del proyecto de otras que puedan estar presentes en el servidor:

```
virtualenv –-python=/usr/bin/python3 venv

# activar entorno virual
source venv/bin/activate
```

Una vez activado el entorno virtual se deben instalar las dependencias del proyecto descritas en el archivo requirements.txt.

```
pip install –r requirements.txt
```

La configuración del proyecto debe indicar los parámetros de conexión a la base de datos, una llave secreta y los dominios permitidos para operar correctamente, para esto se debe crear un archivo en la raíz del proyecto con el nombre “.env” y su contenido debe ser el siguiente:

```
archivo: .env
---

DEBUG=False

SECRET_KEY='*h512l-5hftzu3v8e=$7f#52^3g7@ln*h6gd^dk!8nw=2xpg3h'
```

```
ALLOWED_HOSTS=‘api.ecocarga.cl,admin.ecocarga.cl’

DB_ENGINE='django.db.backends.postgresql'


DB_NAME='bd’
DB_USER='ecocarga'
DB_PASSWORD='ecocarga'
DB_HOST='localhost'
DB_PORT='5432'
```

Con los pasos anteriores realizados podemos construir las tablas y poblar con los datos iniciales el proyecto web, esta tarea se realiza a través de los comandos que provee el framework Django:

```
# crear tablas en la base de datos
python manage.py migrate

# cargar datos iniciales
python manage.py loaddata initial_data
```

Los datos iniciales almacenan un usuario[8] para acceder al administrador pero es posible crear uno nuevo siguiendo los pasos descritos en la sección 4.2.3.

Con lo realizado hasta este punto el servidor está listo para ser desplegado a través del protocolo HTTP y HTTPS, para esto se usa el software APACHE y su configuración se realiza en el archivo “/etc/apache2/sites-available/ecocarga.conf” y debe tener el siguiente contenido:

```
LoadModule wsgi_module /usr/lib/apache2/modules/mod_wsgi.so
LoadModule ssl_module /usr/lib/apache2/modules/mod_ssl.so
LoadModule rewrite_module /usr/lib/apache2/modules/mod_rewrite.so

WSGIApplicationGroup %{GLOBAL}
WSGIDaemonProcess ecocarga python-path=/home/user/ecocargaWeb python-
home=/home/user/ecocargaWeb/venv processes=3 threads=30 display-name='srvr-
ecocarga' user="user" group="user"

<VirtualHost *:80>
    ServerName ecocarga.cl
    ServerAlias www.ecocarga.cl

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime

    Redirect permanent / https://www.ecocarga.cl/
```

[8] Nombre de usuario: “ecocarga”, y contraseña: “PACELyG4kcFMY8U2”

```
</VirtualHost>

<VirtualHost *:80>
    ServerName api.ecocarga.cl

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime

    Redirect permanent / https://api.ecocarga.cl/
</VirtualHost>

<VirtualHost *:80>
    ServerName admin.ecocarga.cl

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime

    Redirect permanent / https://admin.ecocarga.cl/
</VirtualHost>

<VirtualHost *:443>
    ServerName ecocarga.cl
    ServerAlias www.ecocarga.cl

    SSLCertificateFile /etc/letsencrypt/live/ecocarga.cl/cert.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/ecocarga.cl/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
    SSLCertificateChainFile /etc/letsencrypt/live/ecocarga.cl/chain.pem

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime

    DocumentRoot /var/www/html/ecocarga.cl
</VirtualHost>

<VirtualHost *:443>
    ServerName admin.ecocarga.cl

    SSLCertificateFile /etc/letsencrypt/live/ecocarga.cl/cert.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/ecocarga.cl/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
    SSLCertificateChainFile /etc/letsencrypt/live/ecocarga.cl/chain.pem

    # remove /admin from url
    RewriteEngine  on
    # redirect to base
    #RewriteCond %{REQUEST_METHOD} ^(GET)$
    RewriteCond %{REQUEST_URI} !(\.css|\.js|media|fonts|img|\/login\/$) [NC]
```

```
    RewriteRule ^/admin/?(.*)$ $1 [R=301,L]

    # if resource are not static or media
    RewriteCond %{REQUEST_URI} !(\.css|\.js|media|fonts|img|\/login\/$) [NC]
    RewriteRule ^(.*)$ /admin/$1 [PT,L]

    Alias /static /home/server/ecocargaWeb/static
    <Directory /home/server/ecocargaWeb/static>
        Require all granted
    </Directory>

    Alias /media /home/server/ecocargaWeb/media
    <Directory /home/server/ecocargaWeb/media>
        Require all granted
    </Directory>

    <Directory /home/server/ecocargaWeb/ecocargaWeb>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>

    WSGIProcessGroup ecocarga
    WSGIScriptAlias / /home/server/ecocargaWeb/ecocargaWeb/wsgi.py

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime
</VirtualHost>

<VirtualHost *:443>
    ServerName api.ecocarga.cl

    SSLCertificateFile /etc/letsencrypt/live/ecocarga.cl/cert.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/ecocarga.cl/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
    SSLCertificateChainFile /etc/letsencrypt/live/ecocarga.cl/chain.pem

    Alias /static /home/server/ecocargaWeb/static
    <Directory /home/server/ecocargaWeb/static>
        Require all granted
    </Directory>

    Alias /media /home/server/ecocargaWeb/media
    <Directory /home/server/ecocargaWeb/media>
        Require all granted
    </Directory>

    <Directory /home/server/ecocargaWeb/ecocargaWeb>
```

```
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>

    WSGIProcessGroup ecocarga
    WSGIScriptAlias / /home/server/ecocargaWeb/ecocargaWeb/wsgi.py

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log reqtime
</VirtualHost>
```

Lo primero que se realiza es la carga de los módulos para conectar APACHE con el proyecto web (mod_wsgi), el uso de HTTPS (mod_ssl) de apache y por último la habilitación de un módulo para redireccionar solicitudes al servidor (mod_rewrite), usado en el virtual host admin.ecocarga.cl. A continuación, se establece la configuración para mod_wsgi, cuantos procesos y threads usará, la versión de Python que cargará en cada proceso y que usuario ejecutará el proceso.

Las definiciones de los primeros tres servidores virtuales cumplen la función de redireccionar el trafico HTTP al puerto 443 para establecer la comunicación por HTTPS. Las siguientes dos definiciones corresponden a los dominios admin.ecocarga.cl y api.ecocarga.cl que se comunican mediante HTTPS y permiten la comunicación con el proyecto web.

El parámetro reqtime indica el formato en que se almacenará cada registro en el archivo access.log de APACHE, a diferencia del que se establece por defecto este agrega el tiempo que tomó cada solicitud en ser respondida por el servidor, su definición debe ser hecha en el archivo “/etc/apache2/apache2.conf” y corresponde a la siguiente:

```
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\" %D" reqtime
```

Para habilitar el servidor es necesario ejecutar el comando de APACHE “sudo a2ensite ecocarga” y reiniciar el servicio mediante el comando “sudo service apache2 restart”, con esto ya estará disponible en la máquina.

La configuración actual no describe la instalación de los certificados para la conexión por el protocolo HTTPS puesto que se puede realizar de varias formas y escapa del alcance de este documento describir todas esas alternativas.

4.8 Configuración actual del servidor

Hoy en día el servidor se encuentra en una máquina virtual en *Amazon Web Service* (AWS). La máquina corresponde a una instancia "t2.micro" que provee los siguientes recursos: 1 core, 500 MB de memoria RAM y un espacio de almacenamiento SSD de 8GB.

El acceso a la máquina se debe hacer mediante una llave privada[9]. La llave privada que acompaña este documento está asociada al usuario ubuntu de la máquina virtual, por lo que se debe acceder a ella de la siguiente forma: ubuntu@ecoarga.cl –i /ruta/a/llave_privada.

Para el proyecto se creó un usuario en la máquina virtual con nombre "server" y contraseña "12server"[10]. Una vez dentro, el código del proyecto se encuentra en la ruta "/home/server/ecocargaWeb". Como el código está escrito en el lenguaje Python, se requiere de un intérprete, para esto se hace uso de un entorno virtual que permite aislar la versión de Python que usa el proyecto, este se encuentra en la misma carpeta del proyecto y lleva por nombre "myenv".

La versión de Python usada para la ejecución del proyecto es la 3.6.6 y las librerías de las cual depende el proyecto pueden ser encontradas en el archivo "requirements.txt".

La base de datos de PostgreSQL usada es la 10.8 y sus parámetros de conexión están definidos en el archivo con nombre ".env", ubicando en el directorio del proyecto.

Las imágenes de empresas eléctricas, marcas de vehículos y modelos de autos pueden ser encontradas en las rutas "/home/server/ecocargaWeb/media/compania", "/home/server/ecocargaWeb/media/marca" y "/home/server/ecocargaWeb/media/modelo" respectivamente, se recomienda subir estas imágenes a través del panel de administración y no directamente en este path puesto que requiere estar asociado a un registo en la base de datos.

La disponibilización del proyecto web a través del protocolo HTTP/HTTPS se hace mediante el software APACHE, y su configuración puede ser encontrada en el archivo "/etc/apache2/sites-enabled/ecocarga.conf", en él se establecen las siguientes acciones:

- El acceso a ecocarga.cl a través del protocolo HTTP es enviado al dominio www.ecocarga.cl usando el protocolo HTTPS
- El acceso a api.ecocarga.cl a través del protocolo HTTP es enviado al dominio api.ecocarga.cl usando el protocolo HTTPS
- El acceso a admin.ecocarga.cl a través del protocolo HTTP es enviado al dominio admin.ecocarga.cl usando el protocolo HTTPS
- El acceso a ecocarga.cl o www.ecocarga.cl a través del protocolo HTTPS es enviado a un archivo estático ubicado en "/var/www/html/ecocarga.cl"
- El acceso a api.ecocarga.cl a través del protocolo HTTPS es configurado para usar el proyecto web

[9] La llave privada junto con los datos de acceso a la cuenta de AWS serán entregadas junto con este informe.
[10] Se recomienda cambiar la contraseña por una con mayor complejidad

- El acceso a admin.ecocarga.cl a través del protocolo HTTPS es configurado para usar la sección de administración del proyecto web

El certificado para el uso del protocolo HTTPS se crea por medio de la herramienta certbot[11] que usa el protocolo ACME[12]. El certificado tiene una duración de 90 días, luego se debe actualizar en el servidor con el comando “sudo cerbot renew”. Actualmente existe un cron (ver con el comando “sudo crontab -e”) que intenta renovar el certificado el día 1 de cada mes.

El registro de errores en una aplicación web permite verificar si alguna acción de usuario no está funcionando como debiese, en la aplicación web existe dos lugares donde se puede verificar algún error en alguna llama. El primero corresponde al archivo de log generado por el framework Django, este se encuentra en la ruta “/home/server/ecocargaWeb/logs/file.log”. El segundo corresponde a un log a nivel de servidor, es decir, el generado por el software APACHE, este archivo puede ser encontrado en “/var/log/apache2/error.log”.

Para administrar el dominio ecocarga.cl se usa el servicio *Route53* de AWS, en la Tabla 3 se muestra la configuración para dominio.

Nombre	Tipo	Valor	TTL
ecocarga.cl.	A	54.207.23.37	300
ecocarga.cl.	NS	ns-1513.awsdns-61.org. ns-1796.awsdns-32.co.uk. ns-707.awsdns-24.net. ns-129.awsdns-16.com.	172800
ecocarga.cl.	SOA	ns-707.awsdns-24.net. awsdns-hostmaster.amazon.	900
admin.ecocarga.cl.	A	54.207.23.37	300
api.ecocarga.cl.	A	54.207.23.37	300
www.ecocarga.cl.	A	54.207.23.37	300

Tabla 3: configuración de dominio ecocarga.cl

[11] https://certbot.eff.org/
[12] https://ietf-wg-acme.github.io/acme/draft-ietf-acme-acme.html

5. Lineamientos de diseño

Esta sección abarcará todos los elementos que influyen en la creación de marca para el proyecto

5.1 Marca

Ecocarga, electrolineras en línea, nace como marca en base a un concepto principal: Consciencia verde, este concepto representa ideas con las que los usuarios de la aplicación se sienten identificados, ideas como eficiencia, consciencia global, vehículo ecológico, tecnología y vanguardia.

Ecocarga es:

- Una aplicación que ayuda a quienes tienen consciencia ecológica
- Una herramienta moderna que hace más fácil movilizarse en autos eléctricos
- Una marca simple, directa, mezcla del medio ambiente y la tecnología

5.2 Norma

Entendemos como norma gráfica a una serie de reglas y condiciones que, aplicados a distintos elementos usados de manera conjunta, crean un sistema visual consistente que da forma a la marca.

Esta sección ayuda a mantener la consistencia de los elementos gráficos independiente del número de personas que trabajen en ellos, para ayudar al reconocimiento de marca por parte de nuestros usuarios.

1.2.1 Logo

El logo (o imagotipo) es la imagen más visible y representativa de la marca, es la imagen con la que los usuarios se encuentran inmediatamente tanto al abrir la aplicación desde su celular como al buscarla en la Appstore o Playstore, y se compone de los siguientes elementos:

Imagotipo:
Es la combinación del Isotipo y el logotipo, es la representación principal para usar de la marca y siempre debemos tratar de usarla de esta manera.

Logotipo:
Son las palabras que pueden leerse del logotipo, su composición no debe ser alterada ya que es parte de la consistencia del logo.

ECOCARGA
ELECTROLINERAS EN LÍNEA

Isotipo:
Es la imagen que acompaña al logotipo, viene de la fusión simplificada entre una electrolinera y una hoja, es la parte más importante del logo ya que se usa además como icono en distintas funcionalidades de la aplicación.

1.2.2 Articulaciones

Esta sección define las diferentes composiciones que puede tener el logo para ser usadas en distintos casos, junto a recomendaciones de uso

Principal:
El imagotipo en articulación vertical es la que debe usarse como preferencia. Siempre que se pueda usar esta articulación, deberá usarse.

Horizontal:
Articulación para espacios reducidos en altura y formatos tipo Banner en la página de descripción de la aplicación en la Appstore y Playstore.

Isotipo aislado:

El isotipo aislado es necesario para contextos de espacio reducido y tamaños pequeños, la legibilidad del isotipo está hecha para ser parte de las interfaces de Android e iOS.

1.2.3 Variaciones de color

Principal:

El logo debe usarse con sus colores principales solo sobre blanco y leves tonos de grises para mantener la legibilidad de la marca.

Blanco:

Esta es la variación principal, ya que puede usarse sobre el gradiente característico de la marca para dar variedad a las piezas gráficas y mejorar el impacto visual. También es la opción que debe usarse para funcionalidades técnicas como iconos de la aplicación en notificaciones, iconos de estado y como icono principal. Por último, es la opción para ser usada sobre fondos oscuros.

Negro:

El logo puede usarse en negro única y exclusivamente cuando sea técnicamente imposible aplicarlo de otra forma, ya que es un color que se aleja de los conceptos de la marca.

1.2.4 Proporciones y tamaños

Las proporciones del imagotipo respecto al logotipo y sus márgenes están definidos por la distancia entre el icono y su texto, esta medida es la que se usará para definir los márgenes exteriores en todas las articulaciones del logo

Articulación principal

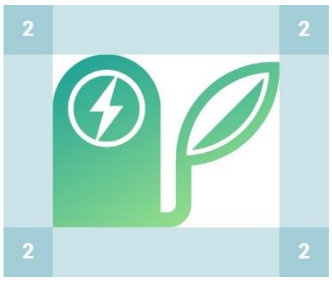
En esta composición la diferencia entre el ancho del texto y el ancho del icono es proporcional a 6 veces la distancia entre el icono y su texto inferior, además los márgenes, o área de seguridad, es de 2 veces esta distancia

Articulación horizontal

En articulación horizontal los márgenes se mantienen igual que en la composición anterior, y el texto se centra verticalmente respecto a su ícono, con una separación de 1 arriba y abajo, y 2 a la izquierda y derecha

Articulaciones de imagotipo y logotipo

En ambos casos los márgenes o área segura debe ser de 2 veces la distancia base, y para el logotipo su área segura arriba y abajo debe mantener la proporción con y sin el imagotipo acompañándolo.



1.2.5 Usos correctos e incorrectos

El impacto del logo de una marca depende de su uso apropiado. Cualquier cambio o inconsistencia en su uso, ya sea de color, estructura o forma, disminuirá la pregnancia de los conceptos asociados a la electromovilidad y ecología con los que se representa la marca.

Estructura
No alterar la forma en la que se relaciona el imagotipo junto a su logotipo

Colores y elementos adicionales
No cambiar los colores fuera de la paleta de colores aquí definida ni agregar elementos adicionales.

Fondos
Respetar las normas de aplicación del logo.

1.2.6 Formatos y aplicaciones

5.3 Elementos adicionales

.