**PROYECTO FULL STACK**



Gonzalo González Ventura

David Orive Ramírez

INDICE

* Introducción 3
* Diagramas 4-8
  + Diagramas UML 4
  + Diagramas E/R 5
  + Diagramas entidad relación 6
  + Diagramas relacionales 7
  + Descripción diagramas 7-8
* Requisitos de usuario 8
* Casos de uso 9
* Funcionamiento del sistema 9
* Interfaces 10
  + Mockups 10
  + Usabilidad 11
* Manuales 11
* Pila tecnológica 12
* Comparación de tecnologías 12
* Repositorios 13
* Planificación 13
* Conclusión 14

1. Introducción

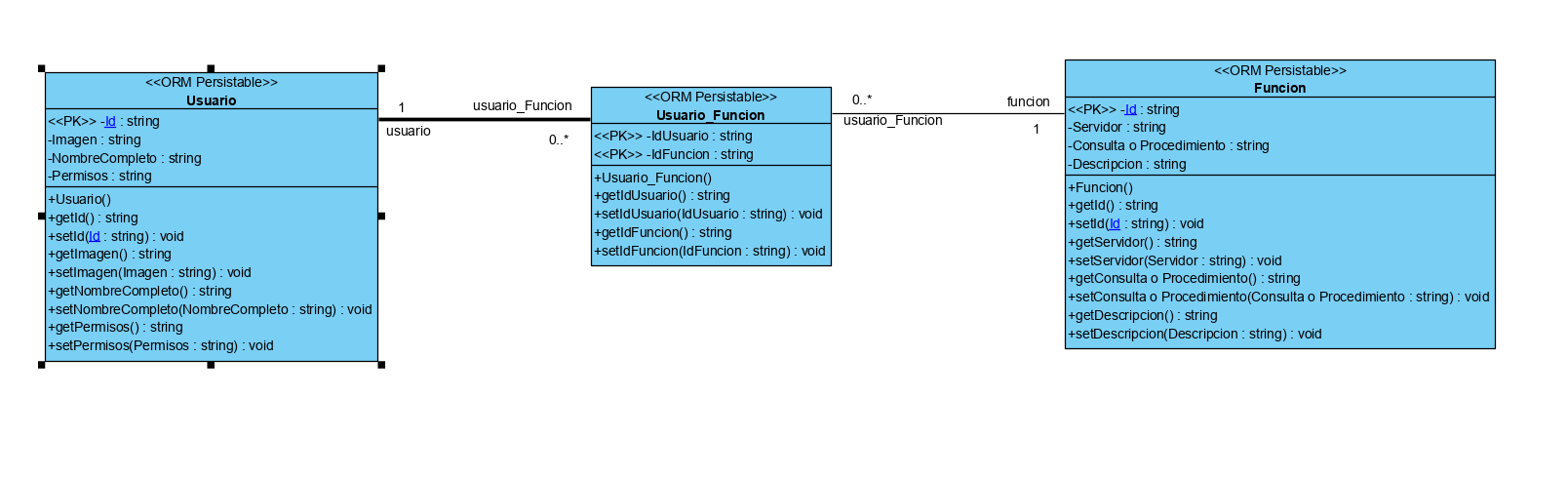
Vamos a desarrollar una aplicación para la empresa Valper Soluciones Y Mantenimientos, S.L. Esta empresa se dedica al asesoramiento, suministro, mantenimiento y soporte de sistemas informáticos. Desarrollo de Software a medida. Comercialización de Software para el sector de la automoción. Ya existe acuerdo de distribución y mantenimiento con empresa multinacional. Desarrollo de aplicaciones de integración para el software anteriormente mencionado para ser distribuidos a nivel nacional.

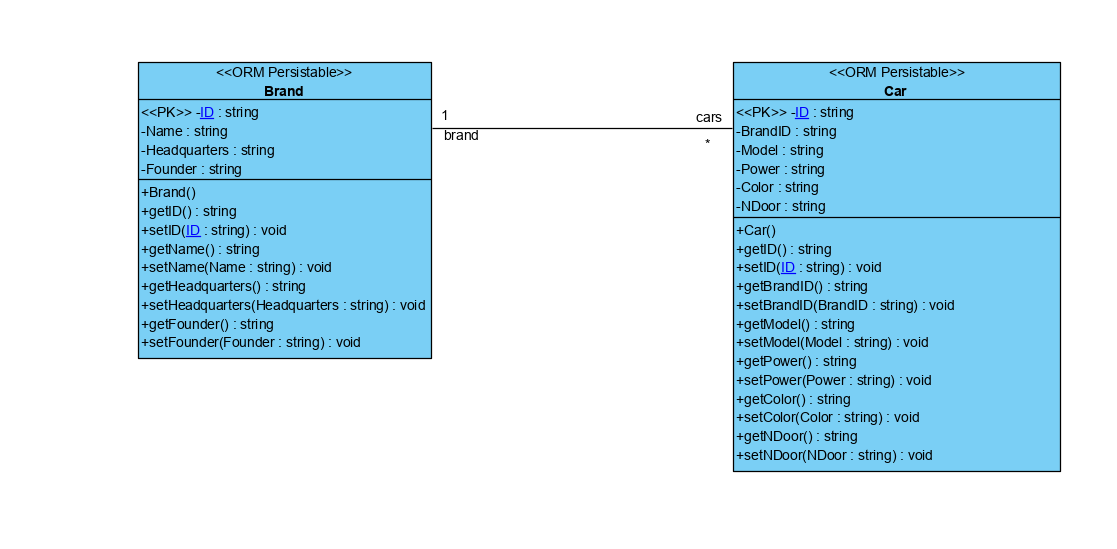


El proyecto va a consistir en una aplicación la cual va a guardar marcas de coche y sus modelos. Además, tendremos la función de crear usuarios los cuales tendrán distintas funciones.

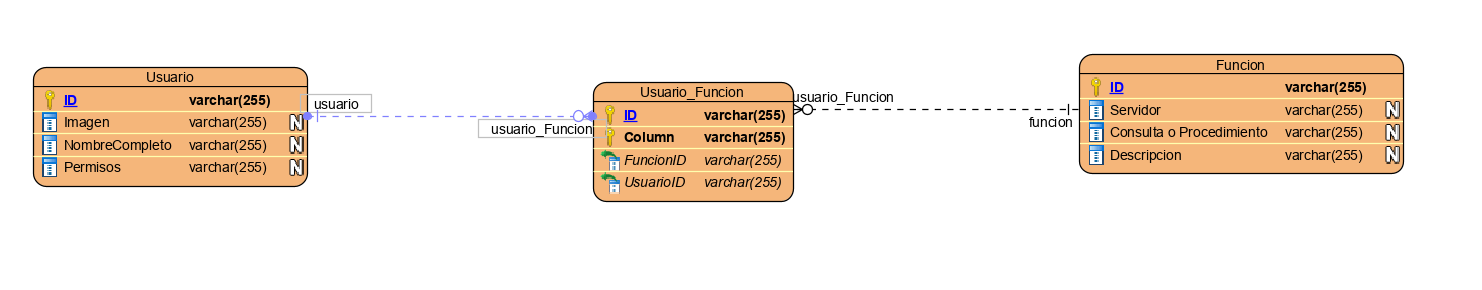
1. Diagramas

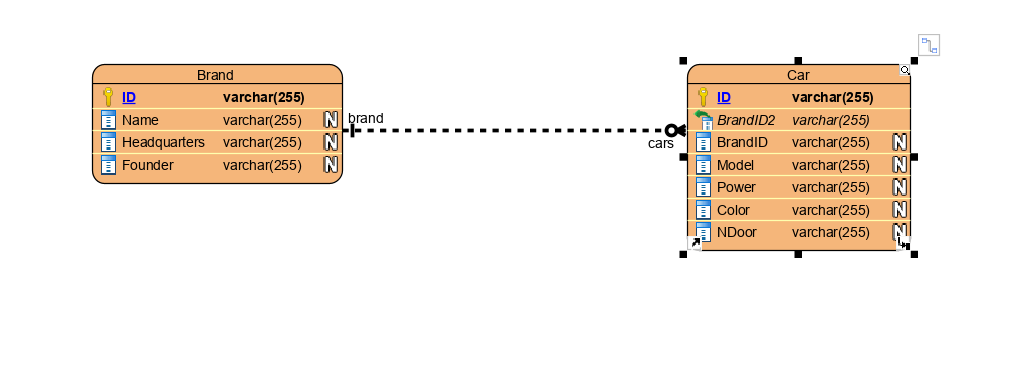
* Diagramas UML



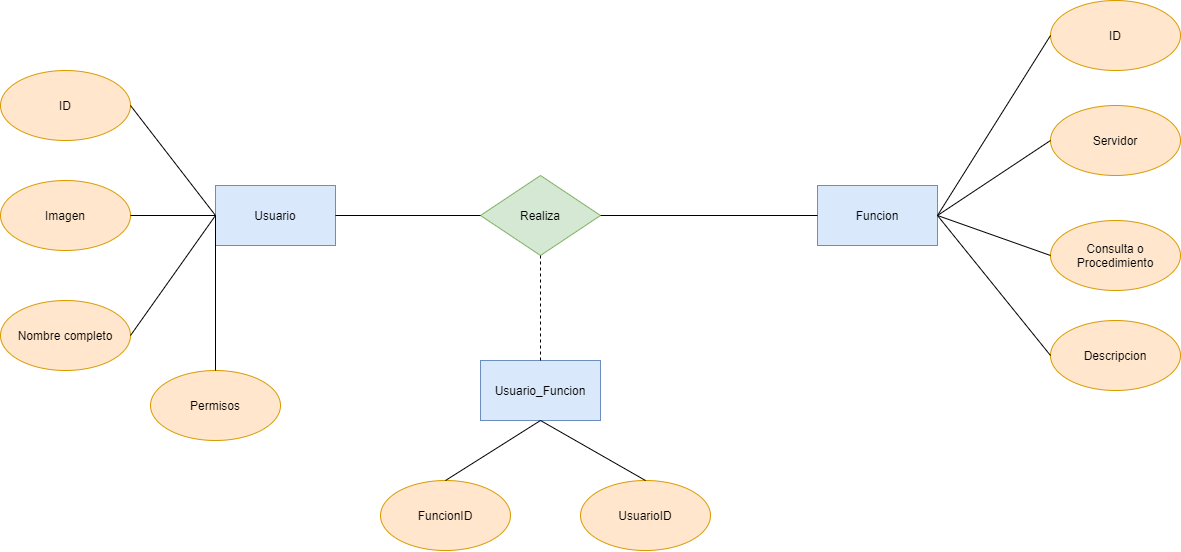


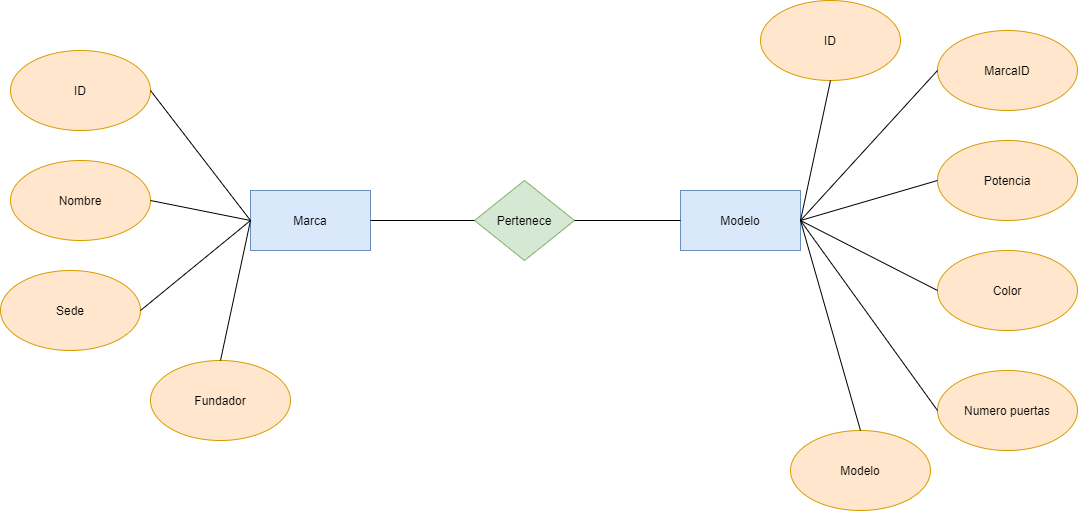
* Diagramas E/R





* Diagramas entidad-relación





* Diagramas relacionales

Usuario(Id,NombreCompleto,Permisos,Imagen)

Usuario\_Funcion(IdUsuario\*,IdFuncion\*)

Función(Id,Servidor,Consulta/Procedimiento,Descripcion)

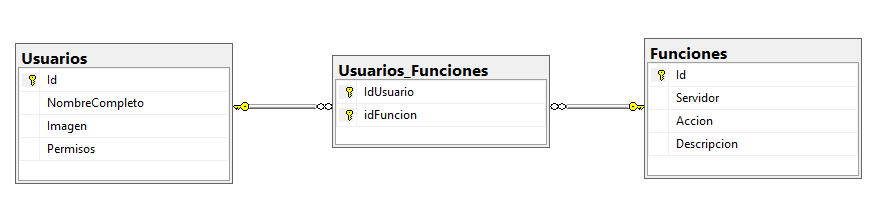
Marca(Id,Nombre,Sede,Fundador)

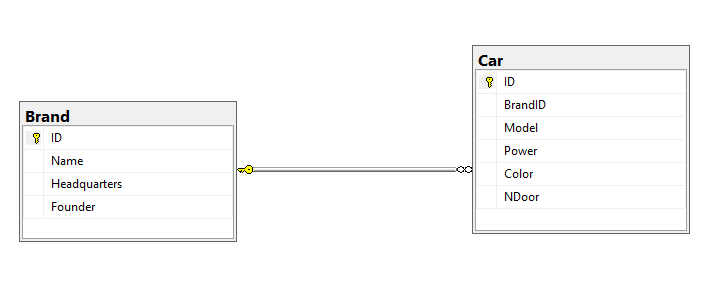
Modelo(Id,MarcaID\*,Potencia,Color,NumeroPuertas,Modelo)

* Descripcion diagramas

Esta aplicación consta de dos bases de datos la de los usuarios y la de las marcas.

* + **Usuarios:** Como vemos en la imagen de abajo se trata de una relación de muchos a muchos entre usuarios y funciones y de esta relación se genera una tabla que es Usuarios\_Funciones, el cual guarda la id de usuarios y funciones. La tabla usuarios guarda los campos id, el nombre completo, la imagen y los permisos que tienen dichos usuarios. La tabla funciones guarda la id, el servidor la acción que estas realizando que puede ser una función o una consulta y una descripción de esta.



* + **Marcas:** Esta tabla consta de una relación de uno a muchos entre las tablas marca y modelo. La tabla marca guarda los campos de la id, el nombre de la marca, la sede y el fundador. La tabla modelo guarda la id del modelo, la id de la marca a la que pertenece, el nombre del modelo, la potencia, el color y el número de puertas.

1. Requisitos de usuario

R1. Aplicación multi idioma

R1.1. Posibilidad de elegir entre español e ingles

R2. Usuario

R2.1. Registrar usuario

R2.2. Login de usuario registrado

R2.3. Usuario administrador

R2.4. Posibilidad de editar usuarios

R3. Marcas de coches

R3.1. Posibilidad de añadir, modificar y eliminar una marca de coche

R3.2. Posibilidad de visualizar las marcas de coche

R4. Modelos de coches

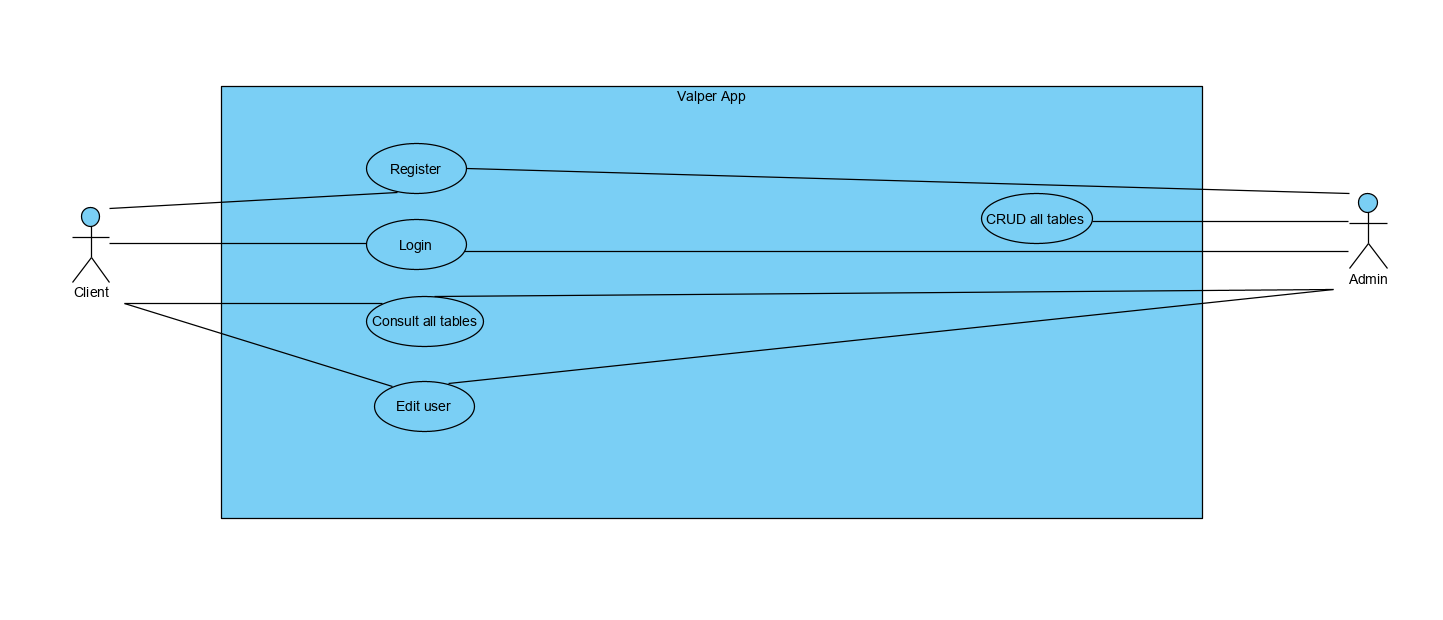
R4.1. Posibilidad de añadir, modificar y eliminar una modelo de coche

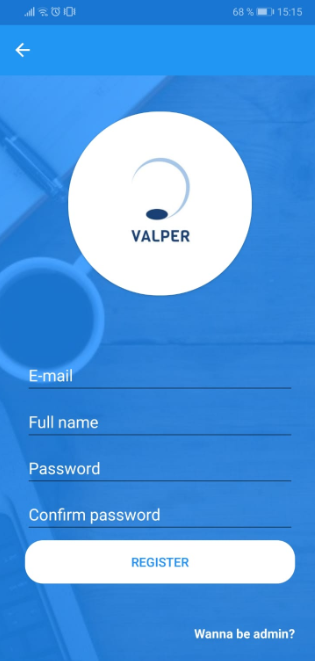
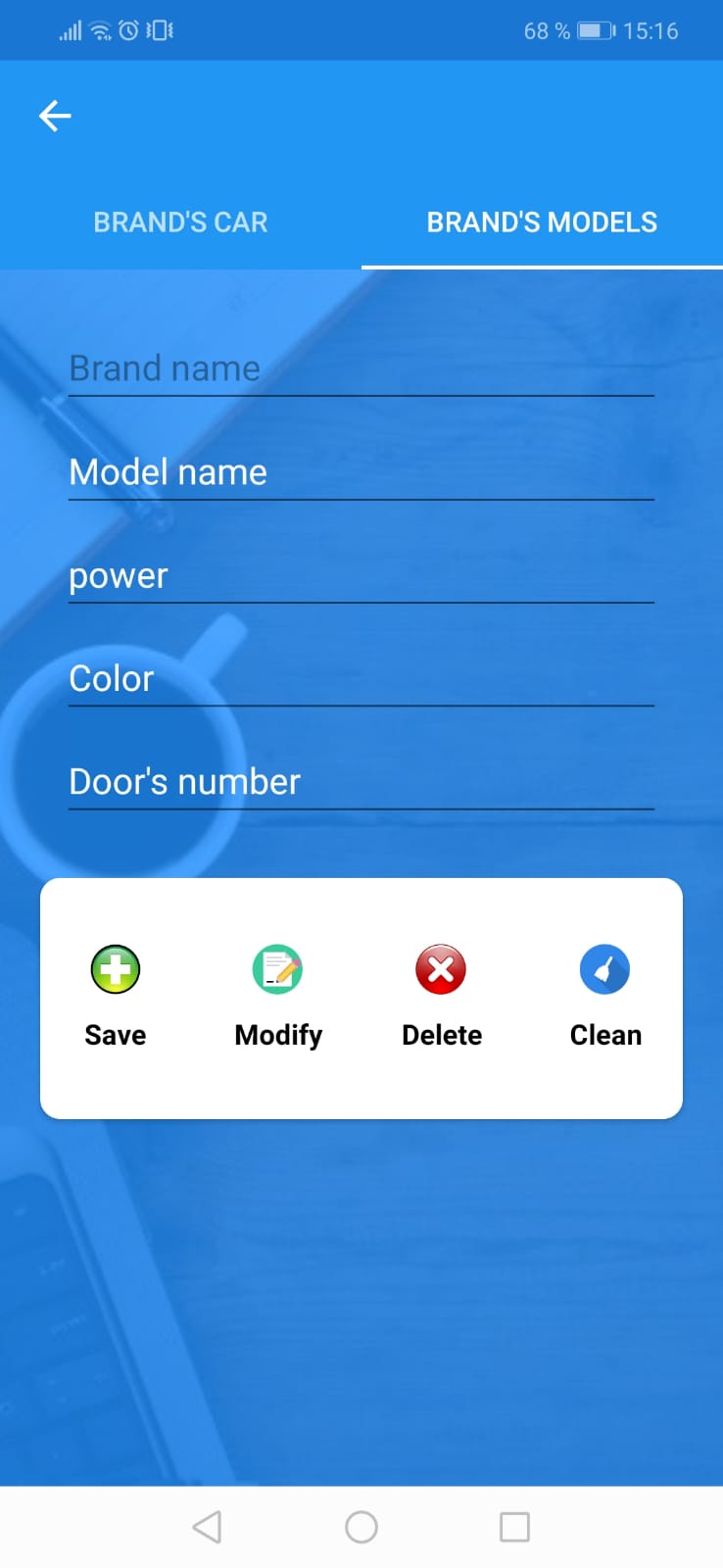
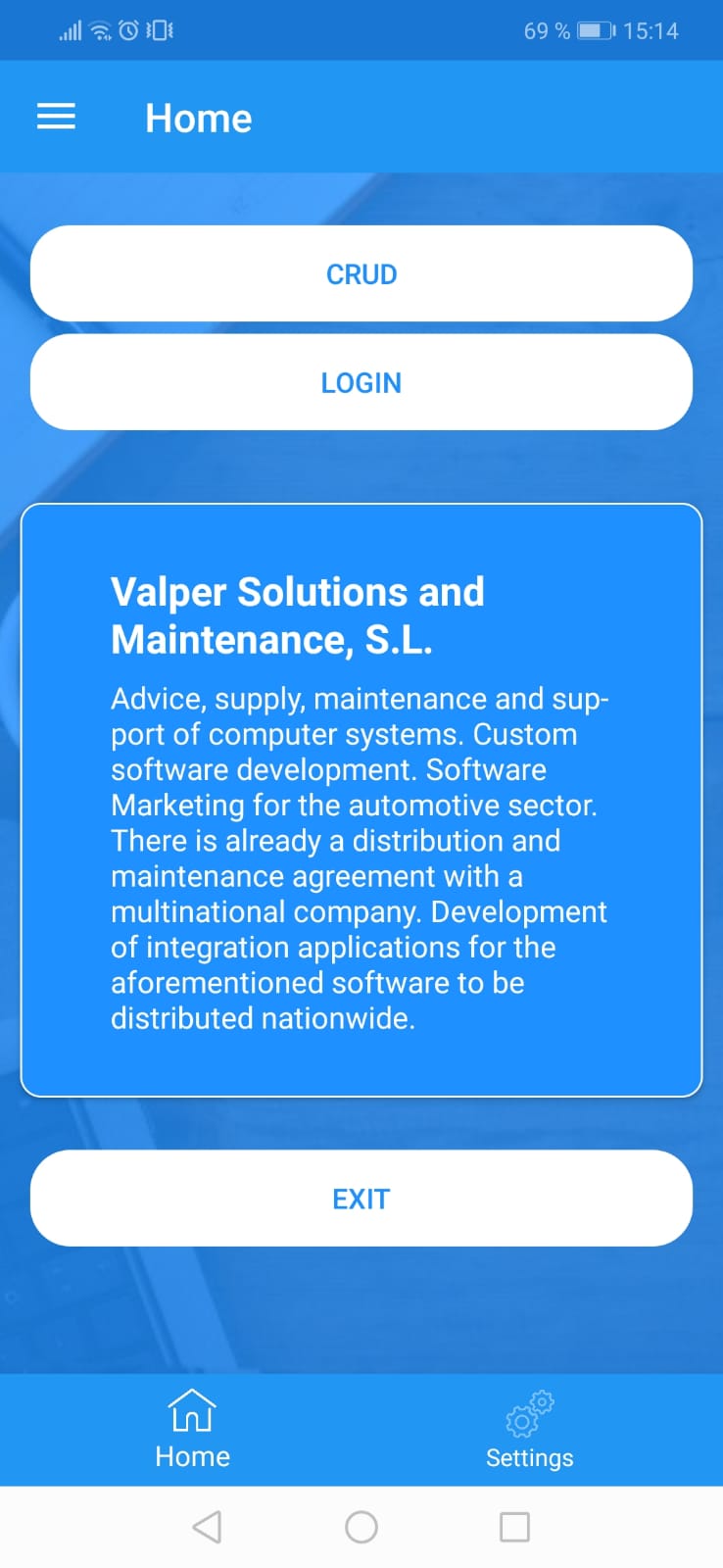
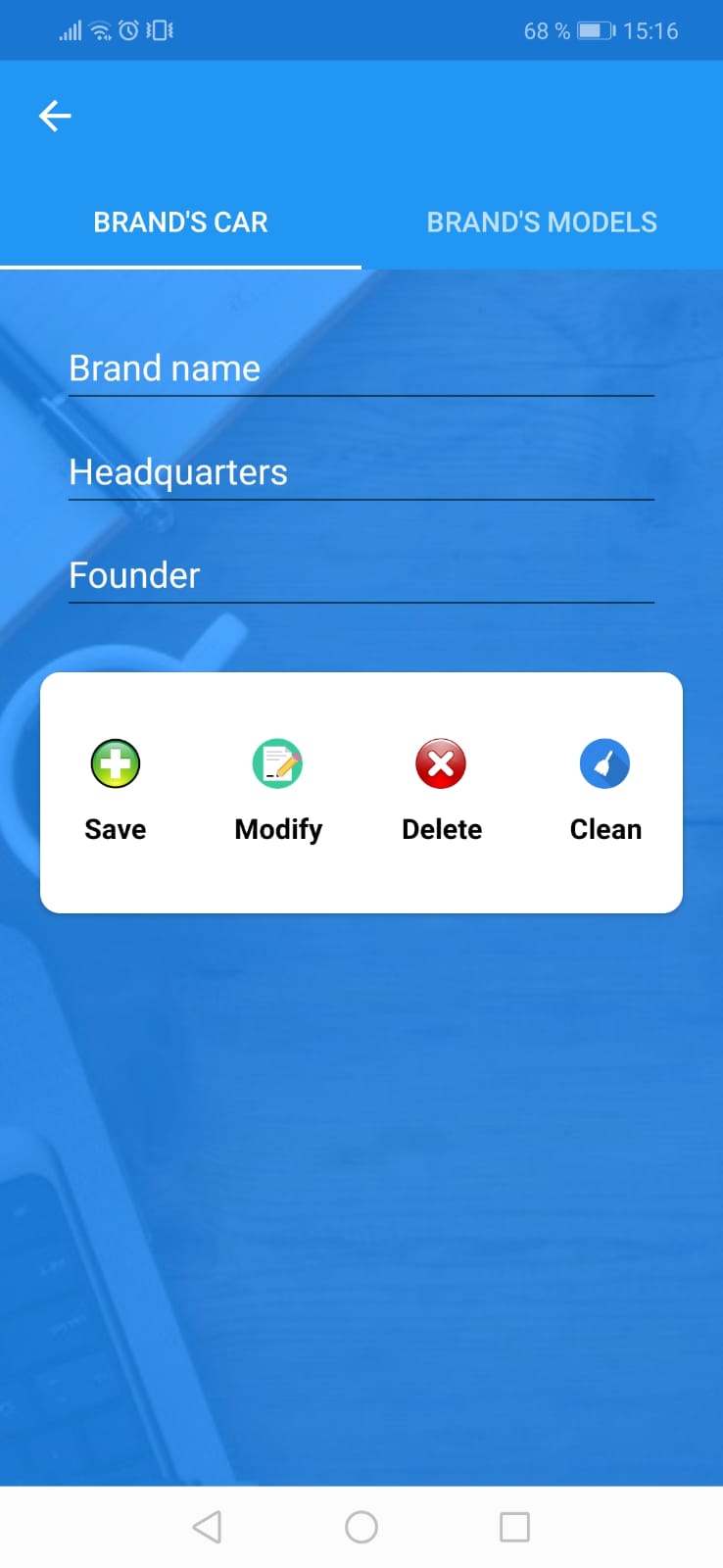
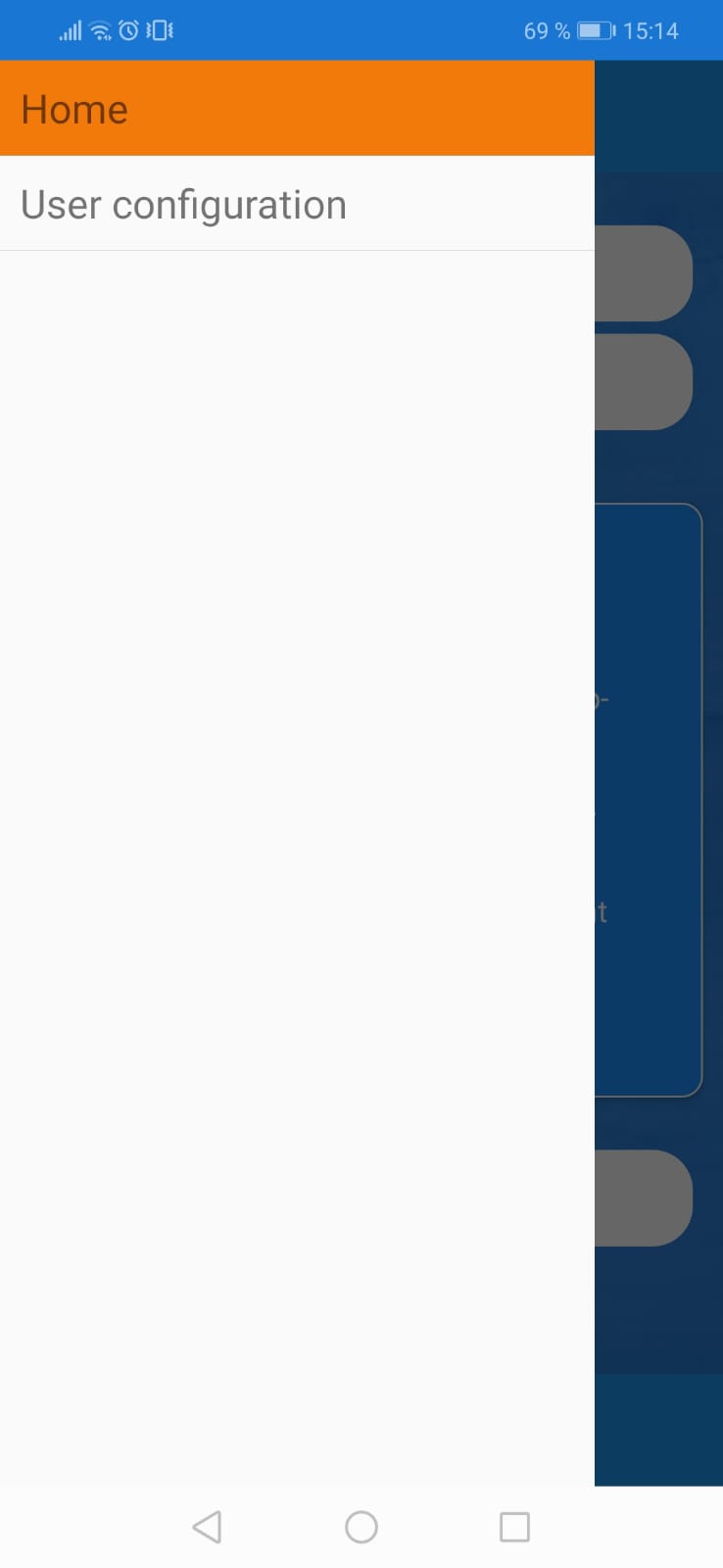
R4.2. Posibilidad de visualizar las modelos de coche

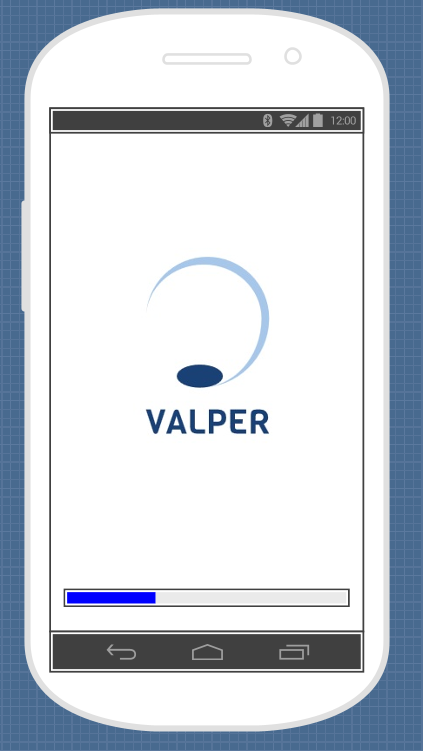
1. Casos de uso

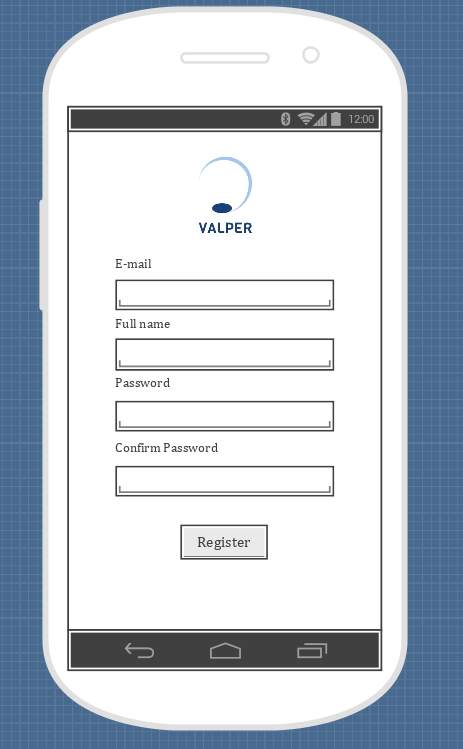
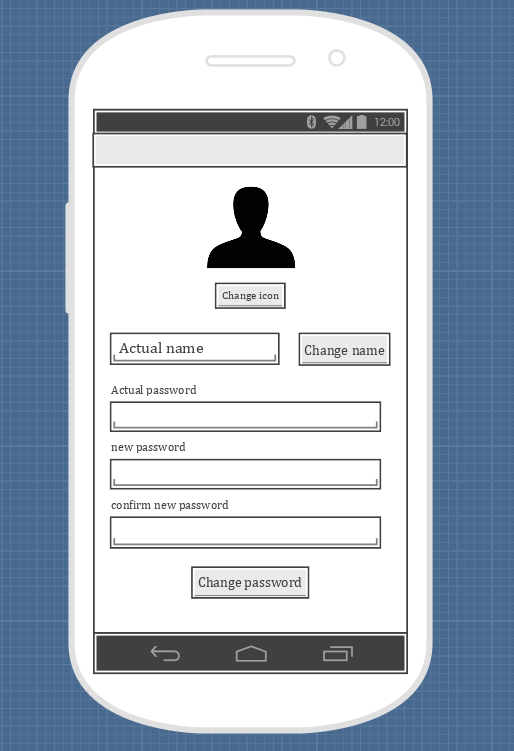
En nuestra aplicación hay dos tipos de usuario el usuario normal y el administrador.

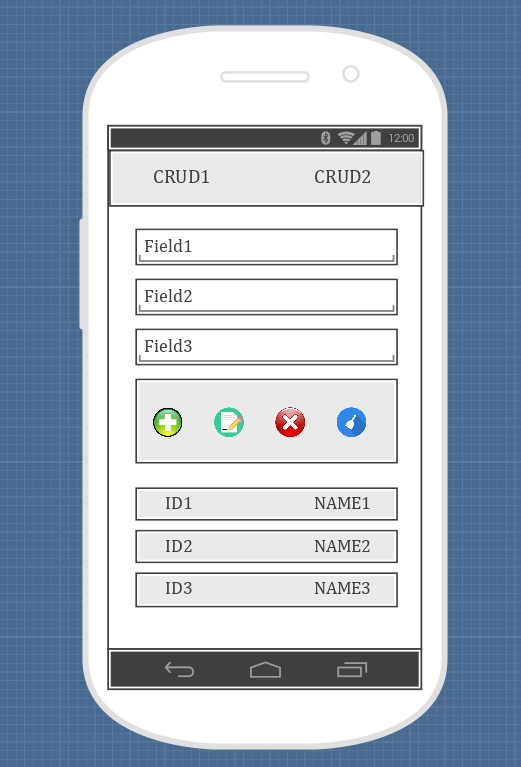
* El usuario normal (client en el diagrama) tiene permiso para registrase loguearse, editar su usuario y por último visualizar todas las tablas.
* El usuario administrador (admin en el diagrama) Tiene todos los permisos del usuario normal y además tiene el permiso de introducir, modificar y eliminar datos de todas las tablas.



1. Funcionamiento del sistema<blob:https://web.whatsapp.com/2dc408ea-f39f-4ad5-ae3f-04974beca62c>
2. Interfaces

* Mockups





* Usabilidad

1. Manuales

* Manual de instalación:

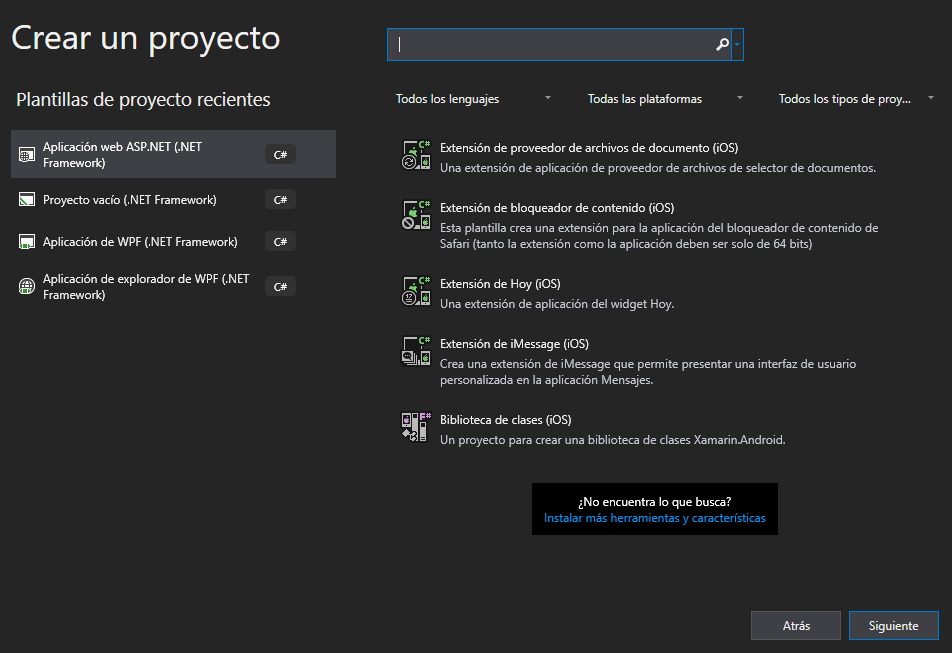
Estas instrucciones le proporcionarán una copia del proyecto en funcionamiento en su máquina local para fines de desarrollo y prueba. Consulte la implementación para obtener notas sobre cómo implementar el proyecto en un sistema en vivo.

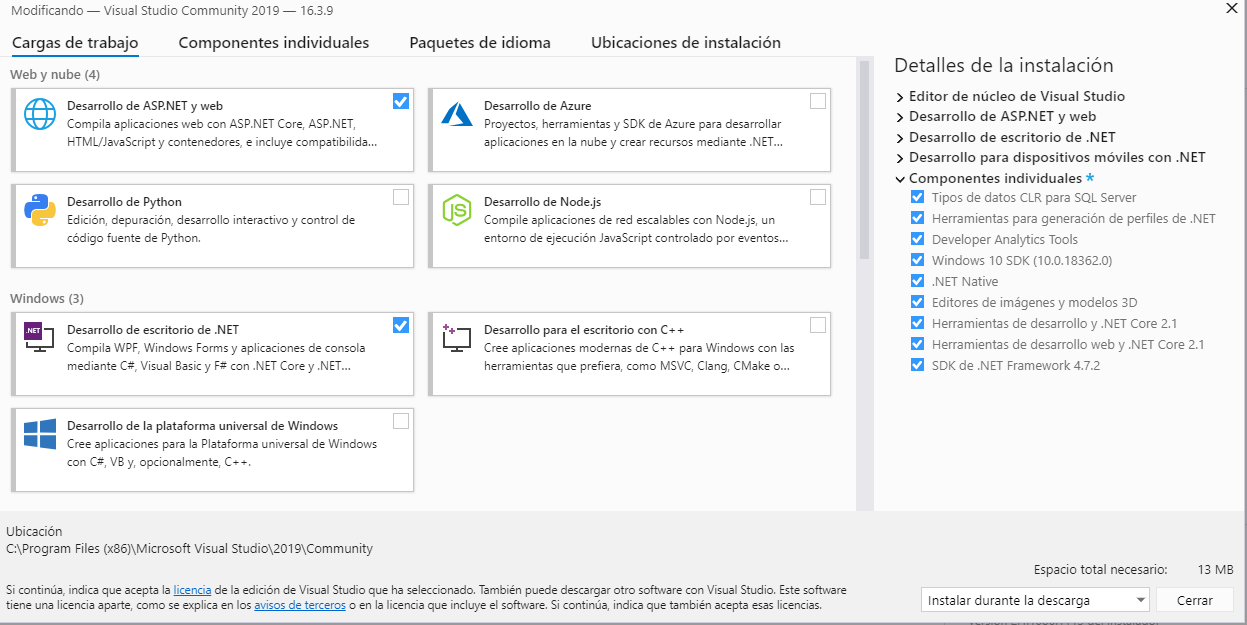
**Prerrequisitos.**  
Qué elementos necesita para instalar el software y cómo instalarlas.

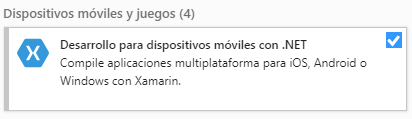
* + IDE: Visual Studio 2019
  + Visual Studio Requeriment (install): Xamarin, ASP.net
  + System database: SQL server
  + Database Management: SQL server Management Studio
  + API REST test: Postman
  + Emulator: Android Studio
  + ORM: Entity Framework

**Instalar la aplicación.**

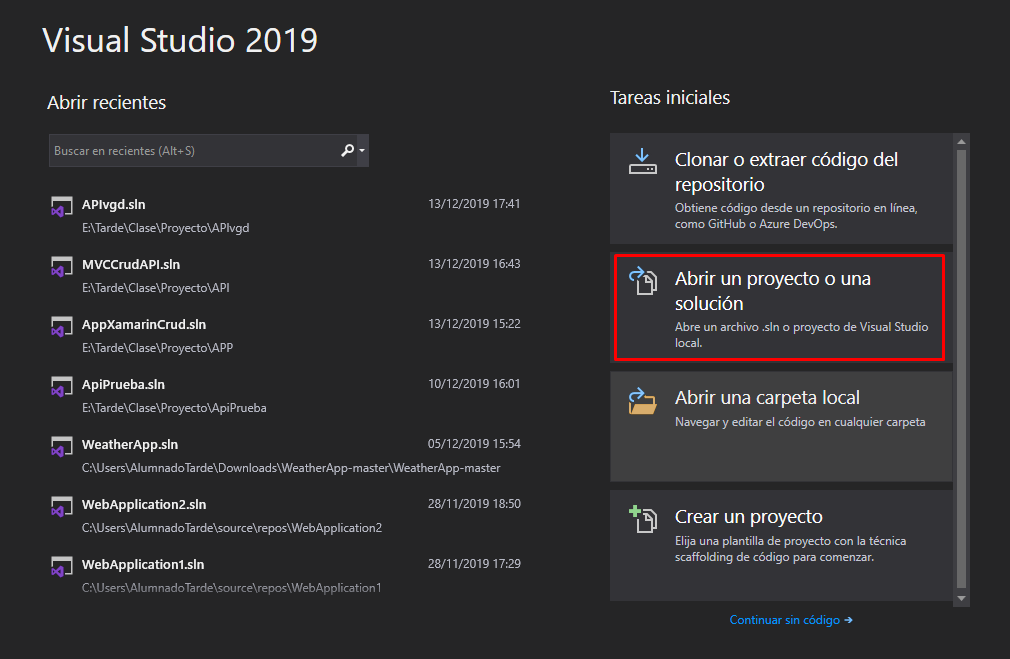
Primero necesitará instalar [Visual Studio 2019](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&rel=16)

A continuación, debe instalar las dependencias:

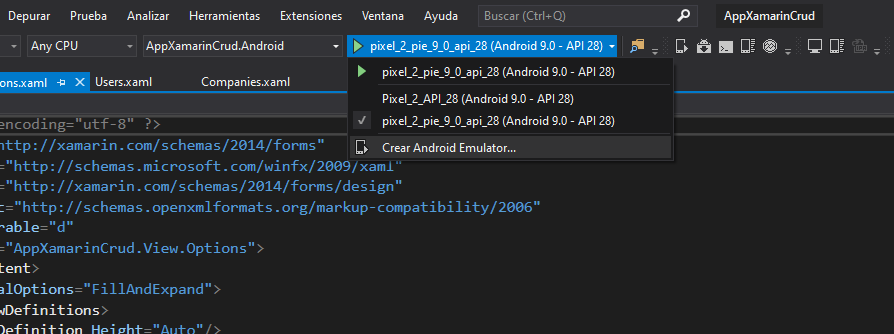




Ahora puede abrir la aplicación Xamarin, seleccionando abrir una solución existente.



Una vez tienes abierto el archivo, deberás crear un emulador de Android si aúnno tienes ninguno.



Y entonces podrás ejecutar la aplicación.

### Instalando la API

### Primero de todo debes instalar SQL server.

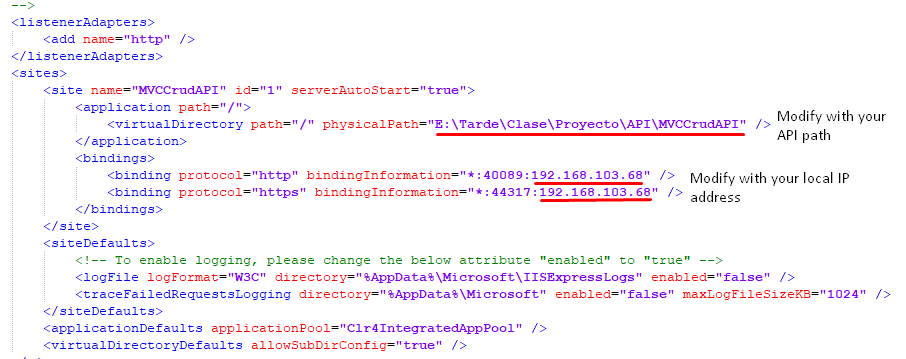
### Después, deberás instalar el SQL server Management Studio.

### Una vez tienes ambos instalados, necesitarás conectarte a la base de datos, para ello añade un nuevo “query”. Arrastra los archivos sql adjuntos en la carpeta “Documentación” y podrás ejecutarlos.

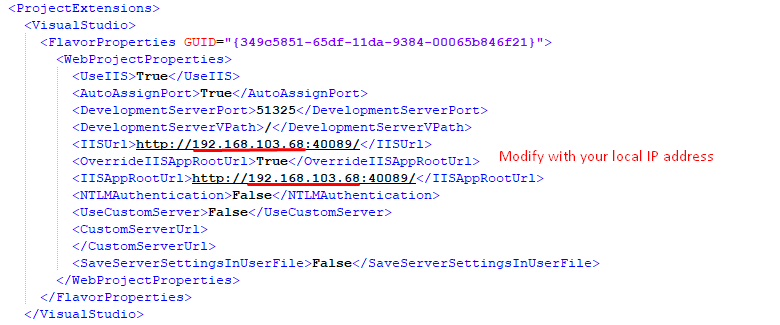
### alt

Ahora hay que modificar 2 archivos siguientes:

API.vs\MVCCrudAPI\config\applicationhost.config



API\MVCCrudAPI\MVCCrudAPI.csproj



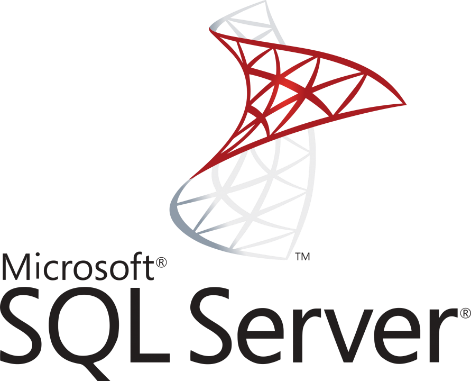
Ahora podrás ejecutar el servidor.

### Conectar la APP a la API

Solo debes acceder al menú de opciones de la aplicación y entonces ir a opciones y seleccionar una IP.

Pila tecnológica

* **IDE:** Visual studio
* **Base de datos:** SQL server
* **Gestor de base de datos:** SQL server management studio
* **ORM:** Entity Framework
* **Diagramas:** Visual Paradigm Enterprise
* **Gestor de peticiones:** Postman
* **Tecnologia:** Xamarin
* **Api:** ASP.net
* **Emulador:** Android Studio



1. Comparación de tecnologías

## Nativa

Es la tecnología que utiliza Xamarin, sus ventajas y desventajas son:

**Ventajas:**

* Acceso complete al dispositivo, en software y hardware.
* Mejor experiencia de usuario
* Visualización desde tiendas de aplicaciones e integración con otros dispositivos inteligentes

**Desventajas:**

* Diferentes lenguajes de programación y habilidades según el Sistema operative
* Costos y tiempo de Desarrollo altos
* No aparecería en búsquedas de google en caso de no tener una web optimizada para móviles

## Web

Sus lenguajes conocidos son JavaScript y HTML, sus ventajas y desventajas son:

**Ventajas:**

* Código de programación reutilizable
* Desarrollo más sencillo y de menor costo
* No necesita instalación

**Desventajas:**

* La conexión a internet es necesaria
* Acceso limitado al hardware del dispositivo
* Pierde visibilidad en las tiendas por no necesitar instalación

## Híbrida

Sus lenguajes más conocidos son HTML5, CSS3 y JAVA, sus ventajas y desventajas son:

**Ventajas:**

* Gran parte del desarollo es compartido entre todas las plataformas (Android, iOS, Windows Phone)
* Menor coste de Desarrollo y diseño
* Mantenimiento y actualizaciones más fáciles de desarrollar
* Basadas en estándares web populares
* Prototipado más rápido

**Desventajas:**

* En ocasiones no se puede acceder a las funcionalidades del hardware del dispositivo
* El diseño de la aplicación será simulado para parecer una aplicación nativa
* Dependiendo de la complejidad de la app, la velocidad y fluidez puede verse perjudicada.

## Progressive Web App

Sus ventajas y desventajas son:

**Ventajas:**

* Tiene interfaz de aplicación nativa
* Funciona independientemente de la conexión a internet
* Envía notificaciones push
* Se actualiza constántemente
* Es instalable

**Desventajas:**

* Rendimiento limitado y consume más batería de lo normal, porque el Código native es más rápido que el Código web
* El Código es interpretado en lugar de compilado
* No puede acceder a todas las funcionalidades específicas del dispositivo

1. Repositorios

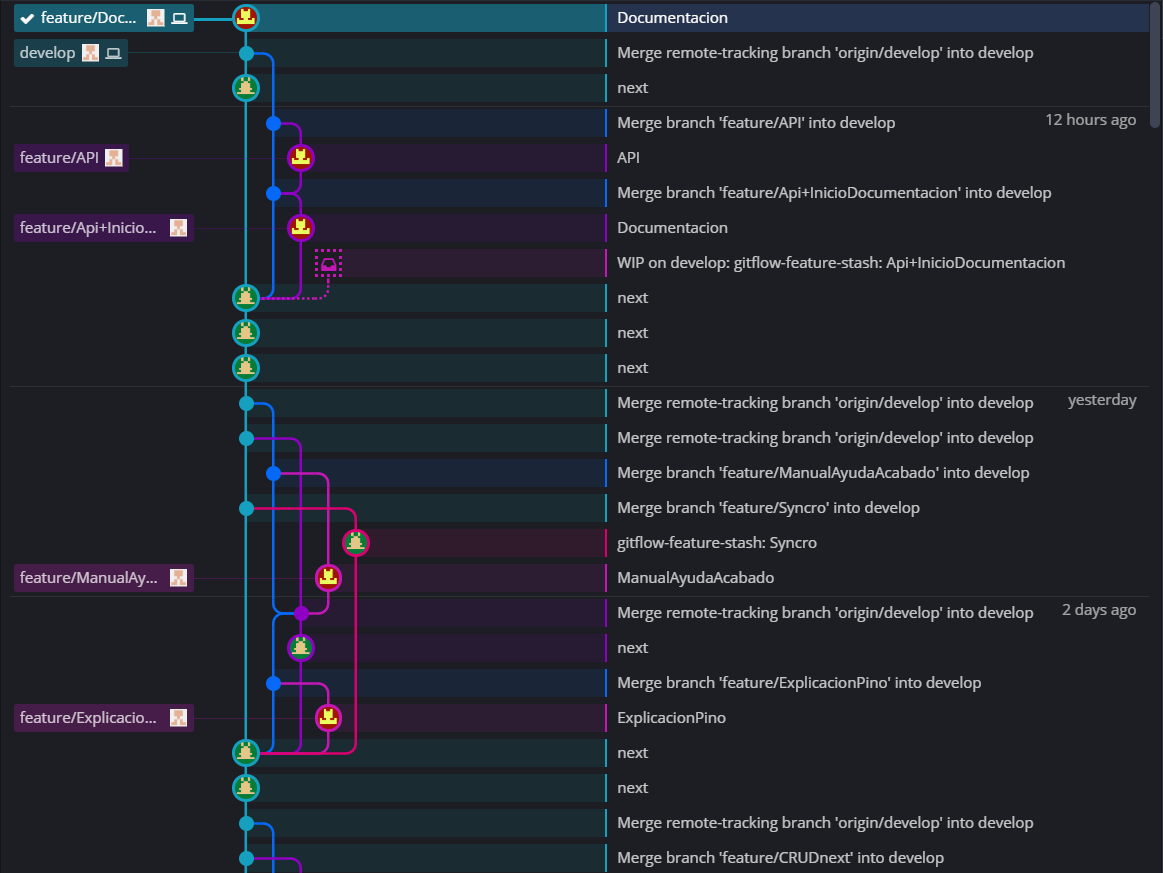
* Repositorio de la aplicación

<https://github.com/GonzaCarry/Full-Stack-Valper>

* GitHub de cada uno

<https://github.com/GonzaCarry>

<https://github.com/doriveramirez>



1. Enlaces y referencias

* <https://docs.microsoft.com/es-es/xamarin/>
* <https://developer.android.com/guide?hl=es-419>
* <https://www.youtube.com/watch?v=0q5QuvyYq-M&list=PLUcTZUs9_XmGtC7uz2mQYo-Dn317wq_P8&index=6&t=0s>
* <https://www.youtube.com/watch?v=jTveozl4cQY>
* <https://www.sap.com/spain/products/crystal-reports.html>
* <https://es.stackoverflow.com/>

1. Postman

<https://asdgdsgsagd.postman.co/collections/8886235-b1da850a-35bf-491d-9cad-cb6665e0b613?version=latest&workspace=54ad114d-cf16-4e12-8269-8b1b9d9f85f0>

# Características de Usabilidad

## Características más importantes

* **Es útil:** Cumple las tareas específicas para la cual ha sido diseñado
* **Es fácil de usar:** Es eficiente, veloz y además permite seleccionar la dirección API
* **Fácil de aprender:** La interfaz es intuitiva.
* **Elegante en su diseño:** Es bonito para el usuario.
* **Es eficiente.**
* **Previsión de errores:** Contiene varias excepciones.
* **Retroalimentación de la interfaz:** Es intuitivo.
* **Simplicidad de diseño de interfaz:** Contiene solo lo necesario.

## Características de interfaz usable

* **El usuario es capaz de iniciar y controlar tareas.**
* **El usuario es capaz de interactuar con la aplicación.**
* **Estética de diseño.**
* **Consistencia de la interfaz.**
* **Simplicidad del diseño.**
* **Retroalimentación.**

## Pautas de diseño

* **Organiza, económica y comunica todos los elementos que presentan las distintas interfaces.**
* **Buen diseño visual.**
* **Buena selección de colores elegidos.**
* **Buena diferencia de contraste entre colores.**
* **Ventanas tipo formulario.**
* **Redacción de texto en la interfaz.**
* **Sigue los principios básicos para el diseño de estructura de interfaces.**

## Principios básicos del diseño de la estructura de interfaces

* **Aporta familiaridad con el usuario.**
* **La interfaz es consistente.**
* **La interfaz es legible.**
* **No provoca sorpresa.**
* **Posee mecanismos para recuperarse de los errores.**

## Usuario final

* **El usuario es el objetivo principal.**
* **Los valores de información son cambiados rápidamente.**
* **Los cambios de un valor son indicados de forma inmediata.**
* **El usuario puede realizar una acción en respuesta a los cambios de la información.**
* **El usuario puede interactuar con la información desplegada mediante una interfaz de manipulación directa.**

## Estructura de la interfaz

* **La estructura de la interfaz lleva:** Menús, Ventanas, Cuadros de diálogos y atajos de teclado.

## Color

* **Comunica ideas de forma rápida gracias a la estética que aporta el color.**
* **Los mensajes se presentan de forma llamativa.**
* **El usuario puede recordar fácilmente los colores.**
* **No se abusa del color.**
* **No se produce confusión debido a los colores.**

## Fuente

* **La fuente utilizada es legible.**
* **El tamaño de la fuente es adecuado.**
* **Lleva los bloques justos y necesarios de texto.**

## Iconos

* **Las pantallas llevan iconos adecuados.**
* **No contiene demasiados elementos gráficos.**
* **Posee equilibrio entre el contraste visual y clara organización.**
* **Posee una diferencia clara entre cada elemento que conforma la aplicación.**

## Elementos interactivos

* **Los elementos que poseen realimentación son fundamentales**
* **El sistema no tarda más tiempo de lo normal en procesar un comando.**
* **Se informa de distintos errores a la hora de llevar a cabo acciones.**

## Reglas de efectiva presentación de los datos

* **El Espacio está separado en áreas según su función.**
* **El espacio está balanceado en ejes horizontales y verticales.**
* **La simetría está presente en los ejes.**

## Diseño de presentación de datos en pantalla

* **Se es conciso con la información que se aporta.**
* **Es fácil de navegar.**
* **Estructura piramidal.**
* **El lenguaje es cercano al usuario.**
* **La información se presenta aislada.**