**PROTOTIPO DE ALARMA PARA AUTOMOVILES UTILIZANDO EL CONCEPTO DE IoT**

****

**LUIS ANGEL MONTENEGRO LOWY**

**DAVINSON CUERO MONTAÑO**

**CORPORACION UNIVERSITARIA AUTONOMA DEL CAUCA**

**FACULTAD DE INGENIERIAS**

**PROGRAMA DE INGENIERIA ELECTRONICA**

**POPAYAN,**

**PROTOTIPO DE ALARMA PARA AUTOMOVILES UTILIZANDO EL CONCEPTO DE IoT.**

****

TRABAJO DE GRADO PARA OBTAR AL TITULO DE INGENIERO ELECTRONICO

**LUIS ANGEL MONTENEGRO LOWY**

**DAVINSON CUERO MONTAÑO**

ESTUDIANTES

**MSc. Ing. JULIO ANDRES MOSQUERA BOLAÑOS**

DIRECTOR

**CORPORACION UNIVERSITARIA AUTONOMA DEL CAUCA**

**FACULTAD DE INGENIERIAS**

**PROGRAMA DE INGENIERIA ELECTRONICA**

**POPAYAN,**

**RESUSMEN**

En el presente trabajo se presenta de forma clara el desarrollo de un prototipo de alarma para automóviles, que consta de aplicación móvil, una comunicación con rede dedicada al IoT, y sobre todo con notificación remota.

**PALABRAS CLAVES**

Internet de las cosas(Iot), sigfox, aplicaciones móviles, notificaciones, Python, firebase cloud messaging(FCM).

Contenido

[CAPITULO 1: EL PROBLEMA 7](#_Toc52739127)

[1.1. PLANTEAMIENTO DEL PROBLEMA 7](#_Toc52739128)

[1.2. JUSTIFICACIÓN 8](#_Toc52739129)

[1.3. OBJETIVOS 9](#_Toc52739130)

[1.3.1. OBJETIVO GENERAL 9](#_Toc52739131)

[1.3.2. OBJETIVOS ESPECÍFICOS 9](#_Toc52739132)

[CAPITULO 2: MARCO TEÓRICO O REFERENTES CONCEPTUALES 10](#_Toc52739133)

[2.1. ANTECEDENTES 10](#_Toc52739134)

[2.1.1. NÚCLEO TEMÁTICO COMPONENTES DE SENSADO 10](#_Toc52739135)

[2.1.2. NÚCLEO TEMÁTICO COMPONENTES DE NOTIFICACIÓN 12](#_Toc52739136)

[2.1.3. NÚCLEO TEMÁTICO COMPONENTES DE COMUNICACIÓN 12](#_Toc52739137)

[2.1.4. NÚCLEO TEMÁTICO COMPONENTES DE IDENTIFICACIÓN 12](#_Toc52739138)

[2.2. BASES TEORICAS 13](#_Toc52739139)

[2.2.1. SEGURIDAD EN UN AUTOMÓVIL 13](#_Toc52739140)

[2.2.1.1. SISTEMAS PARA EVITAR CONDUCCIÓN DESCONTROLADA Y ACCIDENTES 13](#_Toc52739141)

[2.2.1.2. SISTEMAS CUANDO NO SE PUEDE EVITAR EL ACCIDENTE 13](#_Toc52739142)

[2.2.1.3. SISTEMA DE ENCENDIDO 13](#_Toc52739143)

[2.2.1.3.1. PARTES DEL SISTEMA DE ENCENDIDO TRAMO ELECTRICO 14](#_Toc52739144)

[2.2.1.3.2. PARTES DEL SISTEMA DE ENCENDIDO TRAMO ALIMENTACIÓN 15](#_Toc52739145)

[2.2.2. APLICACIONES MÓVILES 15](#_Toc52739146)

[2.2.2.1. APLICACIONES NATIVAS 16](#_Toc52739147)

[2.2.2.2. APLICACIONES WEB 16](#_Toc52739148)

[2.2.2.3. APLICACIONES HÍBRIDAS 16](#_Toc52739149)

[2.2.3. INTERNET DE LAS COSAS (IoT) 16](#_Toc52739150)

[2.2.4. ANDROID, ANDROID STUDIO 18](#_Toc52739151)

[2.2.5. PYTHON (DJANGO) 18](#_Toc52739152)

[2.2.6. FIREBASE 19](#_Toc52739153)

[2.2.7. HEROKU 20](#_Toc52739154)

[2.2.7. GITHUB 20](#_Toc52739155)

[2.2.8. DISPOSITIVOS HARDWARE IOT 20](#_Toc52739156)

[CAPITULO 3: METODOLOGÍA 24](#_Toc52739157)

[3.1. REQUERIMIENTOS 24](#_Toc52739158)

[CAPITULO 4: DESARROLLO E IMPLEMENTACIÓN DEL PROTOTIPO 26](#_Toc52739159)

[4.3. SIGFOX Y DESARROLLO WEB 26](#_Toc52739160)

[4.3.1. SIGFOX 26](#_Toc52739161)

[4.3.2. BASE DE DATOS 32](#_Toc52739162)

[4.3.2.1. SEGÚN LA VARIEDAD DE DATOS GUARDADOS 32](#_Toc52739163)

[4.3.2.1.1. BASE DE DATOS ESTÁTICA 32](#_Toc52739164)

[4.3.2.2.2. BASE DE DATOS DINÁMICAS 32](#_Toc52739165)

[4.3.2.2. FORMA DE ORGANIZACIÓN 32](#_Toc52739166)

[4.3.2.2.1. BASE DE DATOS JERÁRQUICA 32](#_Toc52739167)

[4.3.2.2.2. BASE DE DATOS DE RED 33](#_Toc52739168)

[4.3.2.2.3. BASE DE DATOS RELACIONAL 33](#_Toc52739169)

[4.3.3. DESARROLLO WEB 34](#_Toc52739170)

[4.4. IMPLEMENTACIÓN DE TECNOLOGÍAS 48](#_Toc52739171)

[4.4.1. ENVIÓ DE NOTIFICACIONES 48](#_Toc52739172)

[4.4.1.1 CONFIGURACION DE PARTE DEL SERVIDOR PARA ENVIO DE UN MENSAJE DE NOTIFICACIÓN 51](#_Toc52739173)

[4.5.1.2. CONFIGURACION DE PARTE DEL SERVIDOR PARA ENVIO DE UN MENSAJE DE DATOS 52](#_Toc52739174)

[4.4.2. GITHUB 53](#_Toc52739175)

[4.4.3. HEROKU 54](#_Toc52739176)

[4.4.4. BOOTSTRAP 55](#_Toc52739177)

[4.4.5. JAVASCRIPT 56](#_Toc52739178)

[4.5. DESARROLLO DE APP MÓVIL 56](#_Toc52739179)

[4.6. PUBLICACIÓN DE LA APLICACIÓN WEB EN INTERNET 56](#_Toc52739180)

**LISTA DE IMÁGENES**

[**Imagen 1:** Etapas del bloque electrónico 10](#_Toc52869438)

[**Imagen 2:** Esquema de conexión para la bomba de combustible 10](#_Toc52869439)

[**Imagen 3:** Rotor 13](#_Toc52869440)

[**Imagen 4:** Condensador 14](#_Toc52869441)

[**Imagen 5:** Cobertura de Sigfox a nivel país 26](#_Toc52869442)

[**Imagen 6:** cobertura de Sigfox a nivel departamento 27](#_Toc52869443)

[**Imagen 7:** Pasos para configuración del Callbacks 28](#_Toc52869444)

[**Imagen 8:** Ventana de configuración del Callbacks 29](#_Toc52869445)

[**Imagen 9:** Configuración del Callbacks para enviar a un email 30](#_Toc52869446)

[**Imagen 10:** Configuración del Callbacks para enviar a un servidor externo 31](#_Toc52869447)

[**Imagen 11**: Base de datos relacional 32](#_Toc52869448)

[**Imagen 12:** Instalación de Python 36](#_Toc52869449)

[**Imagen 13:** Progreso de instalación de Python 37](#_Toc52869450)

[**Imagen 14:** Instalación completa de Python 37](#_Toc52869451)

[**Imagen 15:** Verificar instalación de Python en la computadora 38](#_Toc52869452)

[**Imagen 16:** Iniciando consola para crear el entorno 38](#_Toc52869453)

[**Imagen 17:** Creación del entorno virtual 39](#_Toc52869454)

[**Imagen 18:** Activando el entorno 40](#_Toc52869455)

[**Imagen 19:** Instalación de Django 41](#_Toc52869456)

[**Imagen 20:** Instalación de MYSQL 41](#_Toc52869457)

[**Imagen 21:** Verificando instalación 42](#_Toc52869458)

[**Imagen 22:** Consola y explorador de archivos 43](#_Toc52869459)

[**Imagen 23:** Proyecto en Django 43](#_Toc52869460)

[**Imagen 24:**Configuración en el setting.py 44](#_Toc52869461)

[**Imagen 25:** Configuración de la Url 45](#_Toc52869462)

[**Imagen 26:** Creación de los archivos urls.py, forms.py 45](#_Toc52869463)

[**Imagen 27:** Diagrama de flujo de la aplicación web y móvil 46](#_Toc52869464)

[**Imagen 28:** Interface de Firebase 47](#_Toc52869465)

[**Imagen 29:** Proyecto creado en Firebase 48](#_Toc52869466)

[**Imagen 30:** Instalación de Firebase en la aplicación web 49](#_Toc52869467)

[**Imagen 31:** Comprobación de la instalación de Firebase 49](#_Toc52869468)

[**Imagen 32:** Configuración de Firebase en Python 50](#_Toc52869469)

[**Imagen 33:** Mensaje para cada aplicación 51](#_Toc52869470)

[**Imagen 34:** Solicitud para un envió de mensaje desde Python 52](#_Toc52869471)

[**Imagen 35:** Registro en Heroku 53](#_Toc52869472)

[**Imagen 36:** Creando una aplicación en Heroku 54](#_Toc52869473)

[**Imagen 37:** Aplicación en Heroku 54](#_Toc52869474)

[**Imagen 38:** Agregando Bootstrap al proyecto 55](#_Toc52869475)

[**Imagen 39:** Configuración de base de datos 56](#_Toc52869476)

[**Imagen 40:** Base de datos creada en Postgres 57](#_Toc52869477)

[**Imagen 41:** Ventana de datos de Postgres 57](#_Toc52869478)

[**Imagen 42:** Credenciales de la base de datos en Postgres 58](#_Toc52869479)

[**Imagen 43:** Configuración de la base de datos en el proyecto Python 59](#_Toc52869480)

[**Imagen 44:** Debug en True 59](#_Toc52869481)

[**Imagen 45:** Configuración de los archivos estáticos con Whitenoise 60](#_Toc52869482)

[**Imagen 46:** Configuración de los archivos estáticos 60](#_Toc52869483)

[**Imagen 47:** Configuración de la aplicación de Heroku, paso1 61](#_Toc52869484)

[**Imagen 48:**Conectando Heroku y el repositorio paso 2 61](#_Toc52869485)

[**Imagen 49:** Puesta en producción de la aplicación web 62](#_Toc52869486)

# CAPITULO 1: EL PROBLEMA

## PLANTEAMIENTO DEL PROBLEMA

termino seguridad es amplio y acoge varios campos como la seguridad ciudadana, humana, pública, laboral, vial, etc. [1]. Por lo cual este proyecto se basa en la seguridad vehicular (hurto de automóviles). Los sistemas de seguridad son importantes para custodiar los objetos que se consideran valiosos en nuestras vidas y a su vez para la protección de la misma [2]. Debido a la inseguridad vehicular que se está presentando a nivel continental las autoridades de 9 países hicieron una investigación para llevar a cabo una operación que condujera con la captura de más de mil personas y permitir la recuperación de 3.600 vehículos robados en el continente americano [3], entre ellos se encuentra Ecuador que se realizaron 101 capturas, recuperándose 141 vehículos y 2.854 autopartes, del mismo modo en Perú se hicieron 17 detenciones y se encontraron 186 automotores y 129 autopartes, a su vez en Chile se registraron 53 capturas y se hallaron 29 vehículos, toda esta investigación se llevó acabo en el año 2018 [4]. Según el General Jorge Nieto, director de la policía Nacional “Solo en Colombia, las operaciones conjuntas permitieron la recuperación de 386 vehículos y 1.189 motocicletas, avaluados en 25.000 millones de pesos, y de 1.561 autopartes” [3]. Esas cifras dan cuenta del poder de estos grupos, y cuya forma de operar se mostró en la publicación de la revista en el año 2013 de la Policía Nacional, habla sobre las modalidades más utilizadas por los perpetradores en el hurto de vehículos. En la declaración, que hacen los bandidos, afirman que existen varias formas de robar los carros las cuales son: “atraco”, “halado”, “llave maestra”, “estafa”, “el taco”, “falso accidente” [5]. Solo en Colombia se desarticularon 18 bandas y se capturaron a 249 personas sospechosas, en 14 capitales y 21 municipios [3]. Según las estadísticas que maneja la policía nacional del año 2018 solo en el departamento del cauca se reportaron 362 hurtos a vehículos, 61 de ellos fueron hurtados en el municipio de Popayán. Eso demuestra que los esfuerzos que hace la policía para frenar este flagelo, no son suficientes [6].

De acuerdo con las estadísticas anteriores se plantea la siguiente pregunta de investigación: ¿Cómo implementar un prototipo de alarma electrónica que notifique al usuario si su carro está siendo hurtado, utilizando el concepto de IoT?

## JUSTIFICACIÓN

Una de las actividades delictivas que se presenta en el departamento del Cauca es el hurto de automóviles. Según cifras publicadas por el Ministerio de Defensa, del 1 enero hasta el 31 de marzo de 2017 en el Cauca se hurtaron 64 automotores y en ese mismo periodo en el año 2018 se hurtaron 104 vehículos, el departamento del Cauca presento un incremento del 63% para el periodo del 2018. De los cuales en el municipio de Popayán se hurtaron 38 vehículos en el periodo del 2017 y 50 vehículos en el periodo del 2018, Popayán registró un incremento en el 32% de robo de automóviles para el periodo del 2018 [7]. Se han identificado dos tipos de hurtos de vehículos, el primero es cuando el perpetrador solo substrae las pertenencias que se encuentren dentro del automóvil y el segundo cuando se lleva el vehículo [8] [9].

El prototipo estará censando el interior del vehículo e informará al usuario cuando se ingrese de forma no permitida, de otra manera se le informará la ubicación si el automóvil se está moviendo sin consentimiento del usuario.

Las alarmas convencionales que tienen instaladas los automóviles funcionan de la siguiente manera: constan de una computadora central, sensores de apertura y cierre de puertas, sirenas, receptores de radio y baterías auxiliares. Los sensores están instalados en partes estratégicas del carro, pero solo sirven para saber si se ha abierto cualquier puerta, el capó o el baúl, estos a su vez están conectados a la computadora central, esta se comunica con el resto de los elementos como la sirena, las luces o la bocina. Cuando hay una posible entrada forzada al automóvil, enciende las luces del carro delanteras junto con las traseras ejecutando un parpadeo y también hace sonar la bocina, o solamente realiza la alerta la sirena. Otra parte esencial de las alarmas de los carros es el control remoto, este permite activar y desactivar la alarma a una distancia definida, por medio de radio frecuencia, con claves codificadas [10] [11].

Debido a que las alarmas convencionales que están instaladas en los carros no traen: un sensor de proximidad para saber si una persona está en el interior del vehículo y notificación utilizando el concepto de IoT, esto abre una brecha al prototipo que se plantea.

La mayoría de las alarmas para automóviles utilizan una sirena o la bocina, la cual se encarga de hacer ruido en el momento de detectar una irregularidad en el carro, dependiendo de la sensibilidad con la que haya sido configurada se puede activar dando falsas alarmas y a veces este ruido se vuelve molesto, sobre todo en las noches cuando las personas se disponen a descansar. Una persona tarda un promedio 20 segundos en apagar la alarma, si no lo hace este rudio puede prologarse hasta 30 segundos y volver a encenderse. En el sector residencial de Ponciano Bajo (Quito - Ecuador), donde vivía Santiago Viteri cada noche se activaba la alarma del carro de su vecino dos o tres veces, Santiago dice que el ruido de la alarma y también las luces que parpadean eran molestos y fastidiaban a unas 4 familias, el causante de que se activara la alarma e interrumpiera el sueño de estas familias era un gato, debido a la sensibilidad del sistema de seguridad con la que estaba configurado [12].

## OBJETIVOS

### OBJETIVO GENERAL

Implementar un prototipo de alarma electrónica para automóviles, que alerta al usuario utilizando el concepto de IoT.

### OBJETIVOS ESPECÍFICOS

* Diseñar el hardware de censado y validación del usuario de acuerdo a los requisitos del cliente.
* Notificar al usuario del estado de seguridad del vehículo mediante el concepto de IoT.
* Validar la implementación del prototipo en un ambiente controlado.

# CAPITULO 2: MARCO TEÓRICO O REFERENTES CONCEPTUALES

## 2.1. ANTECEDENTES

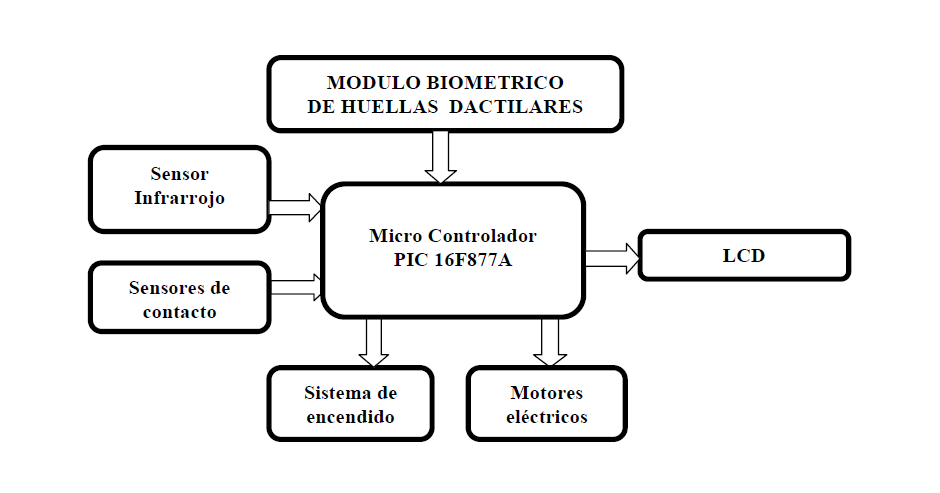
Esta sección será dividida en 4 partes para la descripción de los antecedentes, definidas por los núcleos temáticos. La sección 2.1.1 está relacionada con el componente de censado, la sección 2.1.2 se expone el componente de notificación, la sección 2.1.3 está relacionado con el componente de comunicación por último la sección 2.1.4 indica el componente de identificación.

Esta sección es importante para el desarrollo de este proyecto ya que se pudo ver en los siguientes trabajos que todos tenían fortalezas y debilidades. A continuación, se describen en relación por núcleos temáticos los aspectos más sobresalientes de cada una de las investigaciones.

### 2.1.1. NÚCLEO TEMÁTICO COMPONENTES DE SENSADO

* En el año 2014, el proyecto [2] desarrollado en Caracas-Venezuela llamado “desarrollo de un prototipo de sistema de seguridad biométrica para automóviles, controlado a través del Bluetooth por un dispositivo Android”, donde se usaron los dispositivos: sensor ZFM-20, el microcontrolador PIC16F870, es el cerebro de esta alarma, y el actuador, es un relé JZC-11F de 5VDC encargado de evitar el paso de corriente para que no pueda ser encendido el automóvil.
* La investigación [13] se realizó en la ciudad de Ambator-Ecuador en el año 2011, llamada “bloqueo electrónico en el encendido de un vehículo, para proporcionar un sistema de seguridad contra robos”, implementaron sensores en las puertas, seguros eléctricos y un sensor infrarrojo para detectar movimiento del carro. El controlador es un PIC 16F877A, los actuadores son los motores de bajar y subir los seguros eléctricos de las puertas, y un relé que evita que encienda la bomba de combustible para que no llegue combustible a la sección de ignición. La siguiente imagen es la ilustración del mecanismo general de cómo está integrada la alarma de este proyecto.

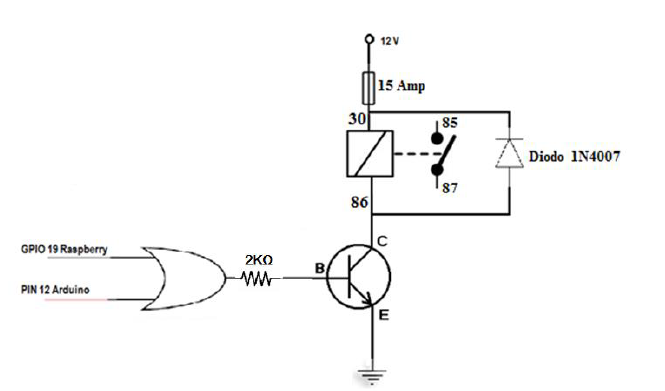
**Imagen 1:** Etapas del bloque electrónico



Fuente [13]

* El sistema de seguridad [14], en el año 2017 en Cuenca-Ecuador desarrollaron la investigación llamada “Diseño e implementación de un sistema de seguridad vehicular mediante reconocimiento facial a través de visión artificial”, emplearon dos controladores, un Arduino mega 2560 para activar y desactivar la alarma por medio de una clave ingresada por un teclado matricial de 4x4, una Rasberry pi 3 modelo B para uso del reconocimiento facial. Un actuador, el cual es un relé de 12V a 30A que permite o impide el encendido de la bomba de combustible. La imagen 2 muestra que el reconocimiento facial como el ingreso de la clave pueden activar o desactivar la bomba de combustible.

**Imagen 2:** Esquema de conexión para la bomba de combustible



Fuente: [14]

* Los autores de [15] en el año 2013 en el municipio de Pereira, desarrollaron el proyecto titulado “Implementación, control y monitoreo de un sistema de seguridad vehicular por redes GSM/GPRS”, donde usaron el controlador Arduino Uno, y los sensores: acelerómetro y monóxido de carbono.

### 2.1.2. NÚCLEO TEMÁTICO COMPONENTES DE NOTIFICACIÓN

* El proyecto [2] no especifica, que utilizaron para informar al usuario de alguna irregularidad que se presente en el vehículo.
* La investigación [13] utilizan una LCD16x2 para mostrar al usuario el estado que presente el automóvil, no especifican si utilizan notificación remota.
* En [14] notifican al usuario del estado del sistema mediante mensajes cortos en una pantalla LCD de 16x2.
* En este componente [15] dispuso la notificación por medio de mensajes de texto a un teléfono celular. Esto le permite notificación a distancia en caso de que el carro presente una irregularidad.

### 2.1.3. NÚCLEO TEMÁTICO COMPONENTES DE COMUNICACIÓN

* El proyecto [2] para el sistema de comunicación utilizó un módulo Bluetooth de la serie RN42, que se comunica con el dispositivo Android, este trae una opción de conexión por bluetooth. Esto le permite una notificación de corta distancia.
* La investigación [13] y [14] no especifican si utilizan una red para comunicación entre el usuario y la alarma.
* Para la notificación del proyecto [15], realizaron la comunicación entre el teléfono celular del usuario y la alarma por medio de la red GSM/GPRS, la cual permite el envío de la información a distancia.

### 2.1.4. NÚCLEO TEMÁTICO COMPONENTES DE IDENTIFICACIÓN

* En [2] y [13] implementaron un módulo biométrico ZFM-20 y un ADEL, para la lectura de huella digital en la identificación del usuario, estos envían una señal al controlador cuando se reconoce la huella del usuario y entonces desactiva la alarma y permitir el encendido de automóvil.
* El en proyecto [14] se fueron por la línea de la investigación por reconocimiento facial, utilizando una mini cámara web con entrada USB y utilizando el lenguaje de programación Python, en una Rasberry pi 3 modelo B para el procesamiento de las imágenes.
* [15] los autores no especifican si realizaron un componente de identificación para esta investigación.

## 2.2. BASES TEORICAS

Para el entendimiento de este proyecto es muy importante tener en cuenta los temas que se mencionan en toda esta sección.

### 2.2.1. SEGURIDAD EN UN AUTOMÓVIL

La seguridad en el automóvil son un conjunto de piezas o dispositivos situados e instalados en lugares estratégicos para evitar: accidentes, conducción descontrolada, mal funcionamiento, etc. Formando así los diferentes sistemas de seguridad.

#### 2.2.1.1. SISTEMAS PARA EVITAR CONDUCCIÓN DESCONTROLADA Y ACCIDENTES

Es la agrupación de aquellos elementos que conforman un sistema que permiten tener una buena conducción y estabilidad del automóvil cuando este se encuentre en movimiento, para así evitar un accidente. A continuación, se mencionarán algunos de estos sistemas:

* Sistema de frenos [16].
* Sistema de dirección [16].
* Sistema de suspensión [16].
* Sistema de iluminación [16].
* Los neumáticos y su adherencia a la carretera [16].

#### 2.2.1.2. SISTEMAS CUANDO NO SE PUEDE EVITAR EL ACCIDENTE

Estos sistemas tienen como objetivo reducir los daños a los ocupantes del vehículo en caso de un accidente. Algunos de los elementos son:

* El cinturón de seguridad [16].
* Las bolsas de aire [16].
* Chasis y carrocería [16].
* Reposacabezas [16].

#### 2.2.1.3. SISTEMA DE ENCENDIDO

Nos centraremos en este sistema ya que esta es una parte esencial del carro y por lo tanto también lo es para este proyecto ya que es la parte que intervendremos.

El sistema de encendido tiene como objetivo producir y distribuir un voltaje con la potencia suficiente para la generación de una chispa en cada una de las bujías, con ello encender la mezcla aire/combustible y generar la combustión. Hay que tener en cuenta que el sistema de encendido varia de pendiendo del combustible que utilicé el motor y modelo del automóvil [17].

##### 2.2.1.3.1. PARTES DEL SISTEMA DE ENCENDIDO TRAMO ELECTRICO

* **Batería:** Es un elemento fundamental de 12V, la que contiene la energía necesaria para activar los circuitos de las luces, parabrisas, accesorios, etc. Y lo más importante es la energización de la bobina y proveer de energía al motor de arranque [18] [17].
* **Llave de contacto:** Se encarga de abrir y cierra el circuito de encendido, a su vez acciona el motor de arranque [17] [19].
* **Bobina:** Es el elemento que convierte el voltaje suministrado por la batería en un voltaje de alta tensión que es requerida por las bujías, para poder generar la chispa y crear la combustión (quemar de la mezcla aire/gasolina). Esta bobina se divide en dos, una es llamada bobinado de baja o primario y la otra se llama bobinado de alta o secundario [17] [19].
* **Bujía:** Permiten hacer el salto de la chispa entre dos electrodos para hacer la combustión dentro de cada uno de los cilindros dentro del motor [17].
* **Distribuidor:** Este elemento tiene la función de repartir la tensión suministrada por la bobina a cada una de las bujías en el momento apropiado. Esta junto al motor y es movido por el árbol de levas [19].

**PARTES DEL DISTRIBUIDOR**

* **Rotor:** Esta construido de un material aislante y dispone de una lámina metálica en la parte superior donde recibe la alta tensión. Cuando el distribuidor gira también lo hace el rotor haciendo de esta manera, la adecuada distribución a las bujías, el contacto es por un arco voltaico que se da debido a la alta tensión [19].

**Imagen 3:** Rotor



Fuente [19]

* **Platino:** La función es trabajar como un interruptor que cambia su estado al ser accionado por una leva, abrir o cerrar el circuito con el bobinado de baja a las mismas revoluciones de giro del motor.
* **Leva:** Determina el estado de los contactos del platino, en el movimiento de este elemento genera dos ángulos, cuando los platinos están cerrados se llama ángulo de cierre y cuando están abiertos se llama ángulo de apertura [19].
* **Condensador:** Este elemento se utiliza como protección contra los picos de alto voltaje producidos por el bobinado secundario, ya que estos pueden dañar el platino causando un mal funcionamiento y en algunos casos impedir el flujo de la chispa [19].

**Imagen 4:** Condensador



Fuente [19]

##### 2.2.1.3.2. PARTES DEL SISTEMA DE ENCENDIDO TRAMO ALIMENTACIÓN

El propósito de estos sistemas es suministrar la cantidad adecuada de aire y combustible al interior de los cilindros para una buena combustión [17].

* **Tanque:** Tiene la función de almacenar el combustible, en su interior hay un flotador que funciona como sensor con la finalidad de indicar la cantidad de gasolina, consta de filtro que se encarga de limpiar el combustible que entra a los conductos que transportan el carburante al motor [17].
* **Bomba:** La función de esta es hacer circular o impulsar el carburante a través de los conductos hasta el carburador, esta puede ser eléctrica o mecánica [17] [18].
* **Carburador:** Tiene como objetivo proporcionar la adecuada medida de mezcla aire/carburante dependiendo de las condiciones de funcionamiento del motor [17].
* **Filtro de aire:** La tarea que realiza es la de evitar el paso de impurezas en el aire ya que estas pueden deteriorar el motor, está constituido de un elemento poroso que está unido con el carburador donde se produce la mezcla aire/combustible [17] [18].

### 2.2.2. APLICACIONES MÓVILES

Son programas diseñados que permiten al usuario realizar actividades profesionales, acceder a servicios, etc. Que se ejecutan en dispositivos que operen en determinado sistema operativo. Los tipos de aplicaciones son:

#### 2.2.2.1. APLICACIONES NATIVAS

Son aquellas aplicaciones en las que para ejecutarlas se debe tener en cuenta el tipo de dispositivo, el sistema operativo y su versión, ya que la principal ventaja que tiene este tipo de aplicaciones es que aprovechan al máximo las cualidades del hardware de cada dispositivo por ejemplo el GPS, acelerómetro, cámara, etc. Esto también genera más costo en el desarrollo ya que se debe crear la aplicación para cada sistema operativo [20].

#### 2.2.2.2. APLICACIONES WEB

Son aquellas aplicaciones que se ejecutan en el navegador del dispositivo móvil, están diseñadas con la tecnología HTML, CSS Y JAVASCRIPT que también sirve para desarrollar los sitios web. La ventaja que tiene el desarrollo de este tipo de aplicaciones es que no necesitan ser instaladas en los dispositivos, también se puede ejecutar en cualquier sistema operativo, solo necesitan tener instalado un navegador y conectividad a internet. Estas aplicaciones tienen una desventaja la cual es no poder utilizar el hardware del dispositivo [20].

#### 2.2.2.3. APLICACIONES HÍBRIDAS

Estas aplicaciones integran lo mejor de los tipos de aplicaciones nativas y web, utilizando tecnología multiplataforma como lo son el HTML, CSS Y JAVASCRIPT, ejecutándose dentro de un contenedor web sobre el dispositivo móvil. Una ventaja que tiene este tipo de aplicaciones es la reutilización de código para múltiples plataformas y estas ya tienen un acceso limitado a las capacidades hardware, la desventaja es que son más lentas en el momento de ejecución que una aplicación nativa y la apariencia no será como la de una aplicación nativa [20].

### 2.2.3. INTERNET DE LAS COSAS (IoT)

El concepto de IoT se refiere a un escenario de interconexión digital de objetos a una red de internet, como electrodomésticos, carros, bicicletas, cámaras o simplemente elementos como zapatos, maletas o cualquier otro que podamos imaginar. Mediante una identificación de dispositivo única y con sensores instalados en los objetos podemos saber dónde y cómo se encuentra este, sin importar la distancia a la que se encuentre el propietario [21].

Hay dos clases de tecnologías para desarrollar proyectos IoT, una es las tradicionales y las dedicadas. Veremos algunas de estas tecnologías a continuación [22].

**TECNOLOGIAS TRADICIONALES**

1. **WIRELESS FIDELITY (WIFI)**

Es una red que no tiene mucho alcance, pero permite él envió de datos con velocidades de hasta 11Mbps, por medio ondas de radio, y admite la conexión de varios dispositivos al mismo tiempo [23].

1. **ZIGBEE**

Es un estándar que está dado por un conjunto de protocolos con una capacidad de alta calidad de comunicación inalámbrica que permite hacer la utilización con radio digital de bajo consumo. Su objetivo principal es habilitar redes inalámbricas que tengan la capacidad de control y su monitoreo que sea confiable, de bajo consumo energético, de bajo costo y que funcionen vía radio y de modo bidireccional [24].

1. **BLUETOOTH SMART**

Es una tecnología inalámbrica de comunicaciones al servicio de algunos proyectos de IoT, permite la operación de dispositivos bluetooth y emplear el envió de pequeños paquetes de datos. Esta tecnología es de bajo consumo y corto alcance

1. **SISTEMA GLOBAL PARA LAS COMUNICACIONES MOVILES(GSM)**

La red GSM permite enviar de mensajes de texto, correos, mensaje de multimedia, videoconferencia, llamadas telefónicas etc. Este sistema abarca varios campos dentro de las comunicaciones: redes, telefonía y las radiocomunicaciones. Esta red se encuentra en todo el mundo y usa las bandas de frecuencia de 900MHz y 1800MHz, excepto Estados Unidos que utiliza la frecuencia 1900MHz, permite el envió de datos a larga distancia y gran tamaño [25].

**TECNOLOGIAS DEDICADAS**

1. **LORA Y LORAWAN**

LoRaWan es un protocolo de red que utiliza la tecnología LoRa para comunicar y administrar los dispositivos LoRa, y se compone de dos partes los nodos y Gateways, los Gateways son los responsables de enviar y recibir información a los nodos, los nodos son los dispositivos finales que se encargan de enviar y recibir información al Gateway. LoRa hace referencia a la tecnología patentada por Semtech, que tiene unas características como la baja transferencia de datos máximo que ronda en los 255 bytes, el largo alcance, conexión punto a punto, frecuencia de trabajo de 915Mhz en América [26] [27].

1. **SIGFOX**

Es una red de conectividad de teléfonos a nivel mundial dedicada al IoT, creada para la comunicación de baja velocidad, se basa en una banda muy estrecha (menor a 1KHz) para la comunicación, permitiendo así reducir el costo y consumo de energía de los dispositivos conectados, facilitando la comunicación a grandes distancias, ya que consta de antenas y estaciones base totalmente independientes de otras redes. Esta red permite la transferencia de datos a una velocidad máxima de 1000 bytes/s gracias al uso de UNB (ultra Narrow Band), y un peso máximo de 12 bytes por paquete de información a transmitir. Los dispositivos Sigfox tiene la capacidad de enviar máximo 140 mensajes por día y recibir 4 por día.

Con el uso de esta tecnología se puede monitorear consumo de energía, agua, gas, variables de confort interno del edificio, variables de interés en el sector agrícola, etc.

### 2.2.4. ANDROID, ANDROID STUDIO

Es un sistema operativo creado para dispositivos móviles, desarrollado por Google, y se basa en Linux, las principales características que tiene este sistema es que puede adaptarse a varias resoluciones de pantalla, visualización de varios formatos de multimedia y permite conexiones de WIFI, Bluetooth, LTE, GSM, etc.

En la creación de una aplicación para el sistema operativo Android, lo primero que se debe hacer es cumplir unos requisitos, que son necesarios para que la computadora pueda ser utilizada en el desarrollo de estas aplicaciones. A continuación, se describe el software necesario para el desarrollo de una aplicación nativa de Android:

* **ENTORNO DE DESARROLLO INTEGRADO (IDE)**

“Es un entorno de programación empaquetado como un programa o aplicación, que provee un marco de trabajo agradable, consta de unas características básicas: un editor de código, compilador, depurador y constructor de interfaz gráfica” [28]. El IDE oficial de Google para la creación de aplicaciones para Android es: Android Studio [29].

* **SDK**

Es el encargado de proporcionar las librerías de desarrollo necesarias para crear, probar y depurar aplicaciones Android. Consta de un depurador de código, documentación y un emulador (AVD) que se ejecuta en nuestra computadora [30].

* **JDK**

“Es un software que provee herramientas de desarrollo para la creación de programas en Java” [30].

### 2.2.5. PYTHON (DJANGO)

Python es un lenguaje de programación interpretado versátil multiplataforma orientado a objetos y sobre todo es de tipado dinámico, que nació en los años 90 en Holanda desarrollado por Guido Van Rossum, de licencia gratuita y código abierto que funciona en servidores de aplicaciones Android, Windows, Linux, etc. Es un lenguaje sencillo y legible que costa de varios frameworks donde se aprovecha al máximo el potencial de este lenguaje, tiene un framework para cada actividad: desarrollo web, bigdata, inteligencia artificial, entre otras [31]. donde hoy en día va en la versión 3.8.5 de Python, se puede descargar de la página oficial (<https://www.python.org/downloads/release/python-385/>).

“Un lenguaje interpretado es cuando utiliza un intermediario llamado bycode, que se crean en la primera vez que se ejecuta creando unos archivos con la extensión py en lugar de compilar el cogido a lenguaje máquina” [32].

Django es framework de Python dedicado al desarrollo de aplicaciones web que permite el desarrollo de esta rápido y seguro, es bastante conocido para todos los que han programado alguna vez en Python, cuenta con una gran documentación. Para el desarrollo de una aplicación web en django hay que tener en cuenta que consta de un orden y patrón de diseño modelo vista plantilla(MVT) [33].

**Modelos(models):** son objetos de Python que especifica la estructura de la tabla de la base datos de una aplicación. Por medio de esta clase se puede realizar el CRUD (crear, leer, actualizar y borrar) registros de la base de datos, usando código Python en lugar de escribir sentencias SQL, los modelos son una subclase de (from django.db import models) [33] [34].

**Vistas(views):** son las que dominan o contienen la lógica de la página, estas pueden ser vistas basadas en clases o en funciones. Estas se encargar de recibir las solicitudes HTTP por medio de un método POST o GET y acceder a los modelos y determinar qué datos serán visualizados en una solicitud [33] [34].

**Urls:** son las encargadas de especificar que vista es llamada según la petición o solicitud HTTP, y son conocidas como URLConf, el objetivo de estas son interpretar la URL solicitada por el usuario, llamar la vista correspondiente a la solicitud y pasar variables en caso de que lo requiera [33].

**Plantillas(templates):** es una plantilla HTML que contiene el diseño de la página e insertar elementos a Python por medio de los formularios, estas se encargar de recibir la información de las vistas y organizarlas para mostrarlas en el navegador web [33].

**Formularios(forms):** son los encargados de validar y mostrar formularios HTML [33].

### 2.2.6. FIREBASE

Es una plataforma de Google que permite la creación de aplicaciones web y móviles de forma rápida y sencilla proporcionando un servidor backend para estas. Firebase cuenta con una amplia gama de herramientas:

base de datos en tiempo real no SQL. Autenticación: registro e inicio de sesión por correo y también a través de los proveedores Facebook, Twitter, Github y Google.

almacenamiento de archivos. Hosting que se usa para la publicación de la aplicación. Test lab es una herramienta que se usa para probar la aplicación antes de publicarla. Crash reporting esta herramienta se encarga de reportar errores de las aplicaciones, y el más importante para el desarrollo de este proyecto cloud messaging, esta herramienta permite el envió de mensajes y notificaciones de forma segura a los usuarios registrados en las aplicaciones y es multiplataforma, esta herramienta está incluida todos los paquetes [35].

Firebase te da la opción de crear tus aplicaciones gratuitas, con un máximo de 99 usuarios que usen la aplicación, este paquete lo llaman Spark, está también el paquete Flame que hay que pagar USD 25 al mes esta tarifa es fija, y por ultimo está el Blaze que la tarifa depende del gasto mensual del plan [35].

### 2.2.7. HEROKU

Es una solución de computación en la nube y basada en contenedores Linux, es usada para implementar y administrar las aplicaciones por los desarrolladores. Permite el despliegue de aplicaciones creadas en lenguajes Node.js, RUBI, JAVA, PHP, GO, SCALA, CLOJURE y PYTHON. Heroku incluye una versión gratuita y una paga que cuesta unos USD 7,00 al mes [36] [37].

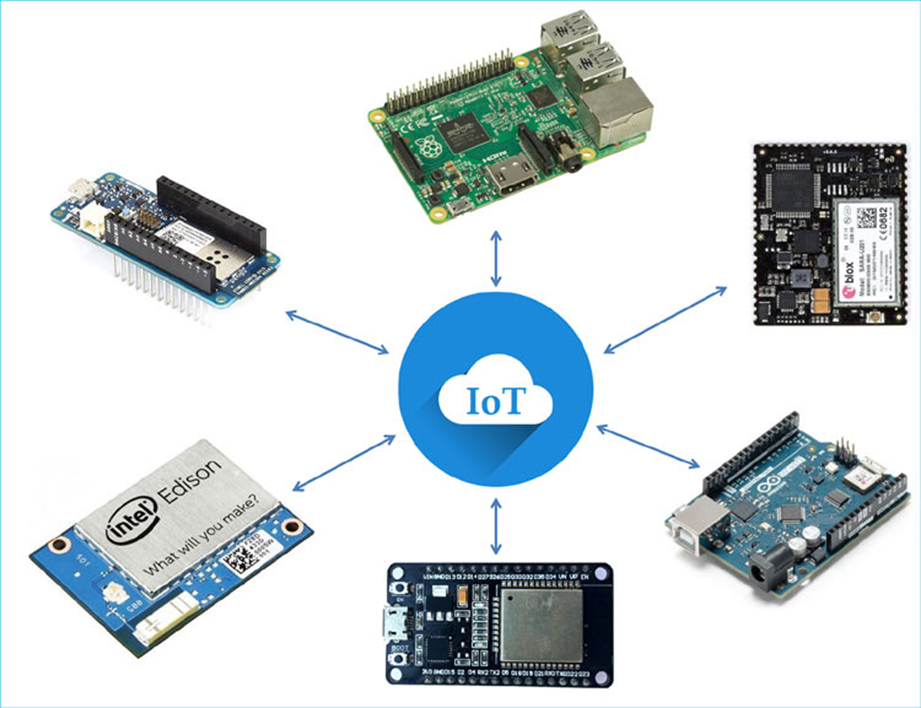
### 2.2.7. GITHUB

Es una plataforma de desarrollo de software colaborativo que alberga documentos Word, código arduino, Python, desarrollo web, etc. en un repositorio, que permite el trabajo en grupo lo cual da acceso a estos para que puedan hacer actualizaciones al código permitiendo así utilizar una herramienta llamada control de versiones [38].

### 2.2.8. DISPOSITIVOS HARDWARE IOT

El hardware cumple una función muy importante en la realización de proyectos IoT. Son piezas fundamentales que posibilitan a los objetos de la vida cotidiana interactúen entre ellos y los seres humanos atreves del internet o redes dedicadas, recopilando información del entorno o interactuado con él. Estos dispositivos cada vez facilitan su integración con cualquier objeto.

IoT utiliza dispositivos electrónicos capaces de medir magnitudes físicas o químicas y transformarla en señales eléctricas (sensores). Estos dispositivos combinados con la capacidad de conexión forman la capa de hardware de IoT [39].



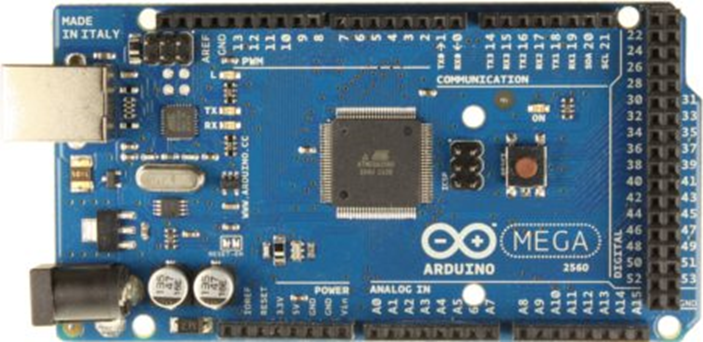
**ARDUINO**

Es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo.

También se utiliza en el desarrollo de objetos interactivos autónomos o puede conectarse a un PC a través del puerto serie utilizando lenguajes como Flash, Processing, MaxMSP, PureData, etc Las posibilidades de realizar desarrollos basados en Arduino tienen como límite la imaginación. Asimismo, su sencillez y su bajo coste, recomiendan su uso como elemento de aprendizaje e iniciación en el mundo de la electrónica digital.

para el desarrollo se trabajó en un arduito mega debido a que cuanta con más entradas y salidas digital a comparación de las otras placas

**Arduino mega**



Características

• Microcontrolador: ATmega2560

• Voltaje Operativo: 5V

• Voltaje de Entrada: 7-12V

• Voltaje de Entrada(límites): 6-20V

• Pines digitales de Entrada/Salida: 54 (de los cuales 15 proveen salida

• Pines análogos de entrada: 16

• Corriente DC por cada Pin Entrada/Salida: 40 mA

• Corriente DC entregada en el Pin 3.3V: 50 mA

• Memoria Flash: 256 KB (8KB usados por el bootloader)

• SRAM: 8KB

• EEPROM: 4KB

• Clock Speed: 16 MHz

**MÓDULO GPS NEO-6M V2**

**SENSOR NEO-6M**

este sensor posee una antena GPS, que permite obtener obtener la latitud y longitud para así, posteriormente, hallar la ubicación del dispositivo.

La comunicación de este periférico con el Arduino se realiza mediante comunicación serial, siendo simplemente necesario para la obtención de datos la configuración de dos pines cualesquiera, como puerto de recepción y puerto de transmisión (RX, TX).

Características

• Modelo: GY-GPS6M V2

• Tensión de alimentación: 3 ~ 5V

• Antena integrada

• EEPROM para el registro de los datos

• Batería de copia de seguridad

• Modo de ahorro de energía

• LED indicador de señal

• Posee 5 agujeros de 3mm para fácil fijación

• Velocidad de transmisión estándar: 9600bps

• Compatible con varios controladores de vuelo



# CAPITULO 3: METODOLOGÍA

Se utilizará la metodología de la investigación iterativa e incremental, pues la iterativa se identifica por ejercer el papel de verificar la funcionalidad de cada actividad de un sistema, la incremental se caracteriza por permitir que cada una de estas sea mejorada y con cada mejora se permite entregar parte de la funcionalidad requerida por el sistema. Este proceso se hace repetitivo cumpliendo con la entrega de cada mejora hasta que se llegue al proceso final. Por lo cual así poder dar el cumplimiento de cada uno de los objetivos planteados con sus respectivas tareas, para lograr finiquitar el proyecto con éxito.

### 3.1. REQUERIMIENTOS

Los requerimientos para el desarrollo de este prototipo partieron del análisis de los sistemas de alarmas convencionales o tradicionales, también hubo un gran aporte por la encuesta que se hicieron a 20 personas que fueran dueños o conductores de vehículos. Estos se analizan con el **reglamento N°97 de la CEPE** **de homologación de los sistemas de alarma para vehículos.**

Primer requerimiento para el prototipo es que se pueda saber el estado del vehículo sin importar la distancia a la que se encuentre el usuario, con este requisito ambas partes de donde se obtuvo la información concuerdan que es necesario la implementación de tecnología que permita él envió de información remota, acerca del estado del carro a una aplicación móvil.

El segundo requisito fue aportado por la información obtenida de la encuesta, el cual es que en el caso de hurto del carro no se pueda encender, se logre saber la posición final en donde se encuentra el automóvil en caso de robo. El tercer requerimiento es obtenido a través de hacer un análisis a las alarmas convencionales donde se concluyó que estos sistemas no cuentan con un sensor para que alerte en caso de que una persona entre rompiendo cualquier vidrio del carro, asimismo incluir un sistema de validación de usuarios.

**VALIDACION DE LO REQUISITOS OBTENIDOS, POR MEDIO DEL REGLAMENTO N° 97 (CEPE) DE HOMOLOGACION DE LOS SISTEMAS DE ALARMA EN LOS VEHICULOS**

Este reglamento nos dice que un sistema de alarma para vehículos, diseñado para proporcionar seguridad adicional en el caso del uso no autorizado del vehículo, debe contener:

Sensor: es el dispositivo que detecta el ingreso o uso del vehículo sin autorización [40].

Avisador: dispositivo que revela que ha sucedido un acceso no autorizado al vehículo [40].

Equipo de control: dispositivo encargado de la activación, desactivación del sistema de alarma y activar los avisadores en caso de intrusión o intervención en el carro [40].

Llave: dispositivo único diseñado para operar la alarma [40].

Inmovilizador: dispositivo que se encargar de impedir el movimiento del vehículo con la fuerza de su propio motor [40].

El sistema de alarma emitirá una señal de aviso en caso de intervención o intrusión, dicha señal debe ser acústica y puede ir acompañada de dispositivos luminosos o ser transmitida por radio, o combinación de ambas. El sistema debe ser capaz de detectar la apertura de cualquiera de las puertas [40].

El reglamento también no da unos parámetros y condiciones de ensayo, donde dice que el sistema debe soportar una temperatura de -40 a 85°C para los dispositivos instalados en el habitáculo o bodega, y -40 a 125°C para piezas que se instalen en el compartimiento del motor, además debe tener una tolerancia a los sobrevoltajes, en una temperatura de 23°C donde se someterá a un voltaje de 15voltios durante una hora y 24 voltios durante 1 minuto [40].

Este reglamento acoge los vehículos de categoría M1 y N1 [40], donde los M1 son los vehículos que tiene capacidad de 8 asientos sin contar el del conductor, los N1 son los destinados al transporte de cargas inferiores a 3,5 toneladas [41].

Como se pueden ver los requisitos obtenidos por medio de la encuesta y el análisis a los sistemas de alarmas convencionales, están acobijados por este reglamento. Donde los sensores utilizados en el desarrollo del prototipo son un sensor de presencia en el habitáculo, un GPS, un ARDUINO es el encargado del control, un lector de RFID es la llave de encargada de activar y desactivar la alarma. El inmovilizador es un RELE, y el avisador es una notificación que va directamente al dispositivo Android del usuario o dueño del automóvil.

**FASE PREPARATORIA**

En esta se logra las actividades con la identificación de los dispositivos a usar partiendo de los requerimientos, entornos de desarrollo de aplicaciones móvil, identificación de conexión a IoT. Las actividades se mencionan a continuación.

* Mediante revisión bibliográfica seleccionar cuales son los dispositivos que se ajustan a los requisitos obtenidos, dada que la selección de estos se da por funcionalidad, precio y documentación, teniendo en cuenta normas en el diseño de alarmas para carros.
* Escoger el entorno de desarrollo de la aplicación móvil dada por tipo de licencia (gratuita) y que se obtenga buena documentación.
* Seleccionar la red por cobertura, precio y fácil adquisición del dispositivo de conexión a la red.

**FASE DESCRIPTIVA**

Comprende a las actividades de recolección de información, funcionamiento de todos los dispositivos, la aplicación móvil y la búsqueda de normas o reglas para los diseños de las alarmas en los vehículos.

* Verificación de funcionamiento de los dispositivos en los componentes de comunicación, censado, identificación, basados en el componente de seguridad.
* Analizar y desglosar la información dada por cada dispositivo para luego ser entregada al componente de interface de usuario.
* Documentación del concepto de IoT, y qué relación tiene con Sigfox.

**FASE INTERPRETATIVA POR NÚCLEOS TEMÁTICOS**

Esta sección consta de las siguientes actividades:

* Se define como va hacer tratada la información obtenida de cada sensor y presentarla al usuario de una forma pertinente.
* Realizar el adecuado envió de información por medio de la notificación.
* Determinar estrategias para el cómo y dónde instalar los dispositivos en el vehículo para el ambiente controlado.

# CAPITULO 4: DESARROLLO E IMPLEMENTACIÓN DEL PROTOTIPO

Este capítulo de divide en 6 actividades, donde se explica detenidamente el paso a paso el desarrollo e implementación del **sistema prototipo de alarma para vehículos,** las actividades tienen un orden lógico y estas son: 4.1. componente de censado, 4.2. identificación y validación de usuario, 4.3. Sigfox y desarrollo web, 4.4. integración de tecnologías, 4.5. creación de la app, 4.6. publicación de la aplicación web en internet y 4.7. instalación y evaluación del prototipo en entorno controlado.

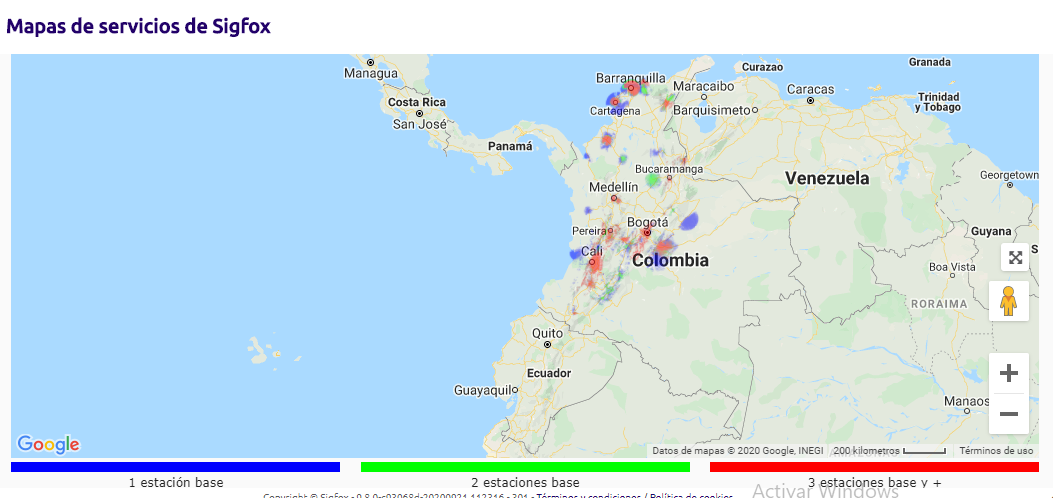
## 4.3. SIGFOX Y DESARROLLO WEB

### 4.3.1. SIGFOX

Se eligió la red de Sigfox para el desarrollo del prototipo ya que es una red dedicada a IoT, su bajo costo, su rango de conexión, fácil acceso a los dispositivos, y sobre todo se acopla muy bien al prototipo, porque en el caso de que los ladrones lleven el vehículo a una zona donde no hay cobertura de internet no hay forma de saber su ubicación, aquí es donde entra Sigfox, y su ventaja es que esta red tiene sus propias antenas para la conexión y envió de información, Sigfox consta de un backend donde se almacena toda la información y se direcciona la información enviada por los dispositivos, en este caso a una página web.

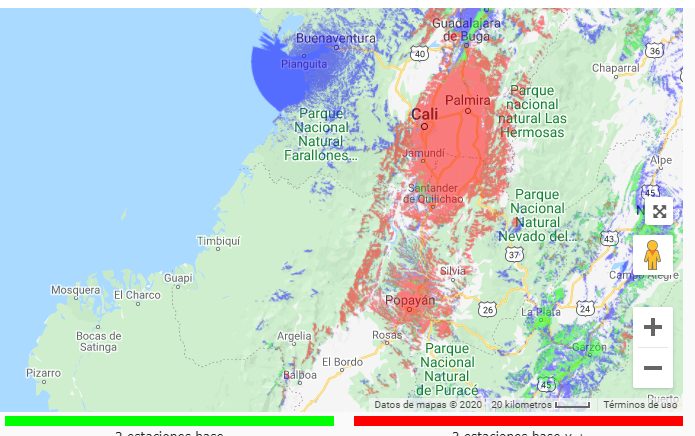
En la imagen 5 se puede ver el mapa de cobertura de la red de Sigfox en Colombia, donde este país está en fase de implementación de esta red, se puede apreciar que las ciudades de Barranquilla, Bogotá Medellín, Cali Popayán, etc. y la imagen 6 se puede ver que la ciudad de Popayán esta todo cubierto y como es la zona donde se desarrolla el proyecto esto nos favorece.

**Imagen 5:** Cobertura de Sigfox a nivel país



Fuente: [42]

**Imagen 6:** cobertura de Sigfox a nivel departamento



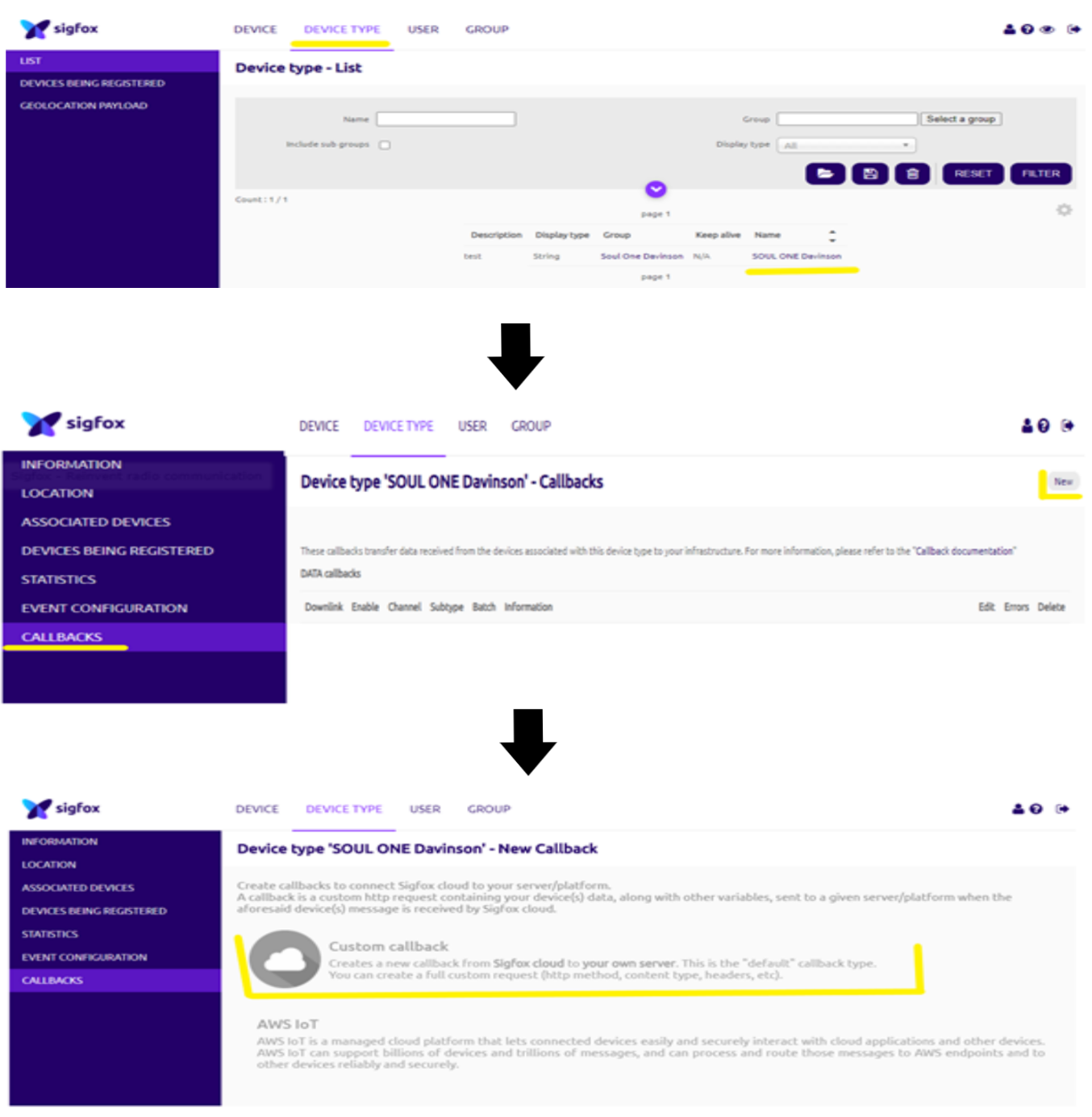
Fuente: [42]

**CONFIGURACIÓN DEL BACKEND DE SIGFOX**

Consta de un Callback o devolución de llamada que tiene como objetivo enviar la información que se recibe de los dispositivos a una plataforma o servidor que permita la manipulación de dichos datos por medio de una solicitud GET, POST, o también puede ser enviado a un correo electrónico, ya que el backend no tiene la capacidad de tomar decisiones por sí solo dependiendo de la información que se recibe. Para hacer la configuración del Callback es necesario que el dispositivo se encuentre registrado.

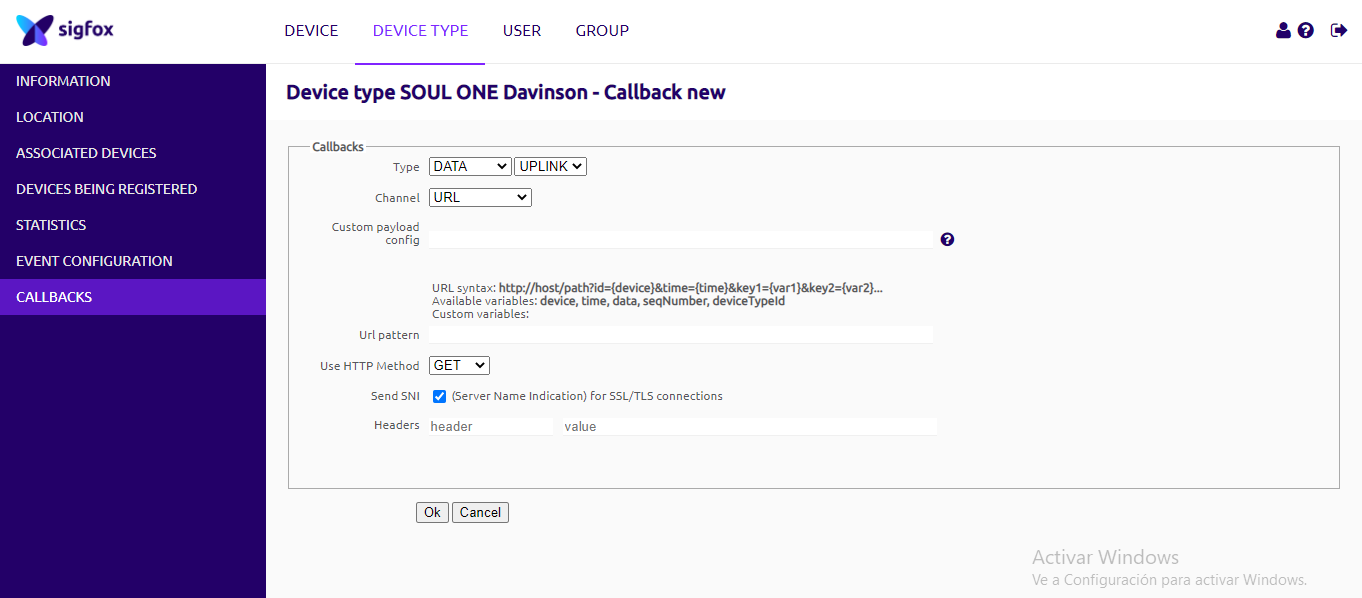
Como se muestra en la imagen 7, dados click en **DEVICE TYPE** y luego no dirigimos al nombre del dispositivo, luego nos abre una página, en la parte inferior izquierda buscamos donde dice **CALLBACKS** y damos click, buscamos en la parte superior derecha donde dice **NEW** y damos click, nos muestra unas opciones y escogemos la que dice **CUSTOM CALLBACKS** y obtenemos una ventana para configurar como muestra en la imagen 8.

**Imagen 7:** Pasos para configuración del Callbacks



**Fuente: [propia]**

**Imagen 8:** Ventana de configuración del Callbacks



**Fuente: [propia]**

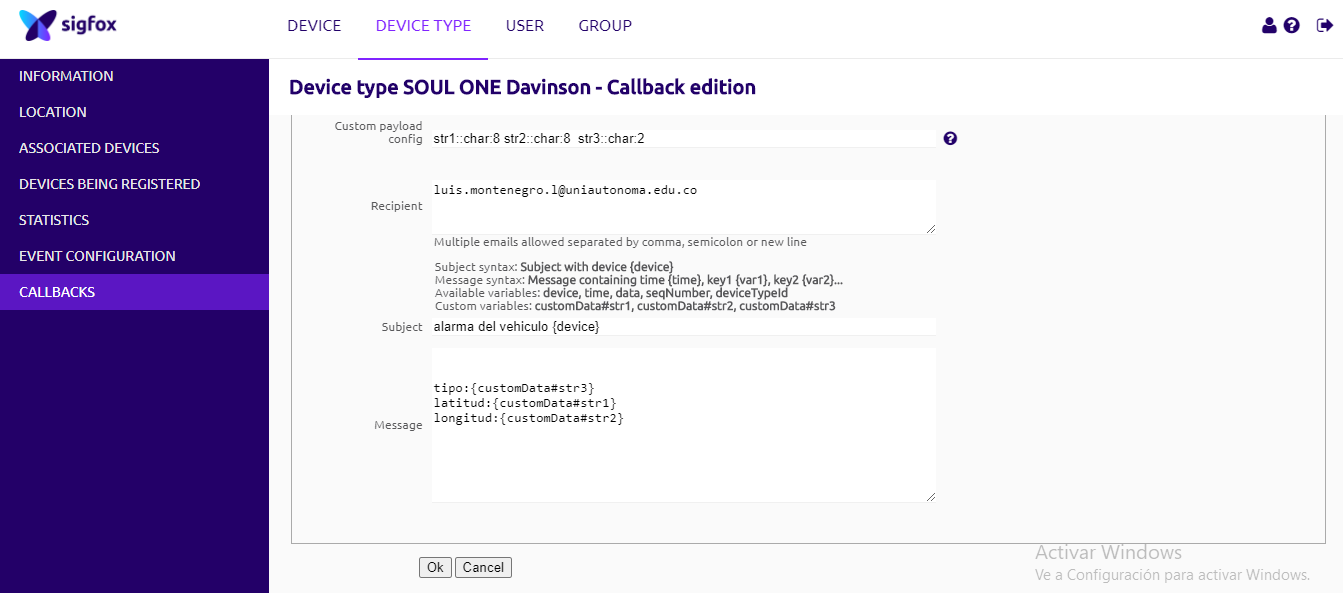
En la imagen 8 se muestra algunos aspectos que deben ser configurados, a continuación, los cuales se explican con sus respectivas opciones en la siguiente tabla:

**Tabla 1:** Atributos y Opciones

|  |  |  |
| --- | --- | --- |
| Atributo | Opción 1 | Opción 2 |
| **TYPE:** tipo de mensaje que se desea enviar. | **DATA:** carga útil del mensaje | **UPLINK:** permite la comunicación en un solo sentido(backend-device). |
| **BIDIR:** permite la comunicación en ambos sentidos (backend-device, device-backend). |
| **SERVICE:** | **STATUS:** envía información sobre batería y temperatura del dispositivo |
| **ACKNOWLEDGE:** confirma la emisión del mensaje solo de bajada en cualquier modo de enlace descendente. |
| **REPEATER:** envía detalles de batería y numero de mensajes repetidos. |
| **DATA\_ADVANCE:** se envía con retraso de 30 segundos. |
| **ERROR:** se activa cuando el dispositivo no cumple con el intervalo de comunicación. | N/A |
| **CHANEL**: | **URL:** envía la información a una página web o servidor externo. | N/A |
| **BATCH URL:** |
| **EMAIL:** envía la información directo a un correo de nuestra elección. |
| **USE HTTP METHOD**: | **POST:** envía información. | N/A |
| **GET:** obtiene información de recurso. |
| **PUT:** actualiza información en un servidor o recurso. |

**Fuente: [propia]**

**Imagen 9:** Configuración del Callbacks para enviar a un email



**Fuente: [propia]**

**Imagen 10:** Configuración del Callbacks para enviar a un servidor externo



**Fuente: [propia]**

### 4.3.2. BASE DE DATOS

La base de datos es una parte muy importante en los proyectos cuando se trabaja con información. En estas podrás registrar y guardar los datos más sobresalientes de tus compradores, lecturas de sensores, etc. Hay varios tipos de base de datos para el tratamiento de dicha información los cuales se hace una breve mención de algunos de estos tipos.

#### 4.3.2.1. SEGÚN LA VARIEDAD DE DATOS GUARDADOS

##### 4.3.2.1.1. BASE DE DATOS ESTÁTICA

Esta base de datos es conocida también como base de solo lectura, utilizadas para principalmente el registro y almacenamiento de la información para después ser estudiada y permitir el entendimiento del comportamiento de dicho conjunto de datos a través del tiempo [43].

##### 4.3.2.2.2. BASE DE DATOS DINÁMICAS

A diferencia de las estáticas esta base de datos permite que la información almacenada pueda ser gestionada [43].

#### 4.3.2.2. FORMA DE ORGANIZACIÓN

##### 4.3.2.2.1. BASE DE DATOS JERÁRQUICA

Son base de datos organizadas en las que la información almacenada esta en forma de un árbol al revés con un orden de importancia. La estructura se construye a partir de divisiones que se conocen como nodos y ramas. Para construir esta base de datos hay que tener en cuenta las siguientes instrucciones:

El diseño de esta base de datos consta de un segmento llamado Raíz, está situado en la parte superior del árbol. Luego está el segmento llamado Padre, este despende descendientes y todos los segmentos padres están al mismo nivel, pueden desprender varios nodos hijos. Por ultimo está el segmento Hijo que es el que desprende del segmento padre, cada uno de los hijos deben tener un solo padre. Una característica de esta base de datos es que son difíciles de modificar después de haber sido creadas [44].

##### 4.3.2.2.2. BASE DE DATOS DE RED

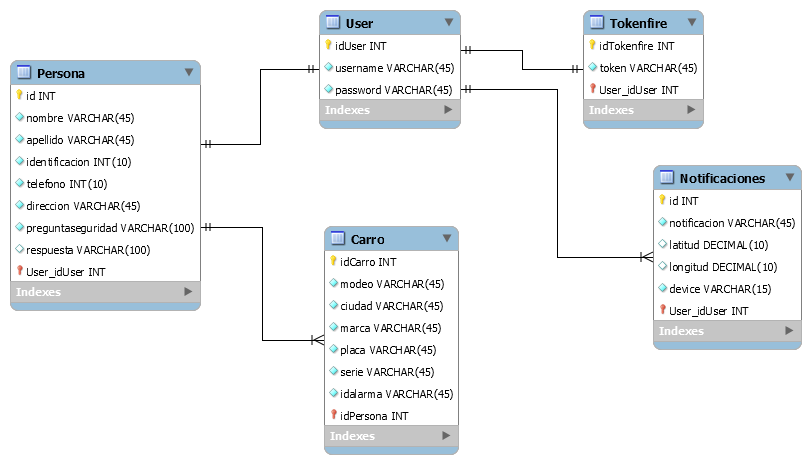
Es una variación de la base de datos jerárquica, la diferencia es que en esta un segmento hijo puede tener varios segmentos padres. En la actualidad estos tipos de base de datos no es de uso frecuente ya que consta de nivel de dificultad en su modificación y adaptación de su estructura [44].

##### 4.3.2.2.3. BASE DE DATOS RELACIONAL

Hoy en día son las más utilizadas en la actualidad, permite gestionar los datos y hacer relación entre ellos. Cada registro es almacenado de forma ordenada en tablas, así permite establecer las relaciones existentes entre datos de forma rápida, sencilla para conceder reportes y análisis. Sus características principales son: fácil gestión, acceder a los datos con rapidez [44].

Para la implementación de este proyecto se decidió utilizar base de datos relacional, ya que esta base de datos nos permite organizar la información de forma clara y precisa. En la siguiente imagen se muestra la base de datos relacional creada para el proyecto.

**Imagen 11**: Base de datos relacional



**Fuente: [propia]**

### 4.3.3. DESARROLLO WEB

El desarrollo de aplicaciones web está en crecimiento y por lo tanto muchos lenguajes de programación dedicados al desarrollo de estas, a continuación, se mencionan algunos lenguajes tales como lo es PHP, JAVASCRIPT, PYTHON, etc. Para el desarrollo de la aplicación web de este proyecto se utiliza PYTHON con el FRAMEWORK DJANGO.

**Tabla 2:** Comparación de lenguajes de programación para aplicaciones web

|  |  |  |
| --- | --- | --- |
| Lenguajes | Ventajas | Desventajas |
| PYTHON | Es multiplataforma. | El despliegue no es fácil para la primera vez. |
| Ordenado y limpio. |
| Rápido desarrollo y sintaxis simplificada. |
| Gran comunidad | Todos los hosting no son compatibles |
| Diseñado para propósito general. |
| Fácil aprendizaje |
| PHP | Multiplataforma | Mas líneas de código |
| Orientado al de desarrollo de aplicaciones web dinámicas. |
| Muchos framework para el desarrollo de aplicaciones web | Más tiempo para aprender. |
| Fácil despliegue. |
| JAVASCRIPT | Además del diseño web también, se usa en las aplicaciones móviles. | Los códigos son visibles en los navegadores. |
| Está dedicado más a la parte del FRON END. | Es inseguro. |
| Multiplataforma. | Los usuarios pueden desactivar JavaScript en los navegadores. |
| Permite crear interfaces dinámicas. |

**Fuente: [propia]**

En los siguientes gráficos se puede observar los tres lenguajes de programación para aplicaciones web que más han sido buscados a nivel mundial. La información con la que se hizo los gráficos fue tomada de Google Trends [44]. El periodo para la gráfica fue desde el 4/10/2015 hasta el día de la consulta (27/09/2020) el rango de tiempo es de aproximadamente 5 años. Donde Python tuvo una suma total de búsquedas mayor a los otros lenguajes donde según Google los 5 países que más buscaron Python fueron: China con un 58% seguido de Singapur con un 56%, Israel con 54%, Estados Unidos con 53% y el quinto lugar es para Irlanda con 49%. Donde Colombia se encuentra en el puesto 46 de 67 países con un 30% al aporte de búsquedas sobre Python [44], ver gráfico 3. Hay que ver el grafico 2 para ver que el inicio de Python en este periodo, donde se muestra que no siempre ha estado en el primer puesto de búsquedas a nivel mundial. En la semana del 27 de septiembre del 2015, Python estaba muy por debajo entre estos 3 lenguajes, las buenas noticias para Python empiezan el 15 de enero del 2017 Python supera a PHP y luego a JavaScript en la semana del 4 de febrero del 2018, de la semana del 20 de septiembre de 2020, se pone como líder a nivel mundial como el lenguaje para aplicaciones web más buscado. Hay que aclarar los 3 lenguajes fueron escogidos por los desarrolladores de este prototipo.

**Grafico 1:** 3 lenguajes para la programación de aplicaciones web más buscados

**Fuente: [propia]**

**Grafico 2:** Lenguajes para programación de aplicaciones web más buscadas en el periodo de 5 años

**Fuente: [propia]**

**Grafico 3:** Lenguajes buscados para programación de aplicaciones web en Colombia

**Fuente: [propia]**

**INSTALACIÓN DE PYTHON Y TODAS LAS LIBRERIAS NECESARIAS PARA EL PROYECTO.**

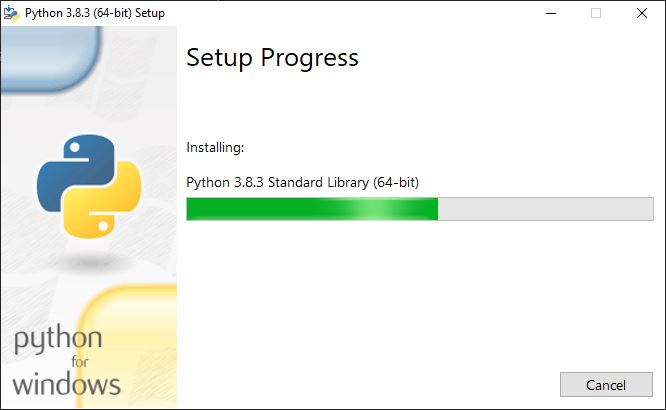
Ahora procedemos a instalar lo necesario para el desarrollo de la aplicación web en Python. Primero descargamos Python del siguiente link <https://www.python.org/downloads/release/python-383/> de donde descargamos la versión [Windows x86 executable installer](https://www.python.org/ftp/python/3.6.4/python-3.6.4.exe), y procedemos a instalarlo en la computadora. Nos muestra una venta como se ve en la imagen 11, primero damos click en **add Python 3.8 to PATH** es para agregar las variables de entorno al sistema del computador, luego pulsamos en **install now,** automáticamente aparece una venta con la barra de progreso de instalación como se muestra en la imagen 12. Por ultimo aparece una ventana como la de la imagen 13, y damos en **close**

**Imagen 12:** Instalación de Python



**Fuente: [propia]**

**Imagen 13:** Progreso de instalación de Python



**Fuente: [propia]**

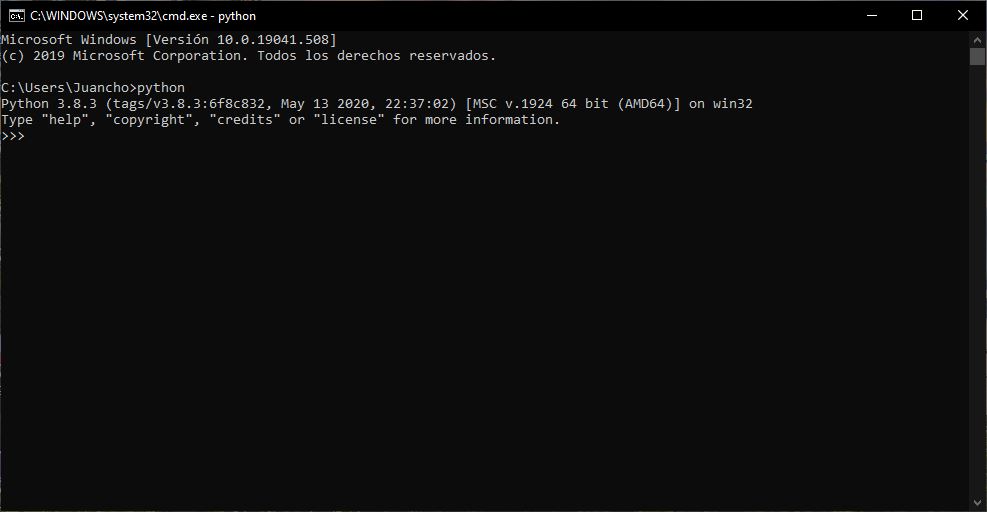
**Imagen 14:** Instalación completa de Python



**Fuente: [propia]**

Ahora para verificar que Python ha sido instalado abrimos la consola y escribimos Python, la siguiente imagen muestra la respuesta afirmativa.

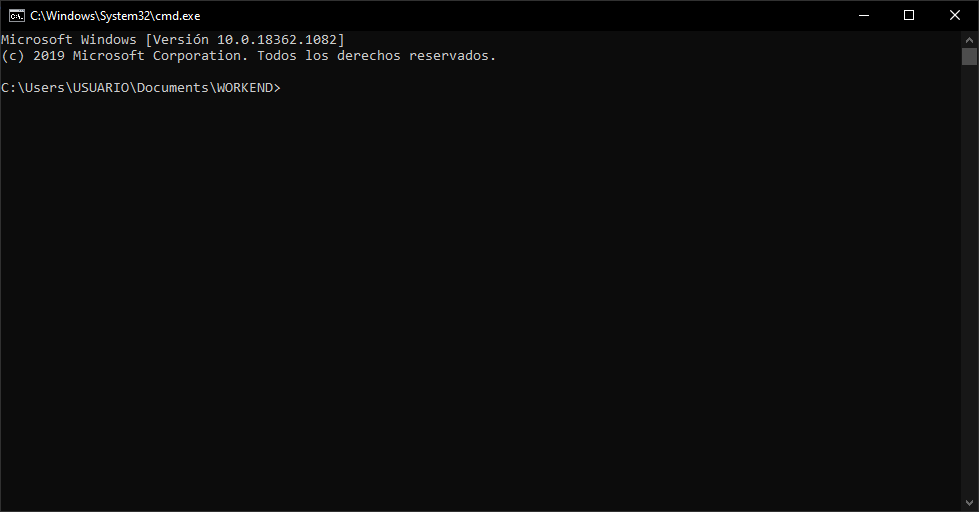
**Imagen 15:** Verificar instalación de Python en la computadora



**Fuente: [propia]**

Después de instalado creamos una carpeta y la abrimos, dentro de ella ejecutamos el cmd, en la siguiente imagen se muestra la ruta de la carpeta, cámbianos esta ruta por la palabra cmd y damos enter, automáticamente nos abre la consola dentro de la ruta C:\Users\USUARIO\Documents\WORKEND, como se muestra en la imagen 12.

**Imagen 16:** Iniciando consola para crear el entorno



**Fuente: [propia]**

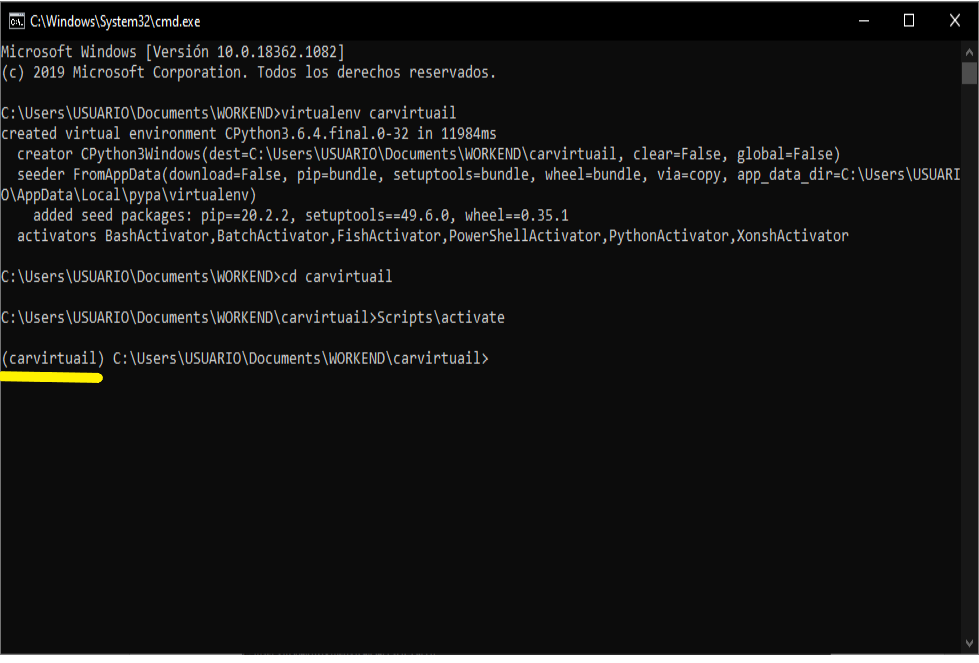
Luego creamos el entorno virtual con el fin de gestionar recursos como librerías, con el comando virtualenv y el nombre del entorno que deseamos poner en este caso que daría así: **virtualenv carvirtuaill**, damos enter y esperamos a que cree el entorno como se muestra en la siguiente imagen 16, donde en la consola muestra que ya ha terminado, verificamos yendo al directorio vemos que nos creó unas carpetas (Lib, Scripts), archivos(.gitignore, pyvenv.cfg) quiere decir que ya termino la creación del entorno. Luego ingresamos el comando **cd carro** para acceder al entorno, y para que todas las instalaciones que se hagan queden dentro de este activamos el entorno con el comando **Scripts\activate.** Que daría algo como se puede ver en la imagen 17.

**Imagen 17:** Creación del entorno virtual



**Fuente: [propia]**

**Imagen 18:** Activando el entorno

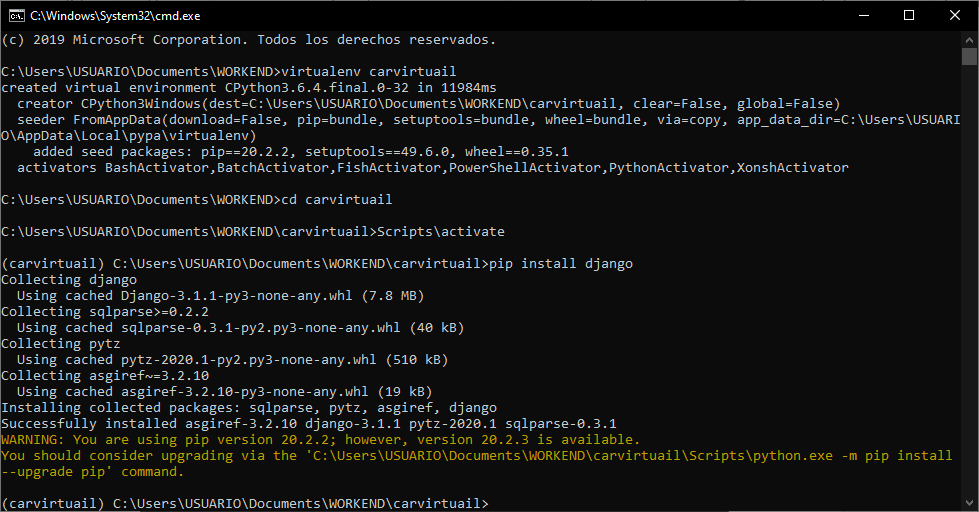


**Fuente: [propia]**

En la imagen 17 en la parte izquierda se logra ver el aval que ha sido activado, sino aparece el nombre del entorno al principio de la línea, quiere decir que no ha sido activado y puede haber algún error.

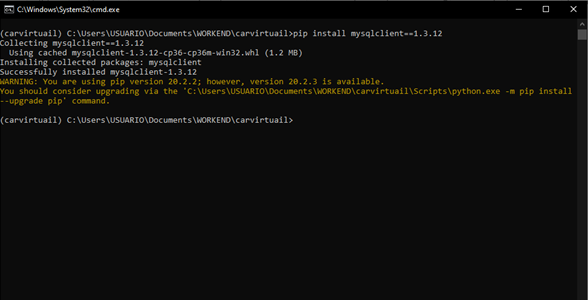
Continuamos con la instalación de los paquetes, Django, mysql. Para instalarlos se utiliza el comando **pip install django** automáticamente inicia la descarga e instala, en este caso dejamos que instale la última versión de Django. Los mismo hacemos para instalar la base de datos solo que le ponemos la versión que se quiere que instale, así **pip install mysqlclient==1.3.12**.

**Imagen 19:** Instalación de Django



**Fuente: [propia]**

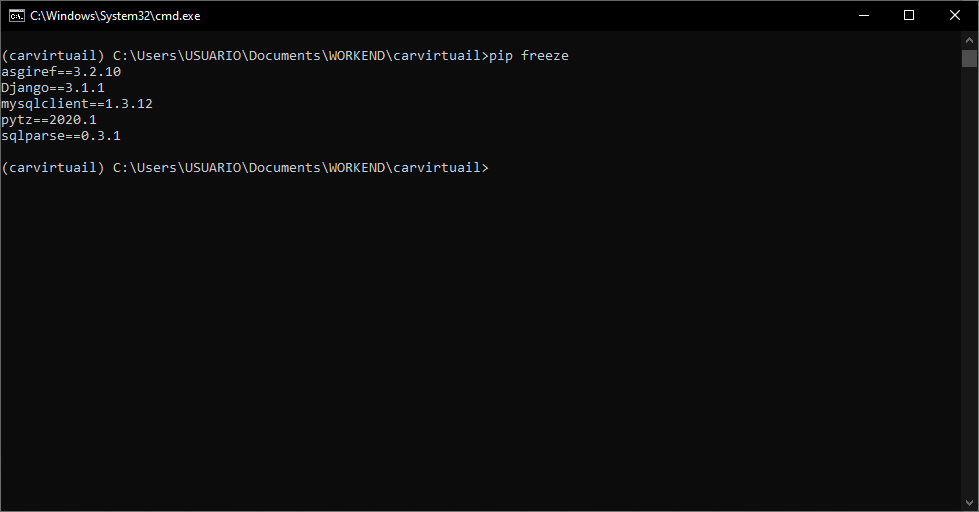
**Imagen 20:** Instalación de MYSQL



**Fuente: [propia]**

Para verificar que lo que se ha instalado en el entorno, utilizamos el comando **pip freeze** y nos muestra los paquetes que se han instalado, como se muestra en la imagen 21. Se ha instalado Django en la versión 3.1.1, y Mysql la versión que se le dio cuando se instaló, los demás paquetes los instala Django automáticamente.

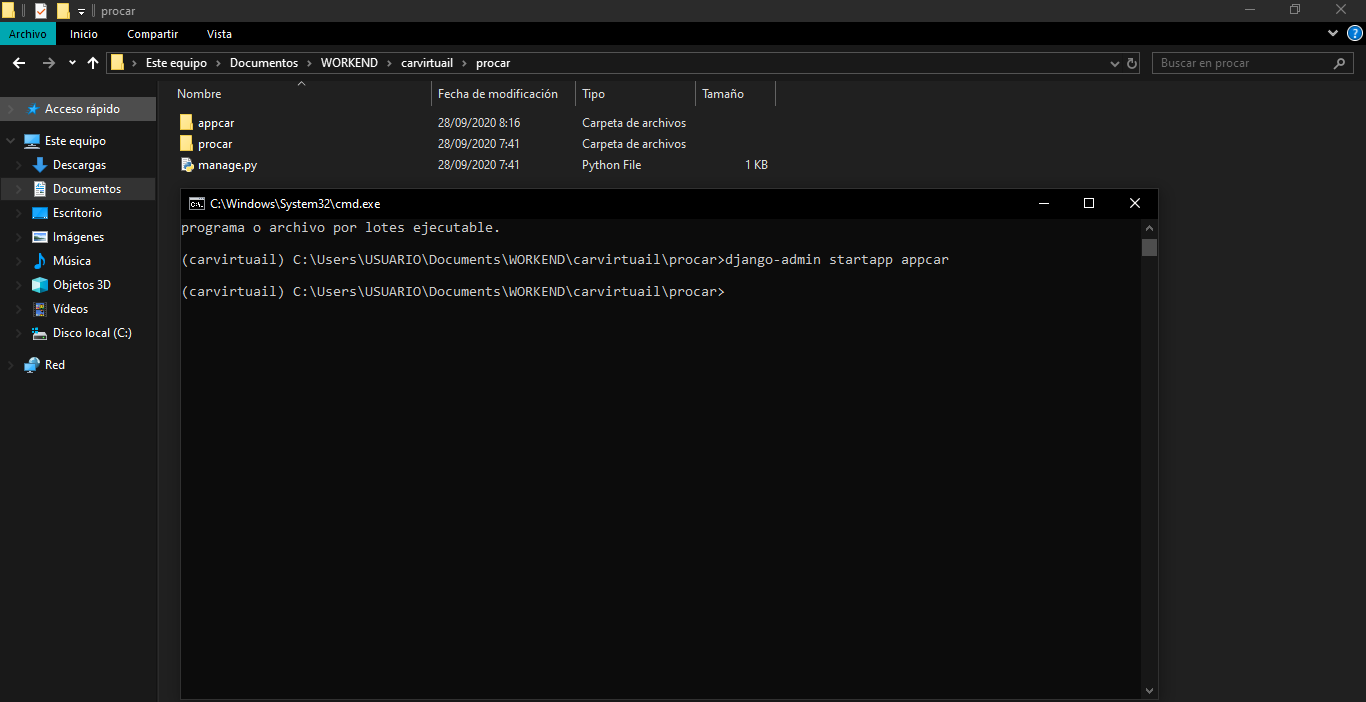
**Imagen 21:** Verificando instalación



**Fuente: [propia]**

Después de hacer la instalación de los paquetes creamos el proyecto con el comando **django-admin startproject,** después de la palabra startproject se escribe el nombre del proyecto entonces que daría asi: **django-admin startproject procar**, escribimos el comando **cd procar,** se crea la aplicación de la siguiente forma **django-admin startapp** agregamos el nombre de la aplicación y que asi: **django-admin startapp** **appcar.** En la siguiente imagen se muestra que la consola no muestra nada, pero si va a la dirección de donde se está ejecutando la consola podemos ver que ha creado las carpetas con los nombres que se les dio.

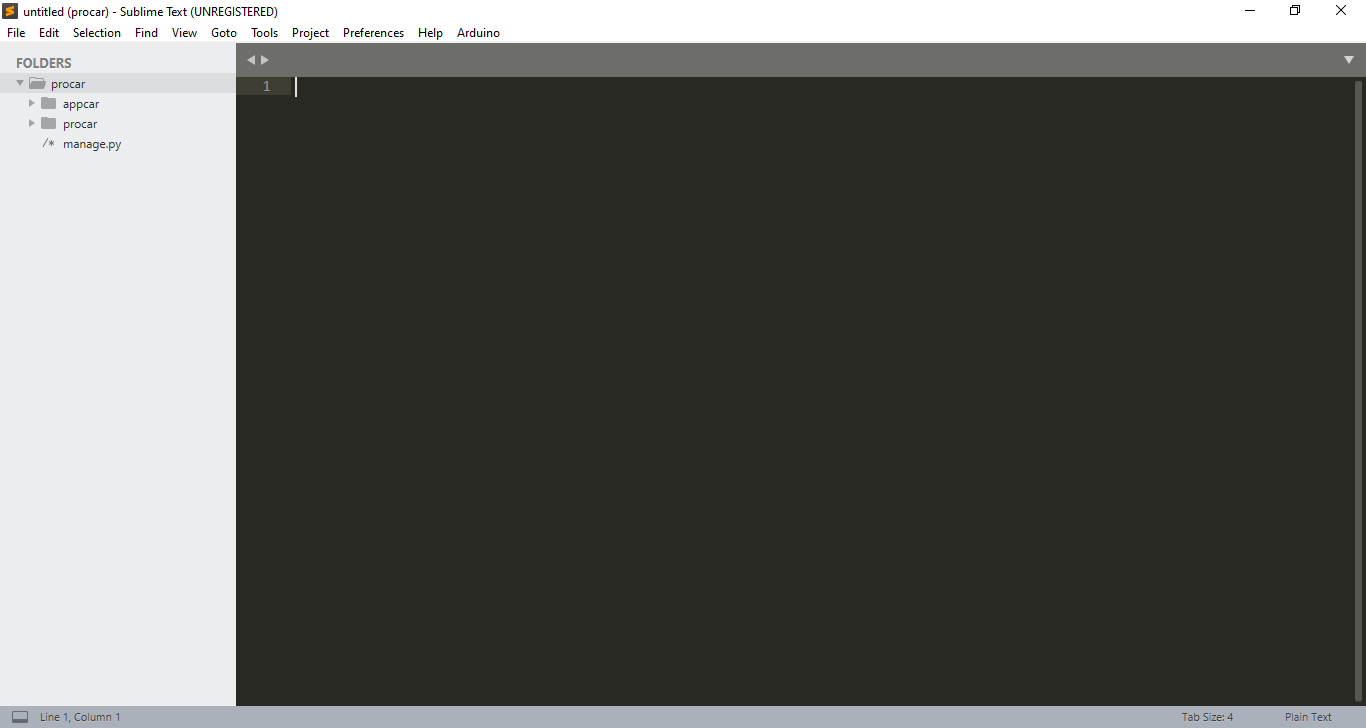
**Imagen 22:** Consola y explorador de archivos



**Fuente: [propia]**

Abrimos el proyecto con un editor de texto para hacer la configuración por que como viene no funciona, nosotros utilizamos Sublime text para editar. En la siguiente imagen se observa la estructura del proyecto en Django. Luego seguimos los pasos de la imagen 24 que son la configuración de las plantillas, urls y el settings. Si se detalla un poco todos los archivos están en un formato py que quiere decir que estamos trabajando sobre Python.

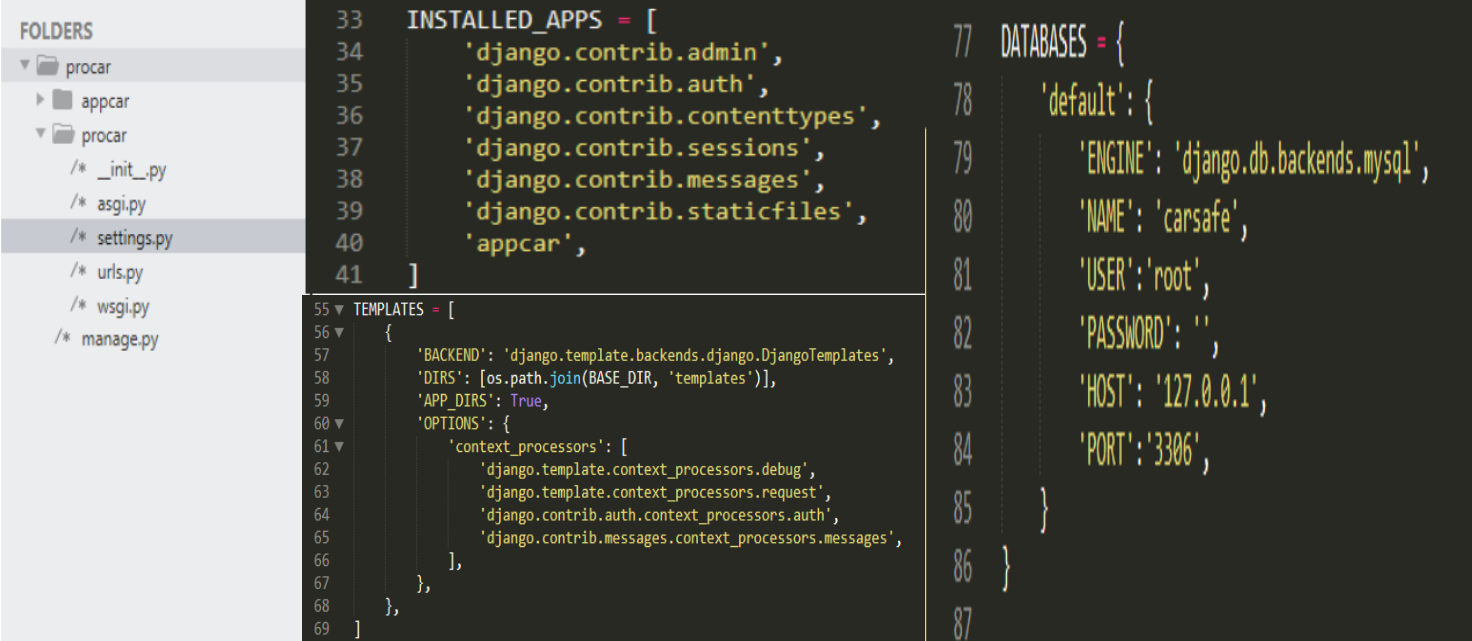
**Imagen 23:** Proyecto en Django



**Fuente: [propia]**

Abrimos la carpeta con el nombre del proyecto, nos muestra unos archivos, buscamos primero el que dice settings.py vamos donde se declaran todas las aplicaciones y registramos la que se creó al inicio, bajamos hasta TEMPLATES y pegamos esta línea ([os.path.join(BASE\_DIR, 'templates')]) donde esta DIRS que cuando se crea el proyecto viene vacía. A continuación, configuramos la base de datos, este proyecto trabaja con Mysql entonces llenamos los campos necesarios para trabajar en Mysql como se muestra en la imagen 24.

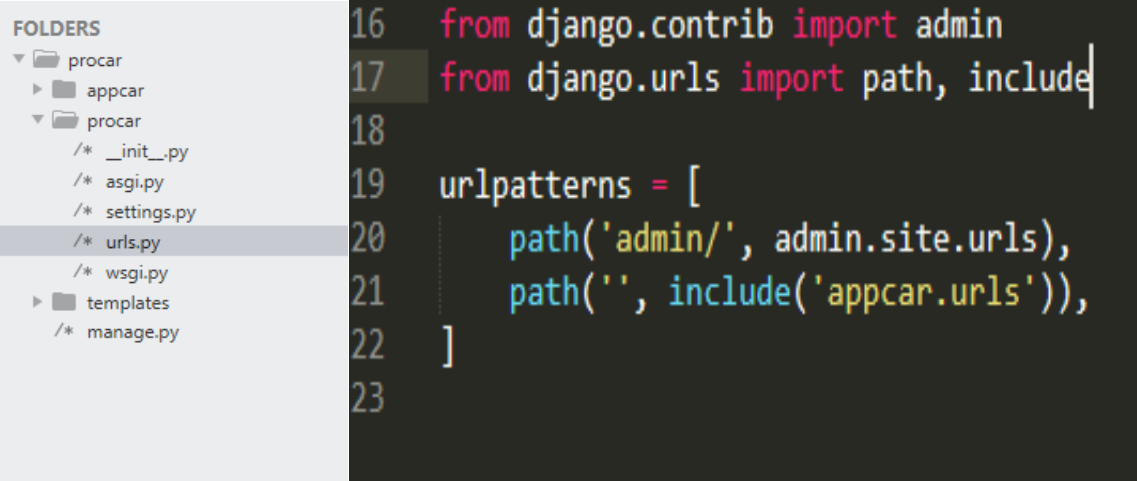
**Imagen 24:**Configuración en el setting.py



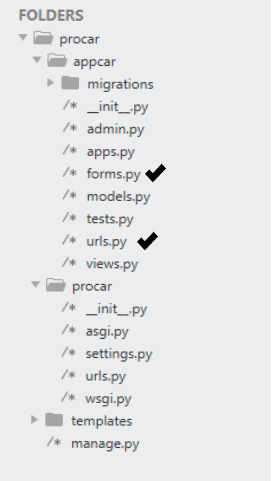
**Fuente: [propia]**

En los pasos siguientes creamos una carpeta con el nombre **templates,** es la que contendrá todos los archivos **html.** Luego navegamos hasta el archivo url.py del proyecto, se incluye las url de la aplicación, se debe crear el archivo urls.py, forms.py en la carpeta de la aplicación.

**Imagen 25:** Configuración de la Url

**Fuente: [propia]**

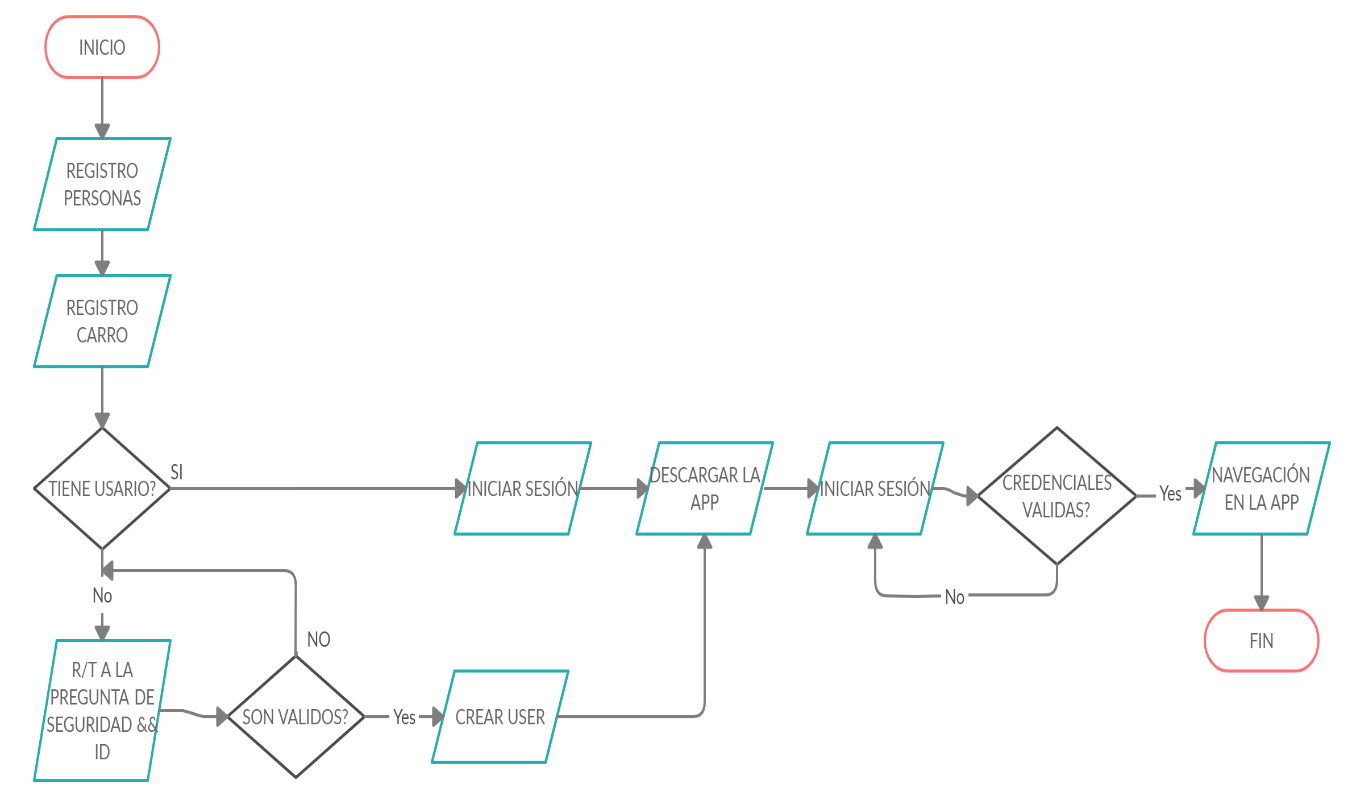
**Imagen 26:** Creación de los archivos urls.py, forms.py



**Fuente: [propia]**

En la siguiente imagen se muestra el diagrama de flujo donde expone el algoritmo de funcionamiento de la aplicación web y móvil.

**Imagen 27:** Diagrama de flujo de la aplicación web y móvil



**Fuente: [propia]**

Compresión del diagrama de flujo de la imagen 25: lo primero es el registro de persona y carro está dado por un recepcionista que pide al usuario dar los datos personales y del vehículo tales como (nombre, apellido, identificación, teléfono, dirección, además se le hace una pregunta de seguridad, y se crea un campo para la respuesta), para el carro seria (modelo, ciudad, marca y placa) para hacer una inserción en base de datos, cuando se termina el proceso de los anteriores registros se le anuncia al usuario que ingrese a la página y haga el proceso de creación del usuario, el usuario debe dirigirse a la parte de creación de usuarios luego ha de ingresar la respuesta a la pregunta que se le hizo en el registro y la identificación, el sistema valida de que se encuentre registrado estos datos, en el caso de que no estén el sistema vuelve a pedirlos, en caso contrario el sistema automáticamente inicia sesión y le permite descargar la aplicación para ser instalada en el teléfono Smartphone. Luego de instalar la app automáticamente abre una venta para el inicio de sesión donde el sistema hace una petición a la aplicación web y está a la base de datos, para validar de que este usuario existe, y posterior mente la respuesta es verdadera se permite la navegación total en la app. Cabe aclarar que el usuario solo recibirá las advertencias de lo que suceda en el vehículo solo si esta la sesión activa, el funcionamiento de la app móvil se expone más claramente en la sección 4.5. creación de la app.

## 4.4. IMPLEMENTACIÓN DE TECNOLOGÍAS

En esta división se expone las tecnologías utilizadas para él envió de notificaciones a dispositivos móviles, también las necesarias para el despliegue de la aplicación web.

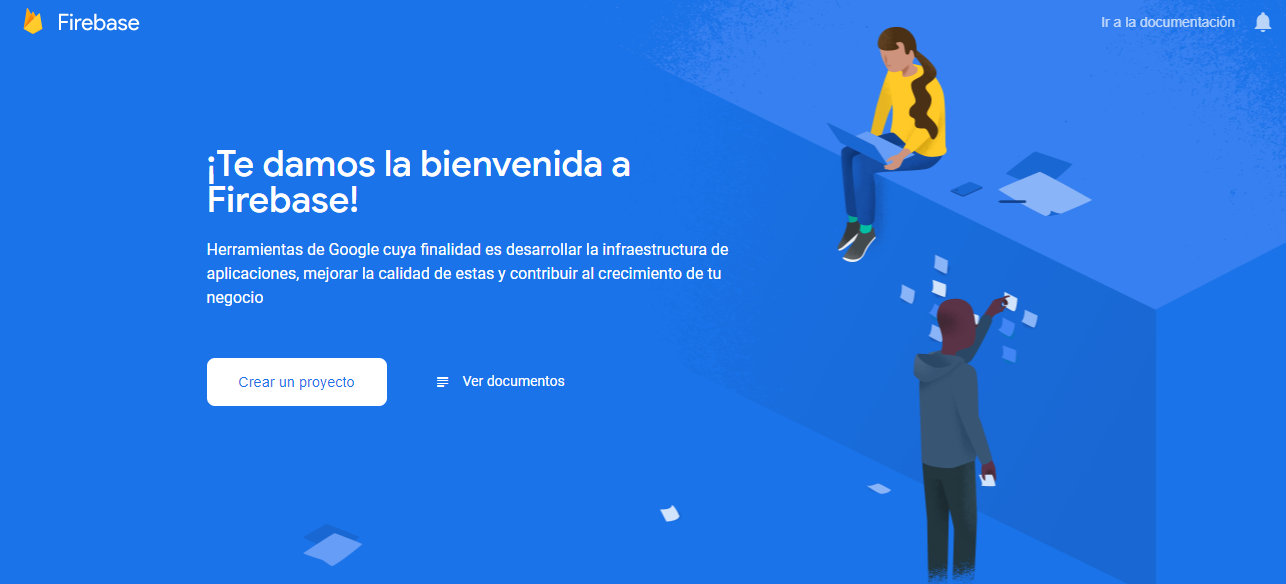
### 4.4.1. ENVIÓ DE NOTIFICACIONES

La tecnología que se utiliza para hacer él envió de notificaciones desde un servidor a un dispositivo Android, es Firebase Cloud Messaging (FCM), es gratuita con una conexión menor a 100 dispositivos después de este número hay que pagar, FCM pertenece a Google. Registra bastante documentación.

La implementación de esta tecnología es así: primero partiendo de que Sigfox envía una solicitud con la información de lo que sucede en el vehículo al servidor externo, ahí se traduce la información para luego ser enviada al dispositivo Android por medio de FCM, para que sea un éxito él envió de las notificaciones hay que instalar y configurar FCM en la parte del servidor, también configurar la app móvil para que reciba las notificaciones.

Nos registramos con un correo de extensión Gmail, llenamos los datos que pida. Después de que se hace el registro nos recibe una interface como la de la imagen 26, donde lo primero que pide es que creemos un proyecto, pulsamos en crear proyecto llenamos el campo: nombre del proyecto, damos en crear. La imagen 27 muestra que ha finalizado con éxito la creación del proyecto. También muestra el plan en el que se ha creado el proyecto en este caso es el plan Spark que es el gratuito, podemos verificar el plan en la bandeja de opciones en la parte izquierda o al frente del nombre del proyecto. Después nos pide que agreguemos una aplicación, este paso está en la parte de creación de la app móvil.

**Imagen 28:** Interface de Firebase



**Fuente: [propia]**

**Imagen 29:** Proyecto creado en Firebase



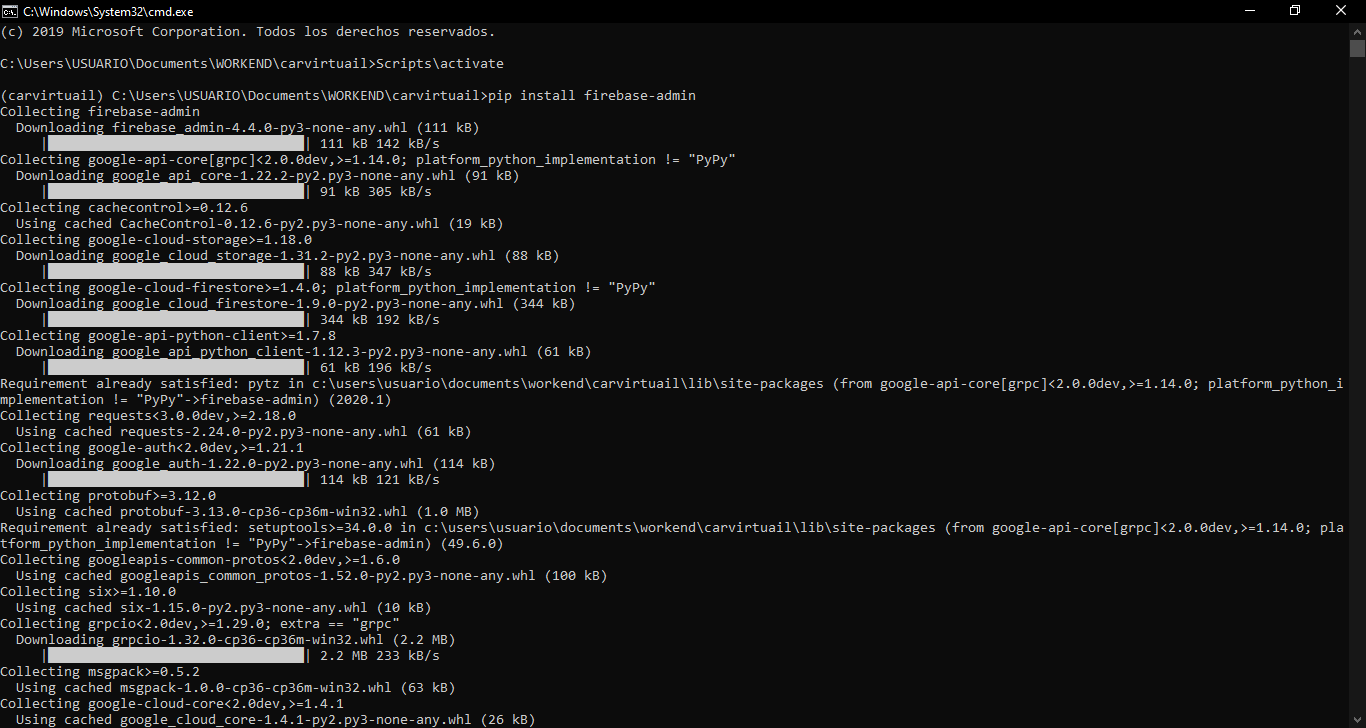
**Fuente: [propia]**

El paso siguiente es configurar el entorno del servidor para que funcione con FCM, según la documentación de Google FCM se puede enviar directamente un mensaje de notificación desde la parte del servidor, creada con las opciones de FCM o simplemente enviar un mensaje de datos con la información sin ninguna prioridad en formato Json [45].

Primero abrimos la consola en el computador, navegamos hasta el entorno del proyecto, lo activamos. Procedemos a instalar FCM en Python de la siguiente forma **pip install firebase-admin,** esperamos a que termine la instalación, escribimos **pip freeze** para comprobar la instalación, en la consola se ve como la imagen 29.

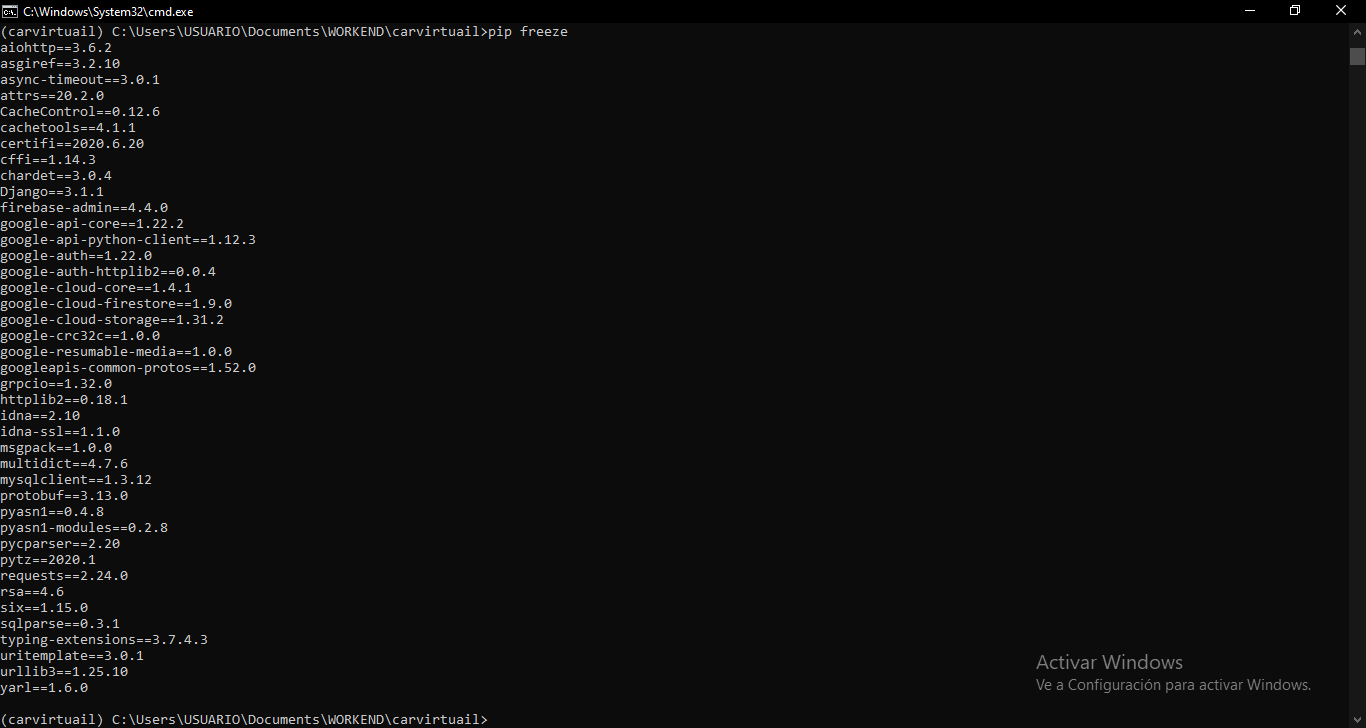
**Imagen 28: instalación de Firebase**

**Imagen 30:** Instalación de Firebase en la aplicación web



**Fuente: [propia]**

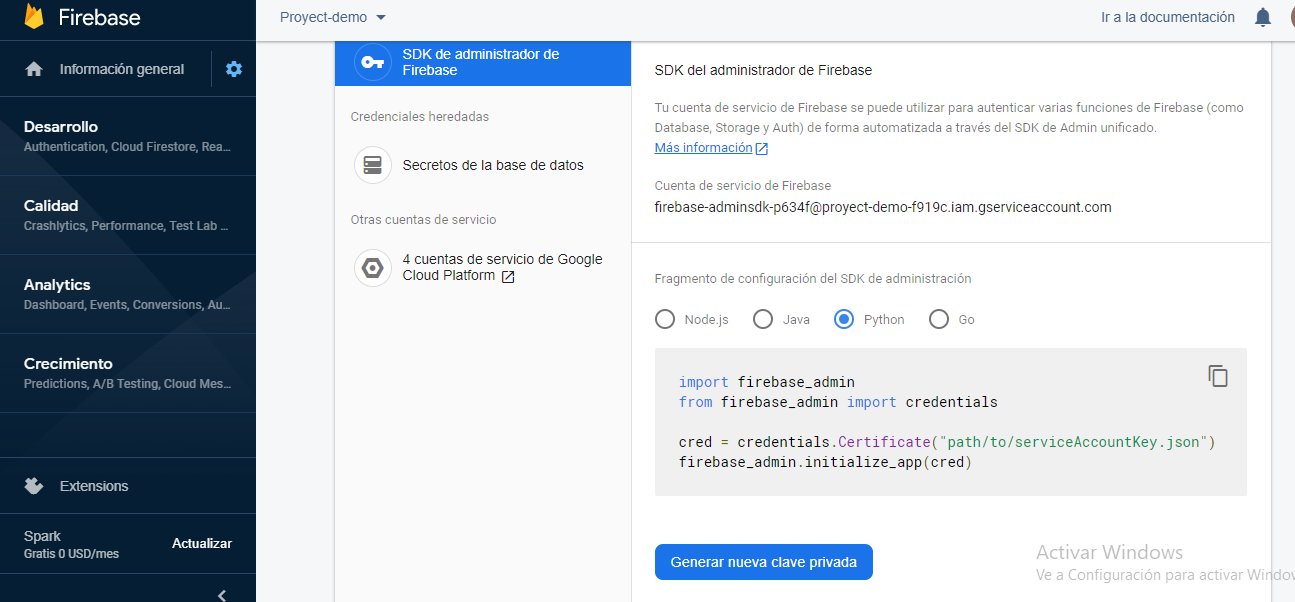
**Imagen 31:** Comprobación de la instalación de Firebase



**Fuente: [propia]**

El paso siguiente es configurar el proyecto con FCM, para ello nos dirigimos a la consola de Firebase, navegamos en la opción de **información general,** siguiendo hasta **configuración del proyecto** y luego **cuentas de servicio** como se muestra en la imagen 30. Ahí Google Firebase muestra como debe ser la configuración para cada lengua de programación en este caso se escoge Python y generamos la una clave privada, automáticamente Firebase nos la genera en un archivo Json descargable, este archivo nos permite el acceso al proyecto de Firebase desde el proyecto de Python, el cual se recomienda guardar en un sitio seguro, con la clave privada que trae otorga acceso a los servicios de Firebase del proyecto.

**Imagen 32:** Configuración de Firebase en Python



**Fuente: [propia]**

#### 4.4.1.1 CONFIGURACION DE PARTE DEL SERVIDOR PARA ENVIO DE UN MENSAJE DE NOTIFICACIÓN

La documentación de Firebase [45] nos ofrece la siguiente configuración para él envió de un mensaje de notificación. Donde se inicia con la palabra mensaje en ingles luego se le pasa un identificador (token) para saber a qué usuario mandar el mensaje. El token está dado en el momento que el usuario inicia sesión en la app móvil, luego se agrega la palabra clave (notification) dándole un título y un cuerpo a la notificación. En la imagen 31 se evidencia que los mensajes también pueden ser enviados por categoría de tipo de aplicación para Android, iOS o webs. Firebase tiene la capacidad de darles un tiempo de vida (TTL) al mensaje enviado.

{  
  "message":{  
    "token":"bk3RNwTe3H0:CI2k\_HHwgIpoDKCIZvvDMExUdFQ3P1...",  
    "notification":{  
      "title":"CARSAFE",  
      "body":"carro en movimiento"  
  }  
  }  
}

**Imagen 33:** Mensaje para cada aplicación



**Fuente:** [45]

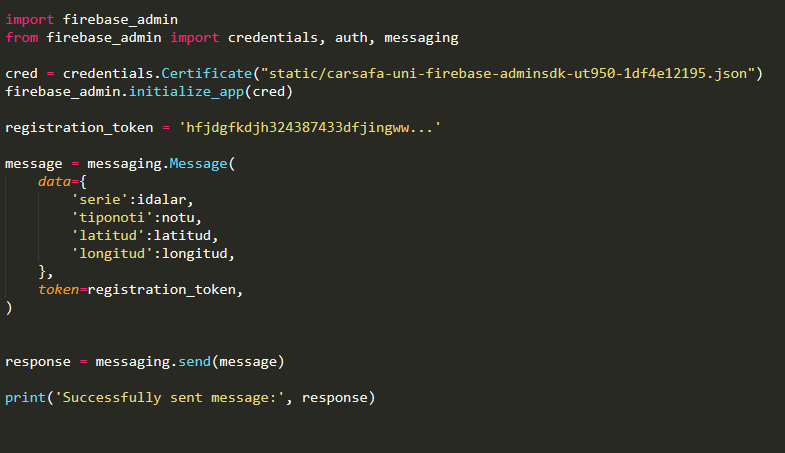
#### 4.5.1.2. CONFIGURACION DE PARTE DEL SERVIDOR PARA ENVIO DE UN MENSAJE DE DATOS

El siguiente código [45] es un mensaje de datos en formato JSON donde la información está contenida por la clave DATA. En el cual se requiere la identificación (token) para saber a qué usuario enviar.

{  
  "message":{  
    "token":"bk3RNhgdjkfjbhbIHKSFOSNKNN548...",  
    "data":{  
      "device" : "4ff34",  
      "latitud" : "2.45883",  
      "longitud" : "-75.54684"  
    }  
  }  
}

En la siguiente imagen se muestra la configuración completa para él envió de un mensaje de datos desde Python.

**Imagen 34:** Solicitud para un envió de mensaje desde Python



**Fuente: [propia]**

Para él envió de la información obtenida de los sensores a través de Sigfox, hasta el servidor externo este proyecto opto por enviar la información en un mensaje de datos, ya que en la creación de la notificación hay que tener en cuenta la versión del sistema operativo del dispositivo [46].

### 4.4.2. GITHUB

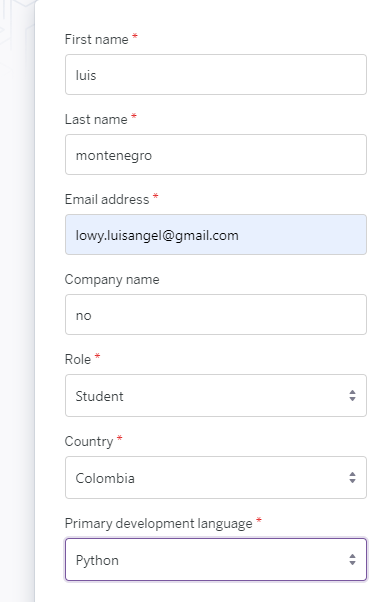
Es un repositorio que nos permite almacenar los códigos y trabajar en grupo, en este proyecto se usa para que cada vez que hagamos un cambio en el proyecto solo desplegarlo una vez, y el servidor automáticamente sincroniza los cambios. Para manejar esta plataforma de desarrollo sigue los pasos a continuación:

Dar click en la url: <https://github.com/>, llenar los campos y listo, tienes un cuenta en GitHub, luego buscamos la opción: repositorios, entramos y damos en crear un repositorio nuevo. Le damos un nombre, seleccionamos la opción de privado o público, la opción privada solo pueden editar, descargar, y ver las personas que estén en colaborativo, es caso contrario es la opción publica ya que todos tiene acceso a él. Por ultimo dar en crear repositorio, listo ya tienes un repositorio listo para subir los archivos que desees.

### 4.4.3. HEROKU

Para el uso de esta plataforma vamos a la url <https://www.heroku.com/>, entrar en la opción registrarse gratis, a continuación muestra una interface donde se llenan todos los campos como se muestra en la imagen 33.

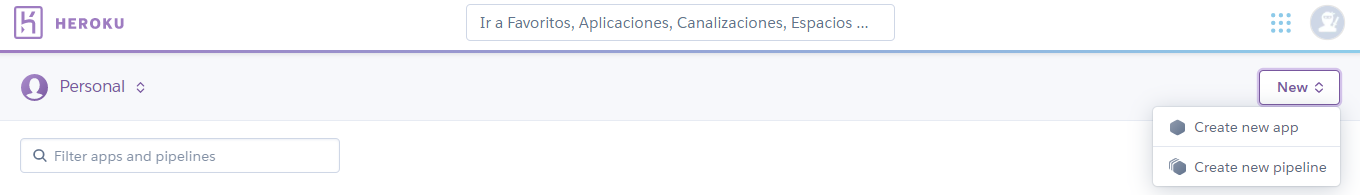
**Imagen 35:** Registro en Heroku



**Fuente: [propia]**

Nos aparece una interface y damos en nuevo, luego se toma la opción de crear nueva app como se ve en la siguiente imagen.

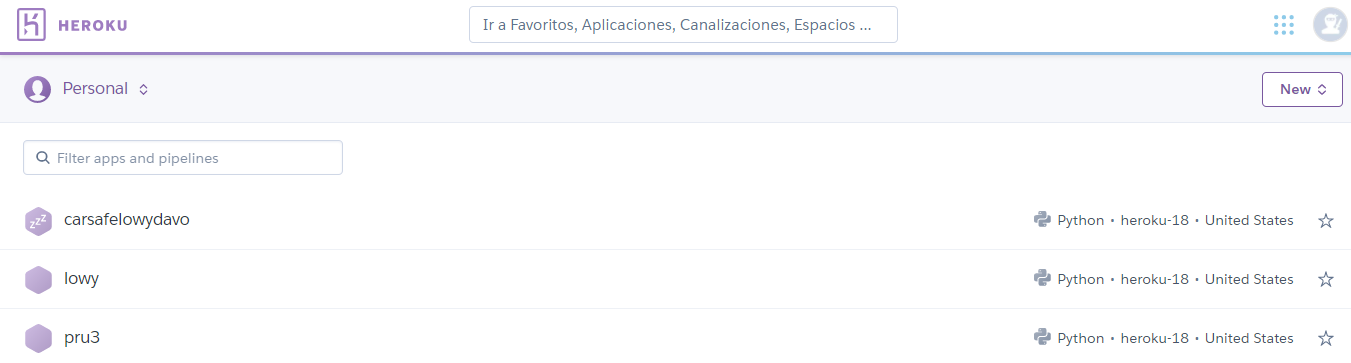
**Imagen 36:** Creando una aplicación en Heroku



**Fuente: [propia]**

Damos un nombre a la aplicación y elegimos un país (Estados Unidos), la demos en crear y listo. Aparece el nombre de la aplicación, país, y lenguaje de programación, como se muestra en la imagen 35.

**Imagen 37:** Aplicación en Heroku

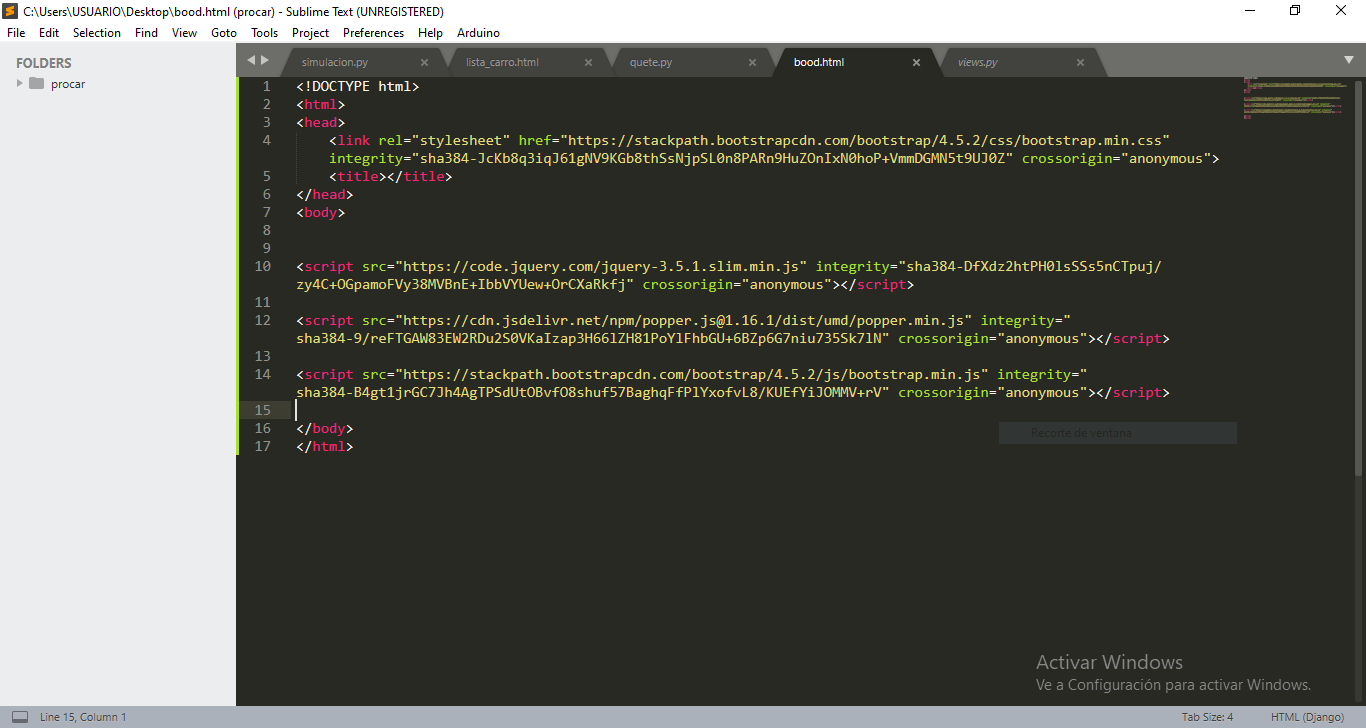


**Fuente: [propia]**

### 4.4.4. BOOTSTRAP

Es un Framework para el diseño web, del lado de Front end. Su uso en los diseños es de fácil implementación, solo basta con agregar las siguientes líneas de código como se explica en la imagen 36 y que da listo para usarse. Les dejo la url directo a la documentación: <https://getbootstrap.com/docs/4.5/getting-started/introduction/>.

**Imagen 38:** Agregando Bootstrap al proyecto



**Fuente: [propia]**

### 4.4.5. JAVASCRIPT

Este lenguaje se utiliza para el lado del Front End, para crear efectos, animaciones, dando una respuesta a ciertos eventos provocados por él usuario, por ejemplo, al pulsar un botón, llenado de formularios, etc. Este lenguaje también tiene una extensión llamada AJAX que permite el intercambio de información con el servidor sin necesidad de recargar la página. JavaScript también tiene Framework que permite programar el lado del servidor o Back-end [47].

## 4.5. DESARROLLO DE APP MÓVIL

## 4.6. PUBLICACIÓN DE LA APLICACIÓN WEB EN INTERNET

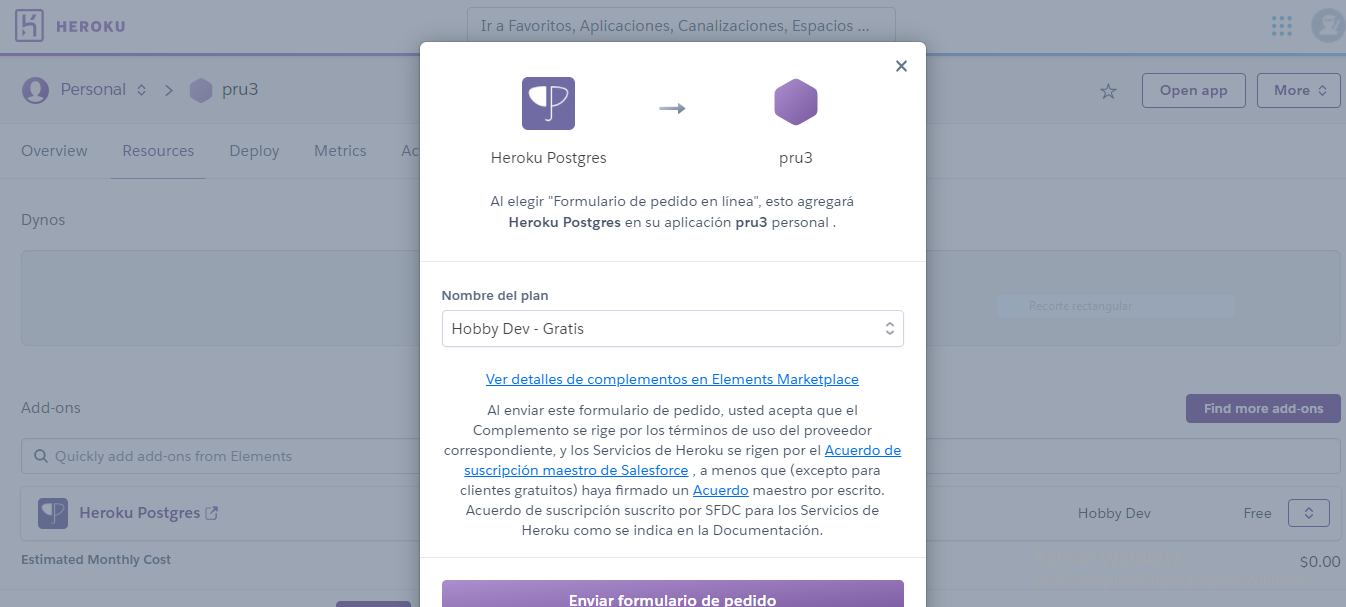
Los anteriores pasos que se realizaron en cuanto configuración de proyecto y base de datos en Django en la actividad 4.3 son solo para trabajo local ahora nos centraremos en la publicación de la aplicación web en internet o como Django lo llama puesta en producción [49].

Lo primero que hacemos es seleccionar el servidor donde queremos poner en producción nuestra aplicación web, en el capítulo 2 en la sección 2.2.7 se habla sobre un servidor llamado Heroku. se escogió este servidor primero porque los desarrolladores de este proyecto no cuentan con un servidor propio, segundo este servidor es gratuito y lo más importante que ya está configurado para hacer la comunicación con la aplicación web hecha en Python.

### 4.6.1. CONFIGURACIÓ DE LA BASE DE DATOS EN POSTGRES

Lo primero que debemos hacer dirigirnos a <https://www.heroku.com/> e iniciar sesión, abrimos la aplicación que se creó en la sección 4.4.3, nos dirigimos a la opción **Resources,** vamos donde dice **Add-ons** y ponemos **Heroku Postgres**, en este paso se está configurando la base de datos con la que vamos a trabajar. Como servidor Heroku no trabaja con Mysql, entonces la opción es trabajar con la base de datos en Postgres. Abre una ventana como se muestra en la siguiente imagen.

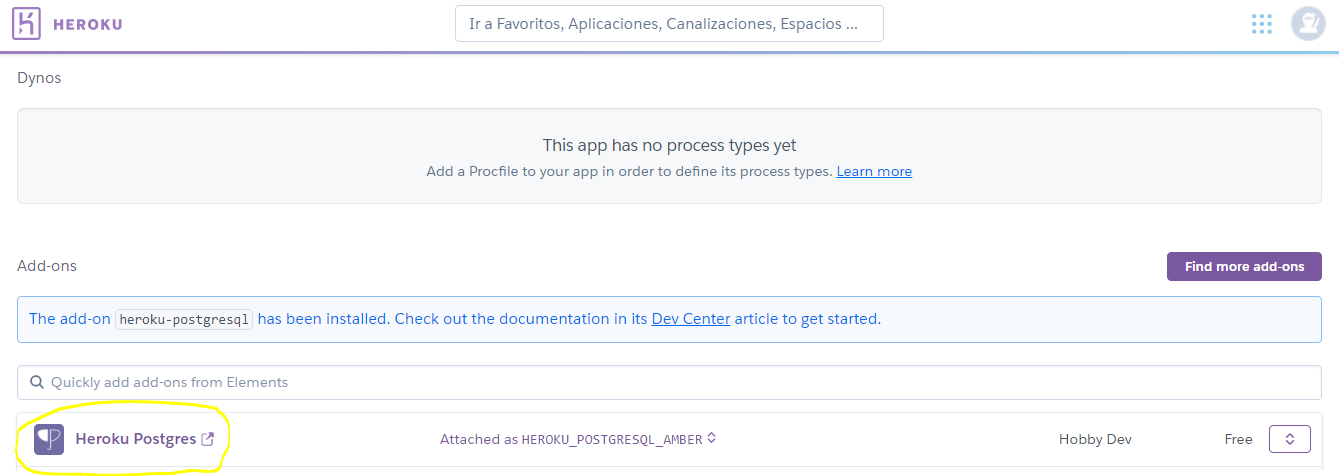
**Imagen 39:** Configuración de base de datos



**Fuente: [propia]**

Después seleccionamos el nombre del plan **Hobby Dev – Gratis**  damos click en **enviar formulario de pedido,** y listo se creó una base de datos en Postgres. En la siguiente imagen se puede confirmar que sí creo la base de datos. El paso siguiente es dar click en el nombre de la base de datos (**Heroku Postgres**) para obtener las credenciales.

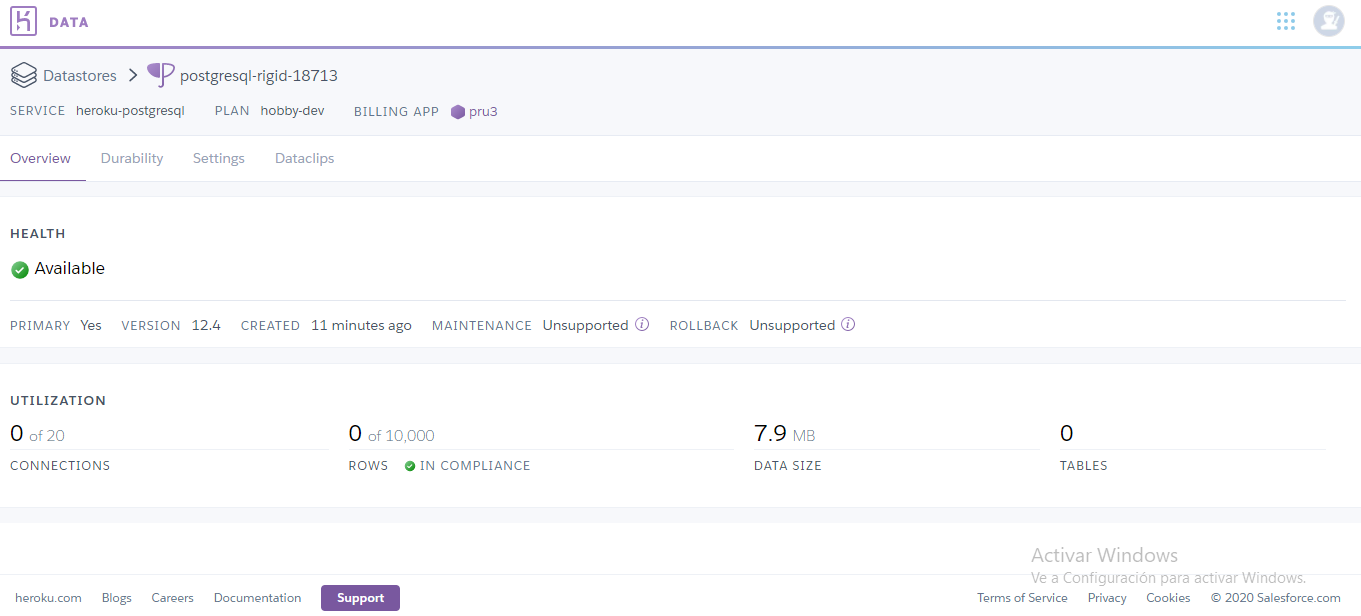
**Imagen 40:** Base de datos creada en Postgres



**Fuente: [propia]**

Abre una ventana como se ve en la siguiente imagen.

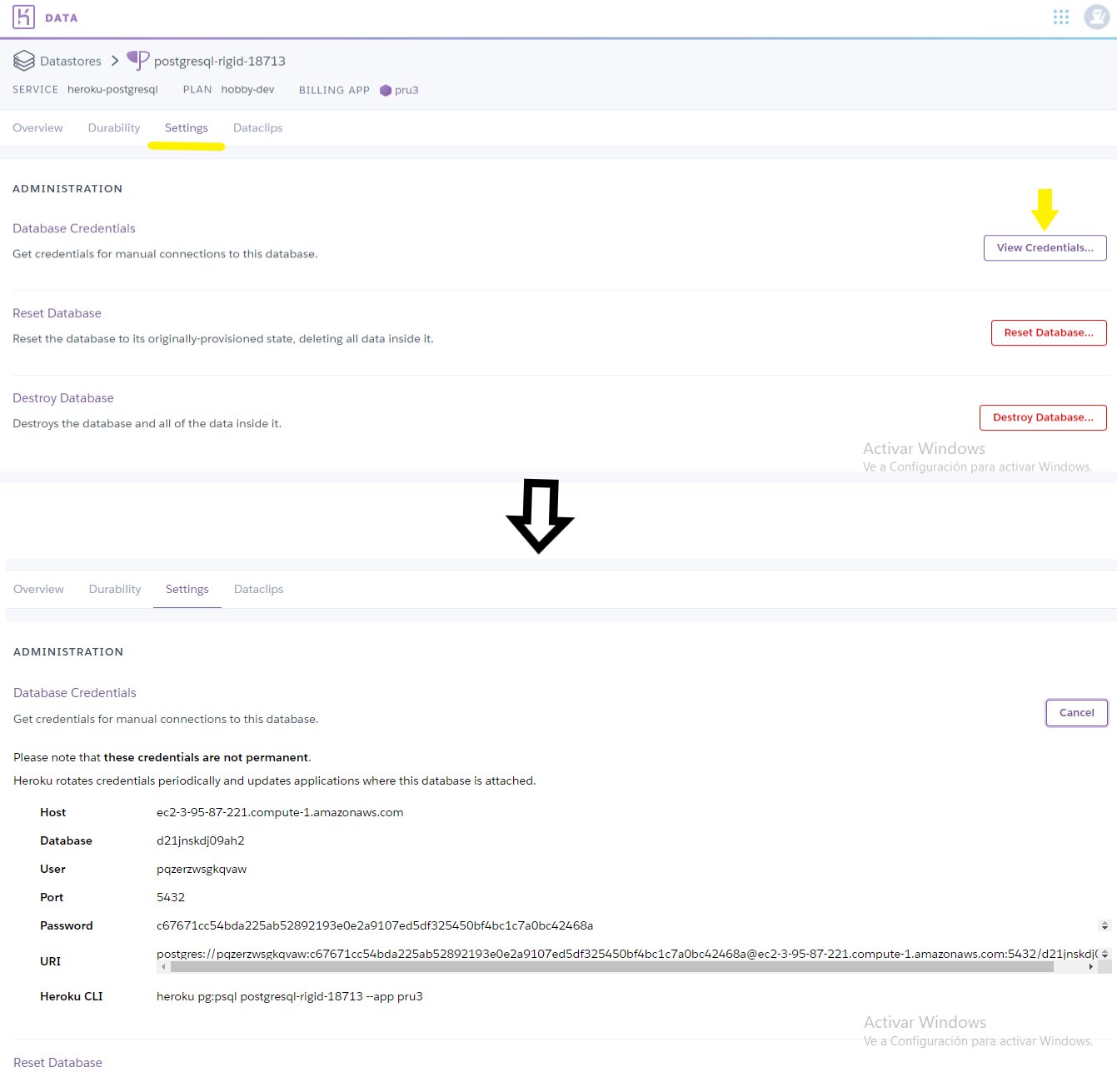
**Imagen 41:** Ventana de datos de Postgres



**Fuente: [propia]**

Ahora nos dirigimos a **settings y a View Credentials,** automáticamente deja ver las credenciales de la base de datos, como se muestra en la siguiente imagen.

**Imagen 42:** Credenciales de la base de datos en Postgres

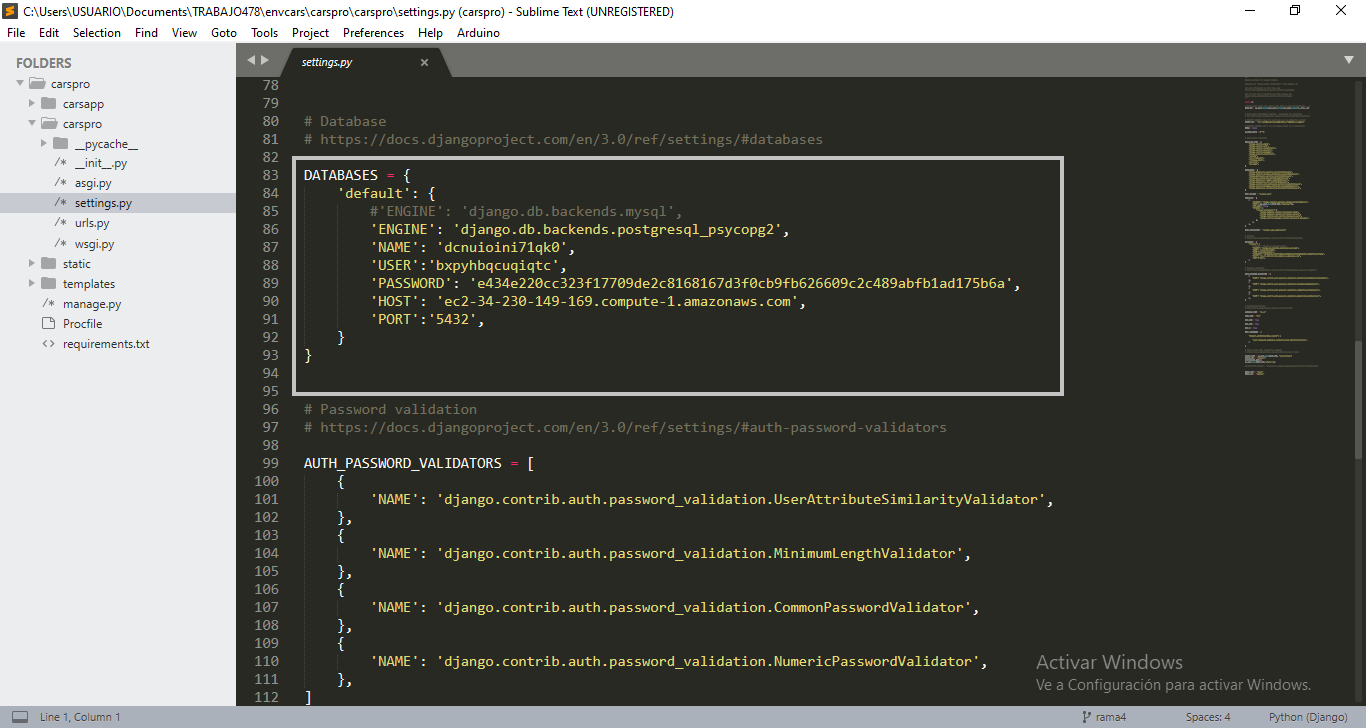


**Fuente: [propia]**

### 4.6.2. CONFIGURACIÓN DE BASE DE DATOS EN PYTHON

El paso siguiente es dirigirnos al settings.py de nuestro proyecto y poner las credenciales, la configuración que se hizo de base de datos en el apartado 4.4, lo cambiamos por Postgres y ponemos las credenciales. Como se muestra en la siguiente imagen.

**Imagen 43:** Configuración de la base de datos en el proyecto Python

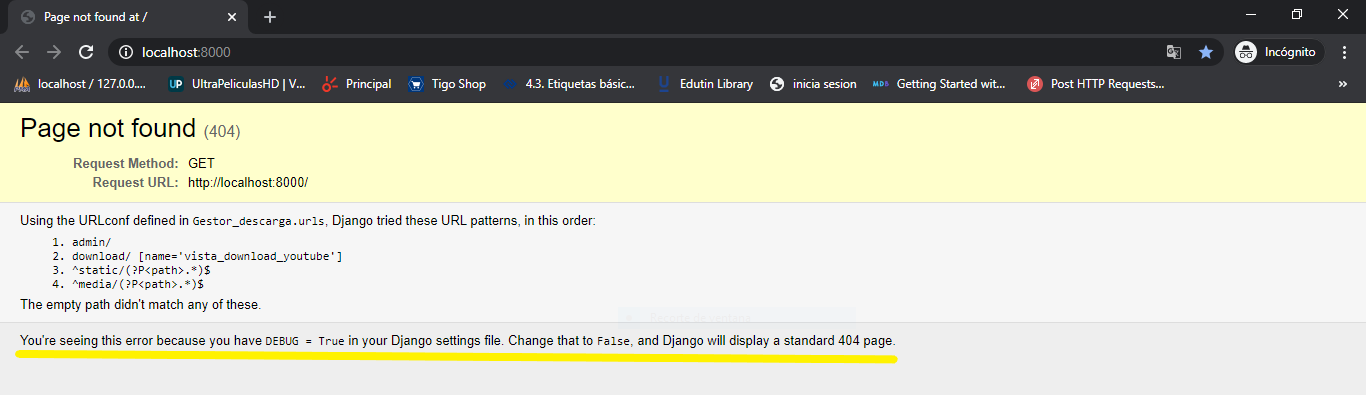


**Fuente: [propia]**

### 4.6.3. CONFIGURACIÓN DEL DEBUG Y GENERACIÓN DE ARCHIVOS

Ahora pasamos la opción **debug** a **false,** esta opción lo que hace es activar/desactivar el modo depuración, la documentación de Django recomienda poner en **false** cuando la aplicación se pone en producción para que cuando haya un error no muestre las rutas de archivos, las opciones de configuración del servidor, esto lo hace más vulnerable a los atacantes [50]. En la siguiente imagen se muestra un ejemplo del modo **debug** en **true.**

**Imagen 44:** Debug en True

****

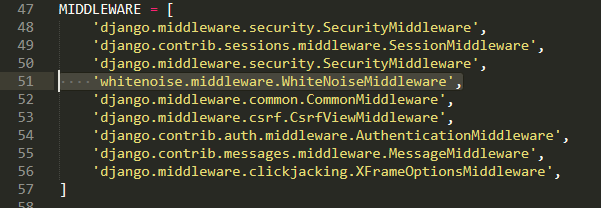
**Fuente: [propia]**

Generamos un archivo con extensión txt llamado **requirements.txt,** la forma para crearlo es poner en consola este comando: **pip freeze => requirements.txt,** para que Heroku sepa que es lo que debe instalar para la puesta en producción de la aplicación web. También hay que crear un archivo sin extensión llamado **Procfile,** este es el encargado de decirle a Heroku los comando que debe ejecutar al inicio de la aplicación.

### 4.6.4. CONFIGURACIÓN DE ARCHIVOS ESTÁTICOS

Los archivosestáticos también son una parte importante del proyecto, ya que son los que contienen los archivos de diseño web, imágenes, etc. Para que la aplicación web los ejecute hay que hacer una configuración, la cual es primero instalar la librería **whitenoise**, posteriormente se hace la configuración que se muestra en las siguientes imágenes. Primero se agrega la línea resaltada en **MIDDLEWARE.**

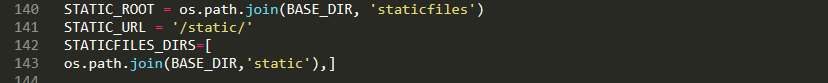
**Imagen 45:** Configuración de los archivos estáticos con Whitenoise



**Fuente: [propia]**

En consecuencia, se ponen unas líneas de código para terminar de configurar los archivos estáticos, así pues, se ven en la siguiente imagen. Con esto ser termina la configuración de parte del proyecto en Django para poner la aplicación en producción, las configuraciones posteriores ya se hacen en la aplicación de Heroku.

**Imagen 46:** Configuración de los archivos estáticos

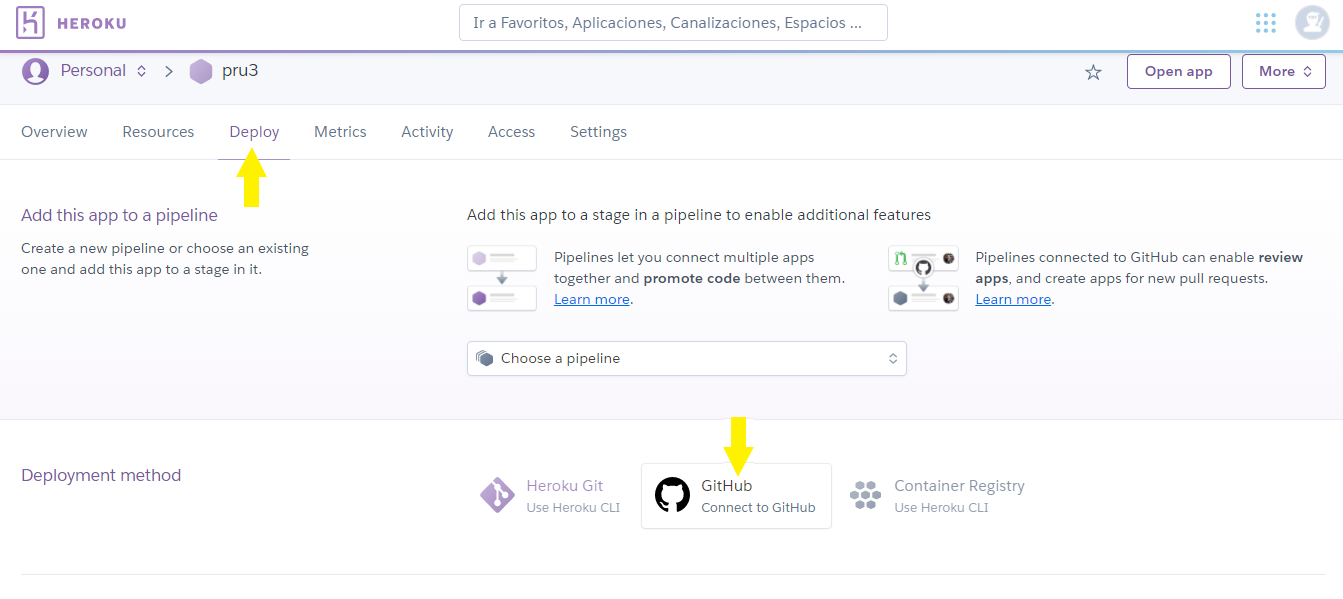


**Fuente: [propia]**

### 4.6.5. PUESTA EN PRODUCCIÓN DE LA APLICACIÓN WEB

Estas son las ultimas configuraciones para poner en producción la aplicación web, nos dirigimos a la aplicación que se creó en Heroku, seguimos los pasos de las siguientes imágenes.

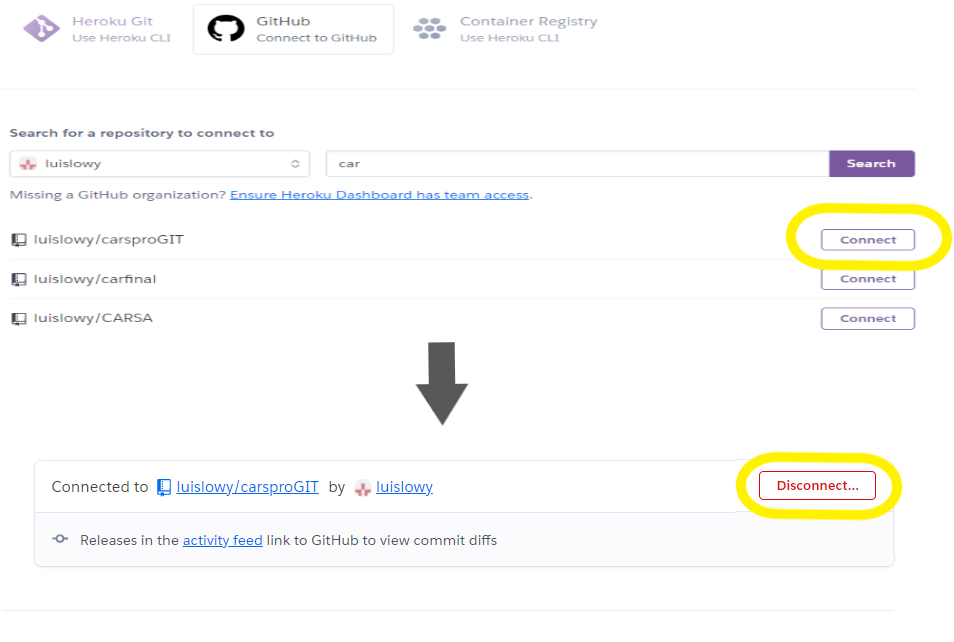
**Imagen 47:** Configuración de la aplicación de Heroku, paso1



**Fuente: [propia]**

Buscamos la opción que dice **deploy,** damos click, muestra una pantalla como se ve en la anterior imagen, ahora se elige la opción de conectar con el repositorio de Github. Llenamos los datos para el inicio de sesión, automáticamente Heroku trae el nombre del usuario del repositorio, a continuación, elegimos el nombre del repositorio donde está el proyecto de Python, como se muestra en la próxima imagen, elegimos conectar. Después nos confirma que se ha conectado al repositorio.

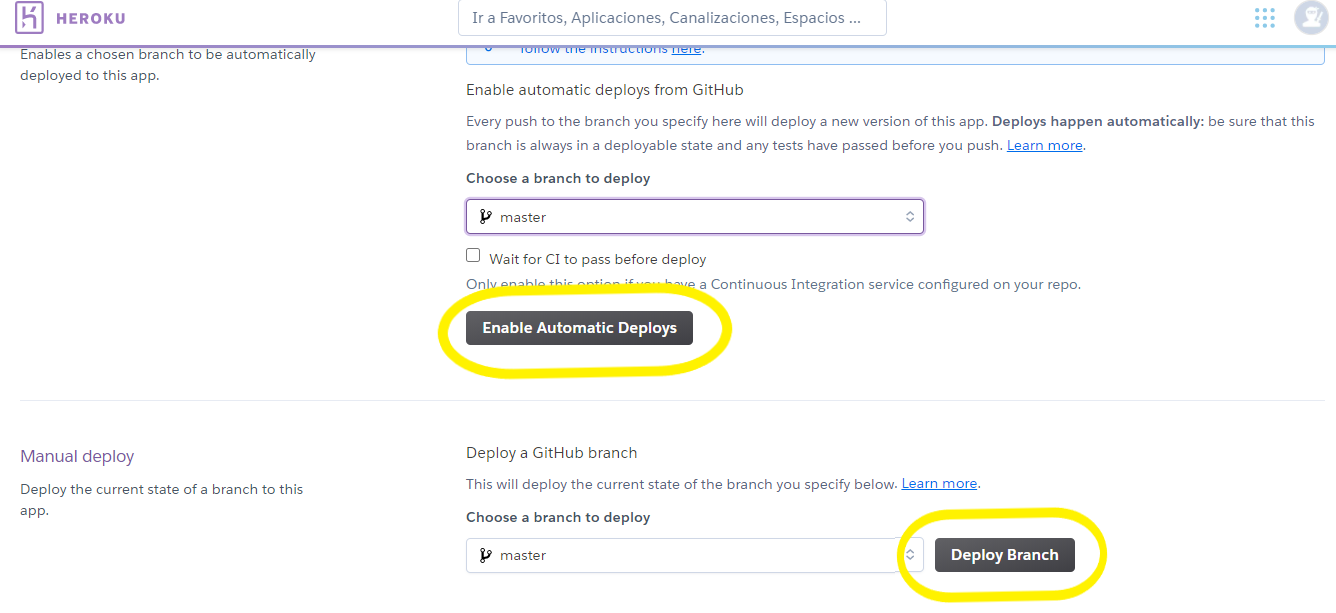
**Imagen 48:**Conectando Heroku y el repositorio paso 2



**Fuente: [propia]**

Activamos la opción de **Enable Automatic Desploys,** es la encargada de actualizar el servidor en caso de que se hagan cambios en el repositorio. Luego damos **Deploy,** esperamos a que termine la instalación. Bien la aplicación está ahora en producción.

**Imagen 49:** Puesta en producción de la aplicación web



**Fuente: [propia]**

**Información an**

**JDK java development kit**

**SDK software development kit**