**1.Introducción.**

WeatherPi es una página web creada con el objetivo de mostrar los datos meteorológicos de diferentes zonas de una forma amigable y sencilla.

Cuando pensamos en esta idea se nos ocurrió pensar en cómo juntar todo lo que aprendimos durante estos 2 años y pensamos en este proyecto en el que utilizamos la mayoría de cosas aprendidas en estos dos años.

Finalmente lo hemos llevado a cabo, y como hemos dicho al principio es una herramienta sencilla y útil para poder ver el tiempo y datos interesantes sobre el mismo reflejados en una web.

**2.Objetivos.**

* Utilizar todos los recursos aprendidos.
* Crear un prototipo de una estación meteorológica en una raspberry.
* Compartir información tomada del ambiente con un servidor de una web.

**3.Creación base de datos.**

Para comenzar con el php necesitaremos una base de datos con los datos que necesitaremos mostrar a traves de la pagina web.

La base de datos esta formada por 4 tablas: estaciones, votos, datos y usuarios. A continuación ponemos lo siguiente en la consola de mysql :

***CREATE TABLE `estacion`*** *(*

*`id\_estacion` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,*

*`* provincia *`* varchar(100) NOT NULL,

*`* pais*`* varchar(100) NOT NULL,

*`Nombre\_Estacion` varchar(50) NOT NULL UNIQUE*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

***CREATE TABLE `votos`*** *(*

*`id\_voto` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,*

*`num\_votos` int(9) NOT NULL,*

*`id\_usuario` int(11) NOT NULL,*

*`Foto` varchar(300) NOT NULL UNIQUE,*

*` Fecha` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

***CREATE TABLE `datos`*** *(*

*`id\_dato` int(20) NOT NULL AUTO\_INCREMENT PRIMARY KEY ,*

*`id\_estacion` int(11) NOT NULL ,*

*`Temperatura` int(5) NOT NULL,*

*`Humedad` int(5) NOT NULL,*

*`Luminosidad` int(5) NOT NULL,*

*`Lluvia` int(1) NOT NULL,*

*`Foto` varchar(300) NOT NULL UNIQUE,*

*` Fecha` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

***CREATE TABLE `usuarios`*** *(*

*`id\_usuario` int(11) NOT NULL AUTO\_INCREMENT,*

*`nombre\_usuario` varchar(20) DEFAULT NULL,*

*`contrasena` varchar(500) NOT NULL,*

*`correo` varchar(50) DEFAULT NULL,*

*`tipo` varchar(16) DEFAULT NULL,*

*PRIMARY KEY (`id\_usuario`),*

*UNIQUE KEY `correo` (`correo`)*

*) ENGINE=InnoDB AUTO\_INCREMENT=36 DEFAULT CHARSET=utf8;*

**Con el trigger para borrar los usuarios.**

*delimiter //*

*CREATE TRIGGER borrar\_usuarios AFTER DELETE ON proyecto.usuarios*

*FOR EACH ROW BEGIN*

*DELETE*

*FROM votos*

*WHERE votos.id\_usuario = old.id\_usuario;*

*END;*

*//*

*delimiter ;*

**Base de datos estacion:**

***CREATE TABLE `estacion`*** *(*

*`id\_estacion` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,*

*`Nombre\_Estacion` varchar(50) NOT NULL UNIQUE*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

***CREATE TABLE `datos`*** *(*

*`id\_dato` int(20) NULL ,*

*`id\_estacion` int(11) NOT NULL ,*

*`Temperatura` int(5) NOT NULL,*

*`Humedad` int(5) NOT NULL,*

*`Luminosidad` int(5) NOT NULL,*

*`Lluvia` int(1) NOT NULL,*

*`Foto` varchar(300) NOT NULL ,*

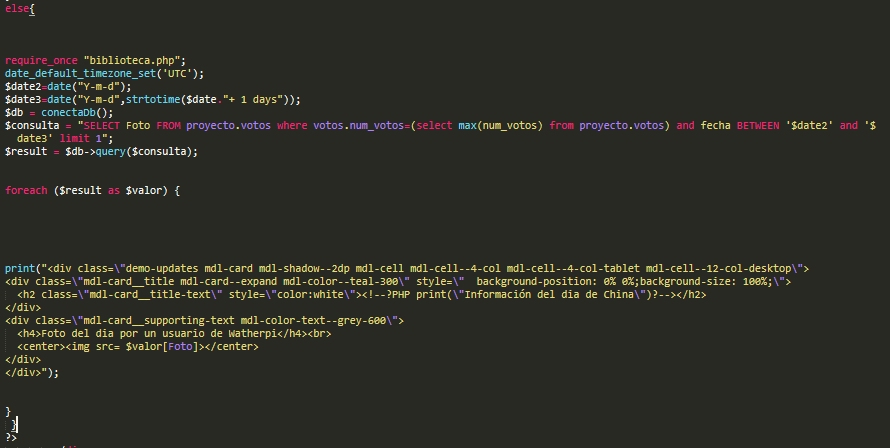
*` Fecha` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

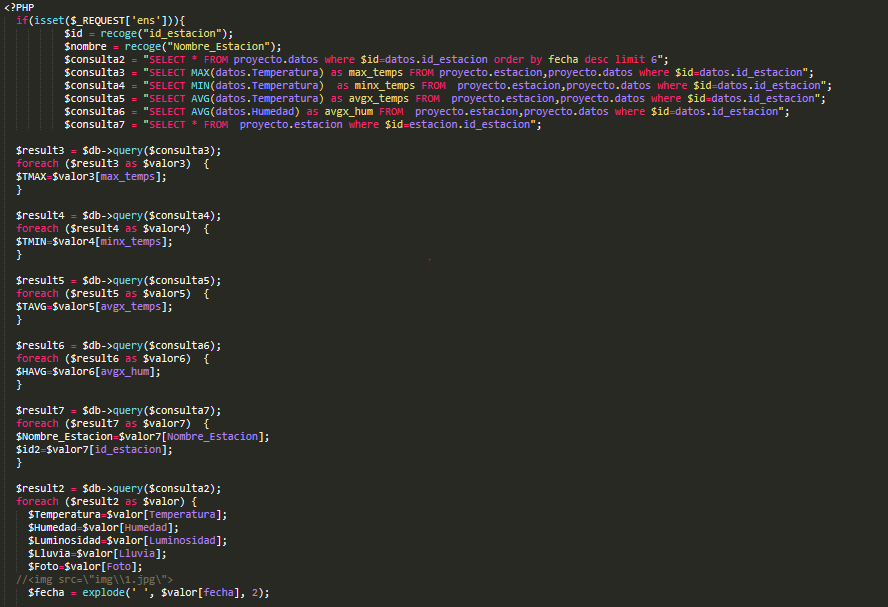
**4.Creación del PHP.**

Index.php

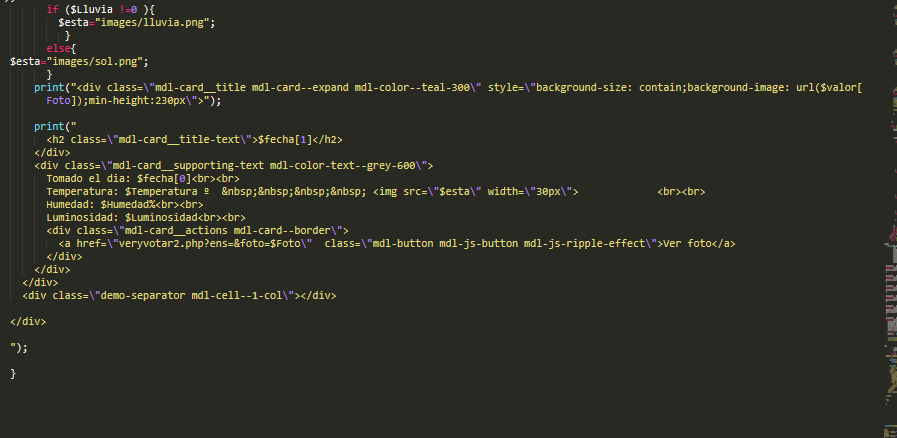
Para el php decidimos que en la página principal se mostraría la imagen más votada del día con el siguiente código:



Lo siguiente seria realizar las consultas necesarias una vez se había seleccionado una opción en el input y mostrar los resultados dentro de sus contenedores.

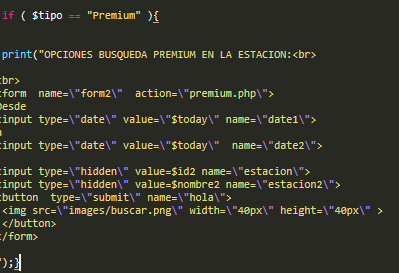


Aquí hacemos las consultas(Decidimos ponerlas por separado ya que así si quisiéramos agregar nuevos valores en la base de datos nos seria mas fácil la modificación).



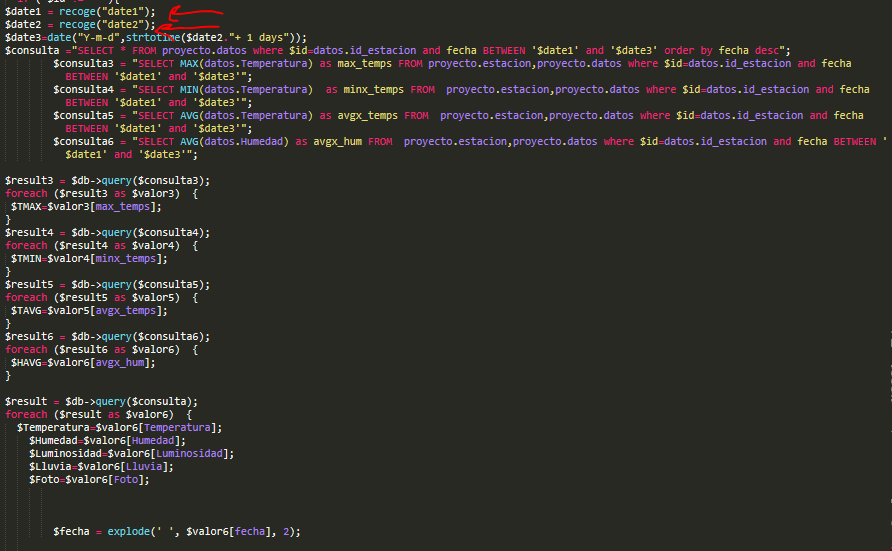
Mostrar los datos.

Restringimos el acceso a los que no sean premium a las acciones premium.



Premium.php

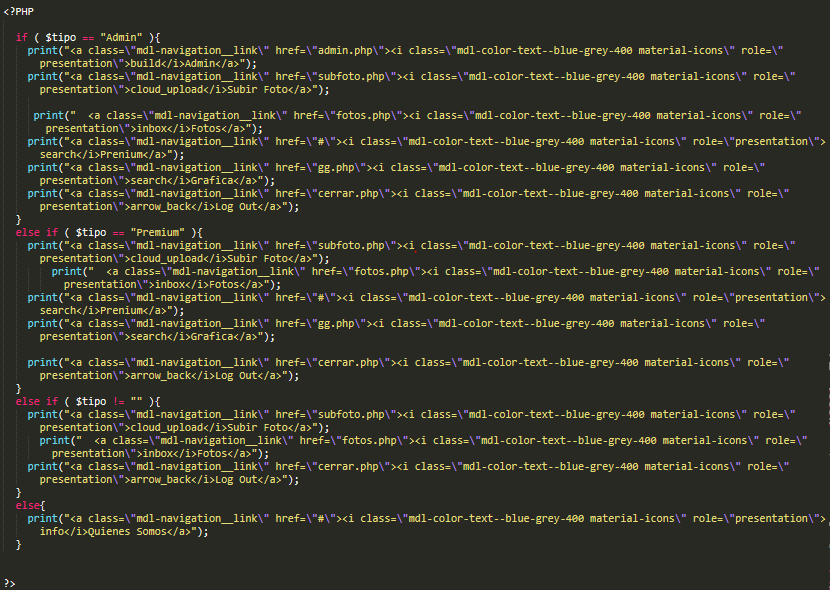
Aquí es donde te manda una vez buscas entre las 2 fechas que decidiste hace un select también al igual que el index pero con la diferencia de que estos datos están entre estas dos fechas.



Las flechas muestran los datos recogidos del index.php

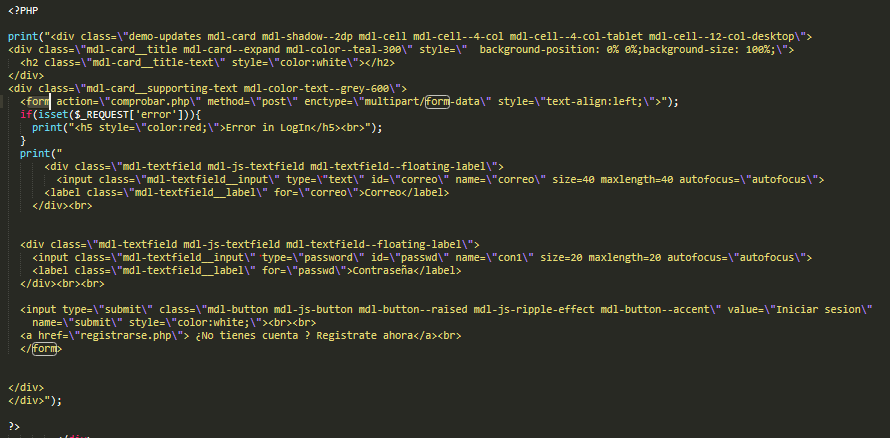
El menú.

Este menú está situado a la izquierda y muestra las acciones que puede realizar el usuario.(Esta presente en todas las paginas)



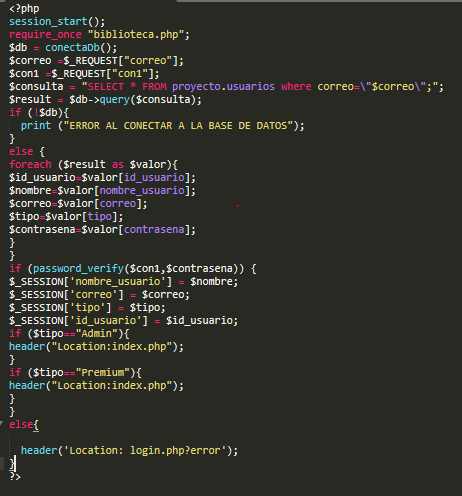
Login.php

Desde aquí saltara un formulario el cual podrás rellenar con tu correo y contraseña en el caso de que ya estés registrado podrás acceder a tu cuenta a través de sesiones, para poder comprobarlo usaremos la siguiente página.



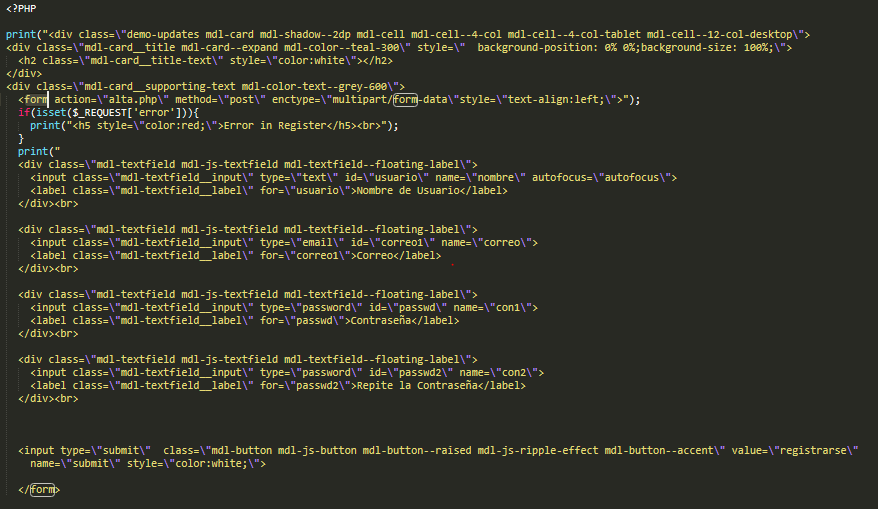
Comprobar.php

Este php establece una sesión una vez comprueba que el usuario existe en la base de datos y la contraseña es la misma.



Registrarse.php

En el caso de no estar registrado esta pagina de encarga de genera un formulario que se manda a alta.php .



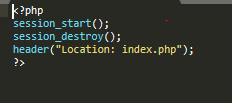
Alta.php

Este se encarga de introducir los datos enviados de registrarse.php en la base de datos.



Cerrar.php

Utilizado para cerrar la sesión si se inicia correctamente.



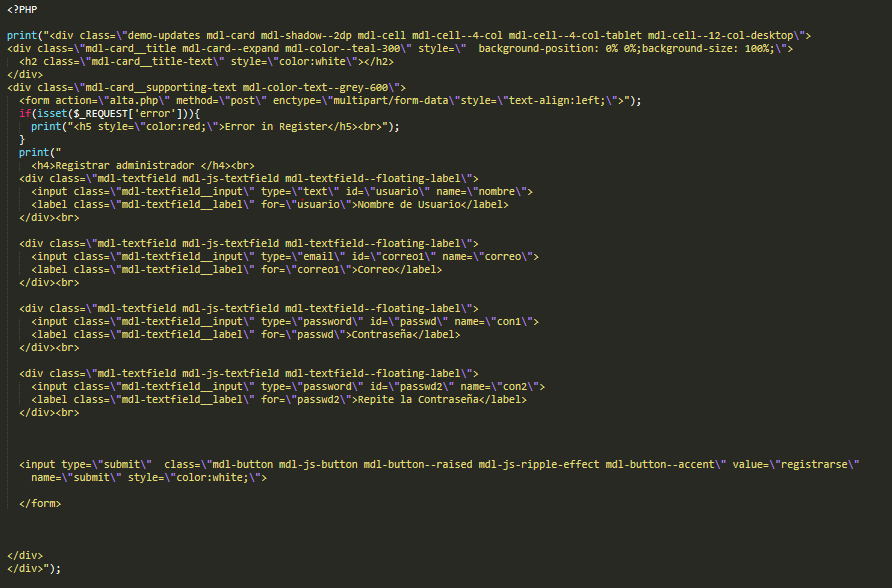
Admin.php

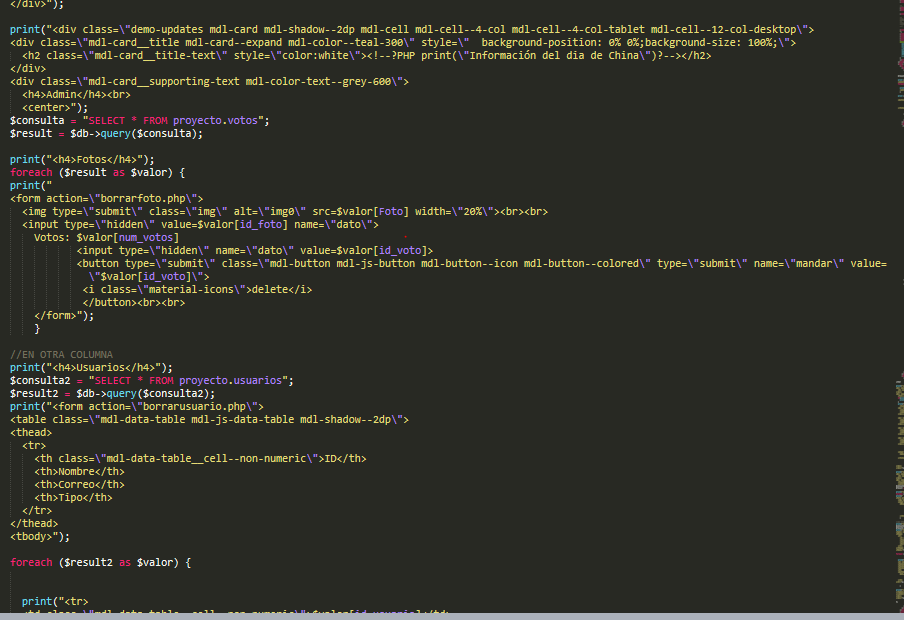
En el caso de que tu sesión sea de tipo admin podrás acceder a esta pagina en la cual tendrás acceso a las opciones de administración.

Hay 2:

* + Crear administrador el cual mandara el formulario a alta.php.
  + Borrar fotos/usuarios:

Usa el siguiente código para mostrar los usuarios y fotos las cuales puedes borrar.



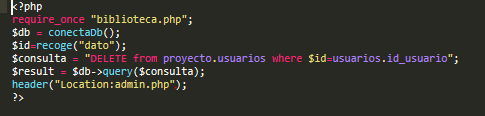




Después al darle clic en el botón de la basura pasara por borrarusuario.php o a borrarfoto.php dependiendo de lo que sea.

Borrarusuario.php

Como su nombre indica a esta página llegan los datos pasados por admin.php y borra el usuario seleccionado .



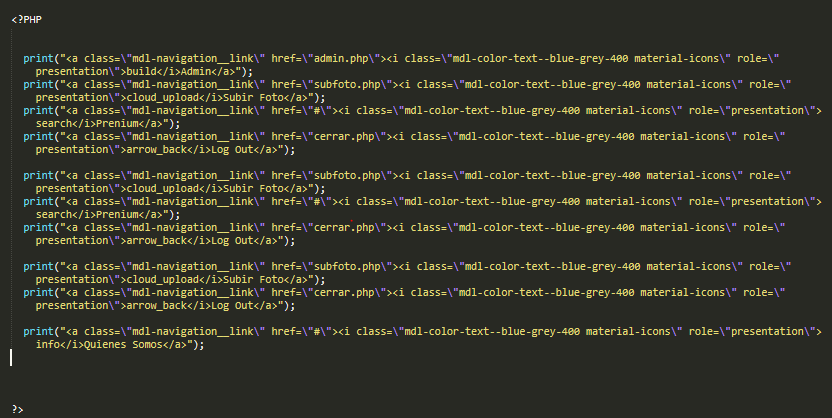
Borrarfoto.php

Al igual que el anterior hace lo mismo pero con las fotos con la diferencia que esta tambien borra las fotos de la propia carpeta.



Subfoto.php

Este php servira para poner simplemente el input que nos pasara el archivo.



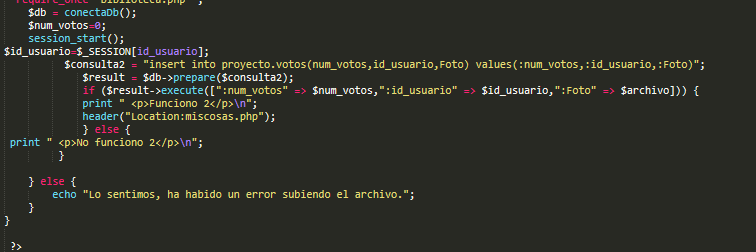
Upload.php

El archivo de subfoto.php se pasara aquí, y es aquí donde subirá el archivo y lo guardara en su capeta correspondiente además de guardarlo en la base de datos.

Aquí se comprueba que la imagen sea válida y se mueve a su lugar en caso que lo sea



Guarda la direccion y el id del usuario en la base de datos.



Fotos.php

Muestra todas las imágenes subidas por los usuarios y puedes votar por ellas 1 vez.



Para controlar los votos utilizamos sesiones.



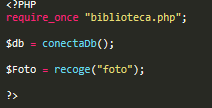
Miscosas.php

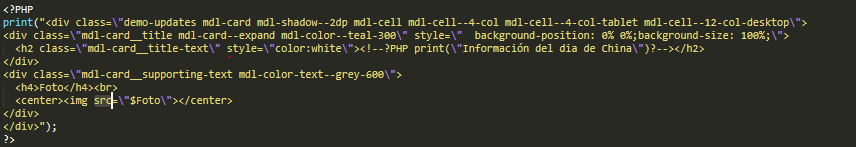
Aquí es donde cada usuario podra ver las fotos que subio a la plataforma y puede decidir eliminarlas si fuera necesario. A traves de borrarfoto.php.(anteriormente mencionado)



Veryvotar2.php

Muestra la imagen que quieras ver de la estacion meteorológica recogiendo la direccion de la foto de index.php.





Gg.php

Utilizamos javascript para sacar informacion de la base de datos y mostrarla a traves de una

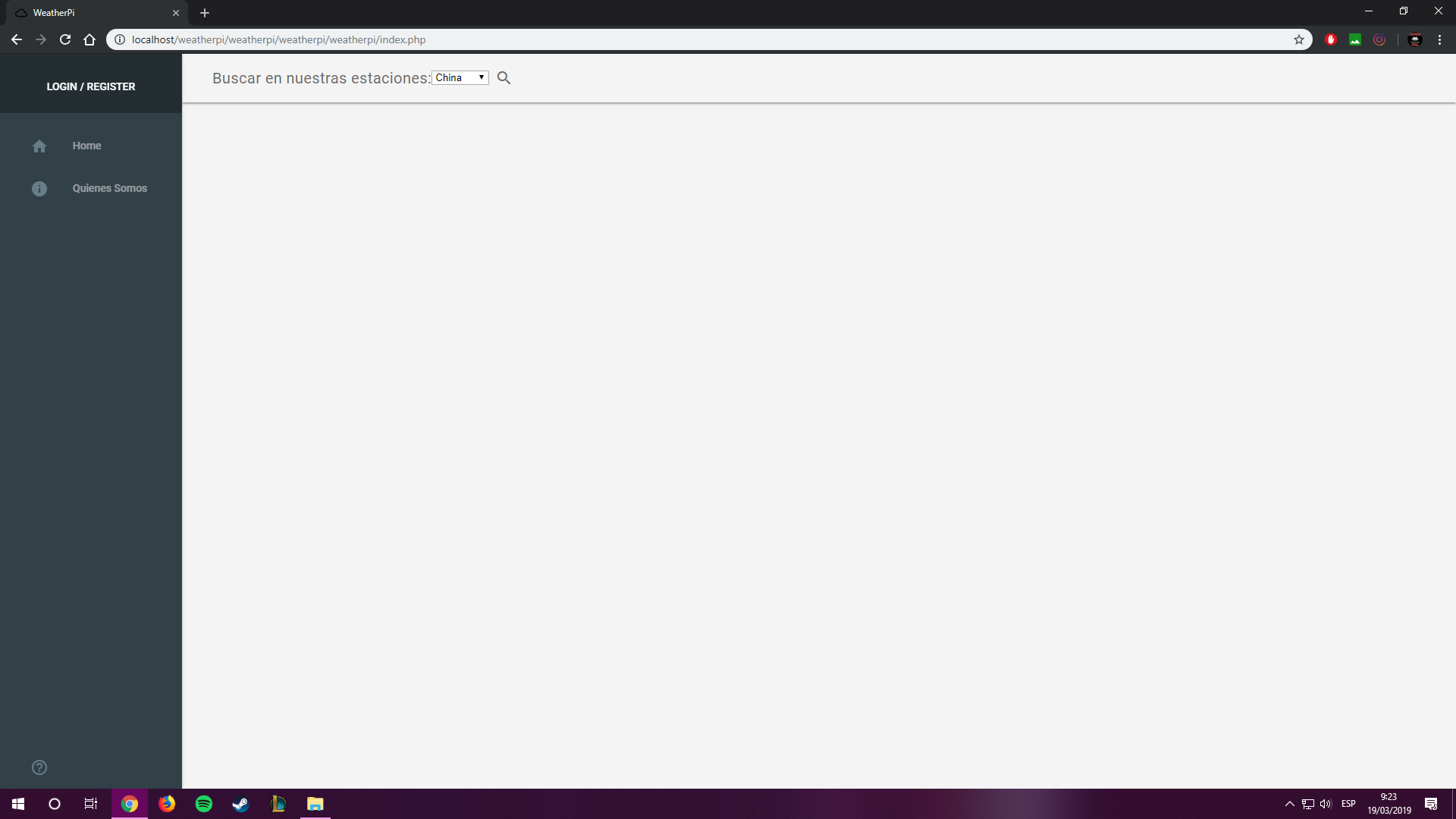
Grafica, estas funciones estan disponibles en la web.

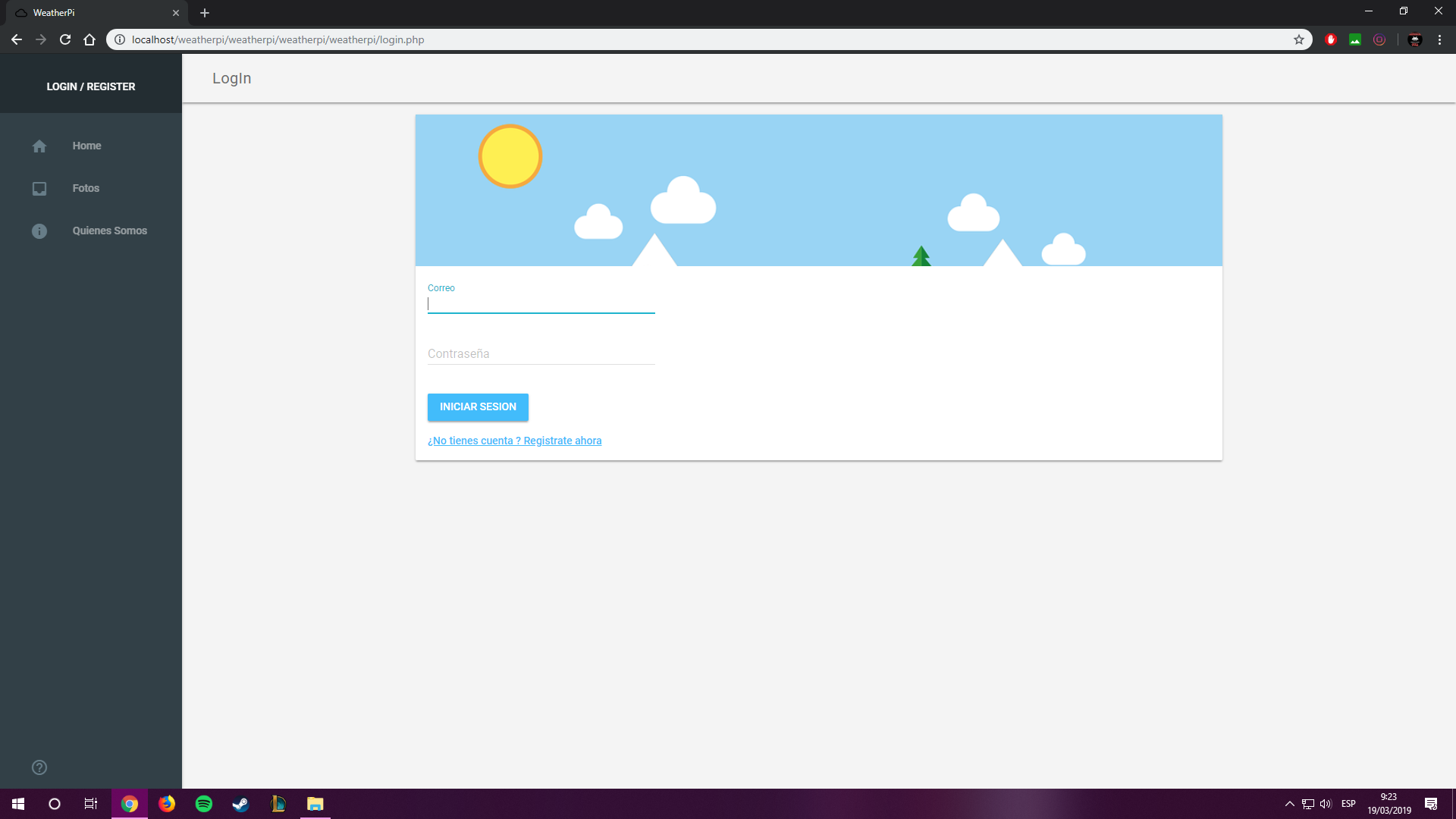
**5.Creación del diseño Web.**

Nuestra web consiste en un lugar intuitivo y cómodo donde poder ver reflejados los daos recogidos por la raspberry y los sensores asi como ofrecer algunas funcionalidades más como pueden ser graficas de los datos o poder votar por las fotos que los usuarios han subido a la web.

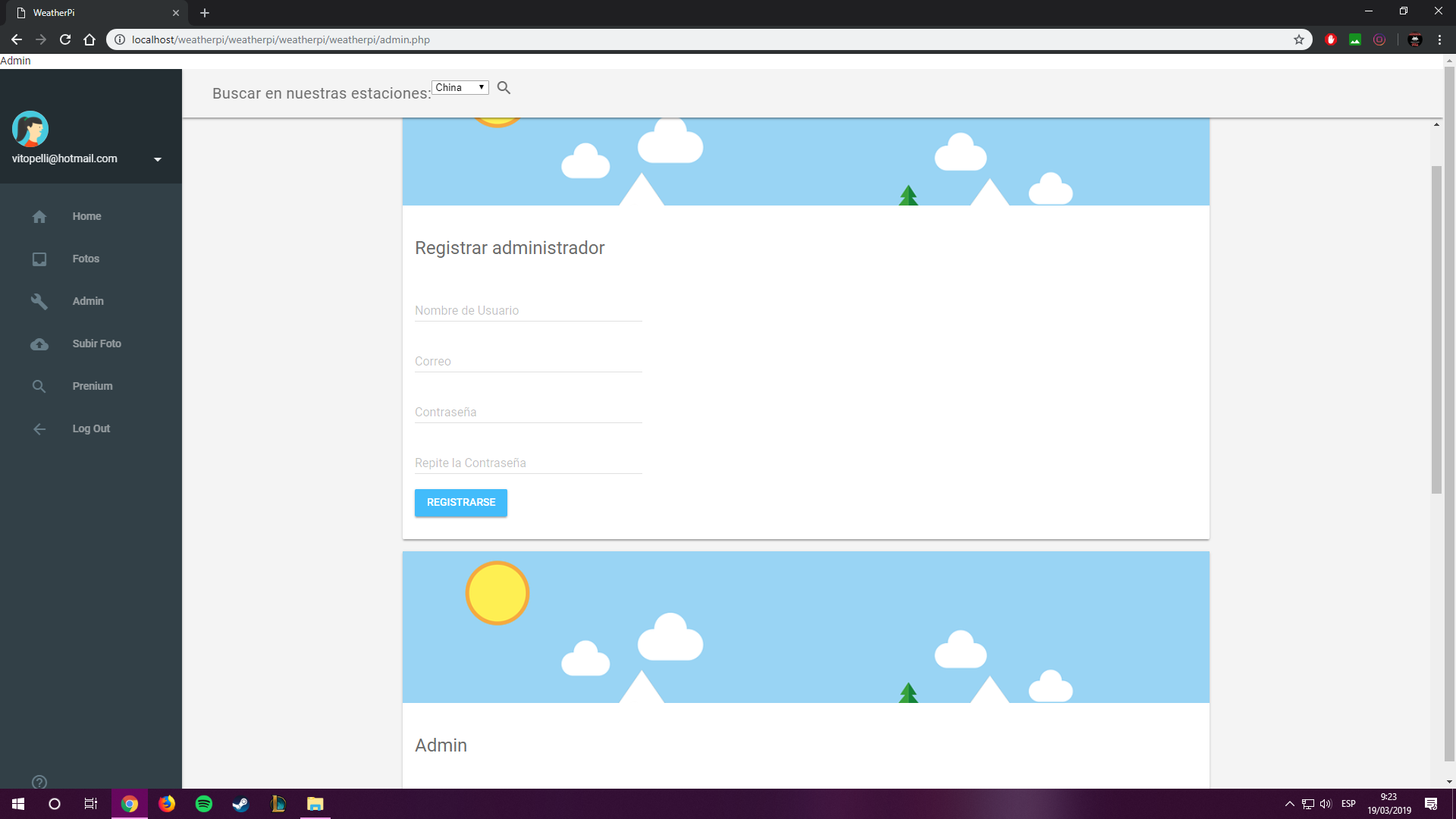
Para todo esto hemos utilizado mysql para administrar las bases de datos donde se almacenan los datos que obtenemos, y php para hacer la web, que está en un servidor local

La página de index es una una página muy simple en la que se muestra en todo el centro una imagen del dia, y a la izquierda tenemos la opción de iniciar sesión, y muy pocas opciones más, que para poder acceder a todas las opciones, se tiene que iniciar sesión con una cuenta, y si quieres acceder a todas las funcionalidades de la web, se tiene que iniciar sesión como premium.

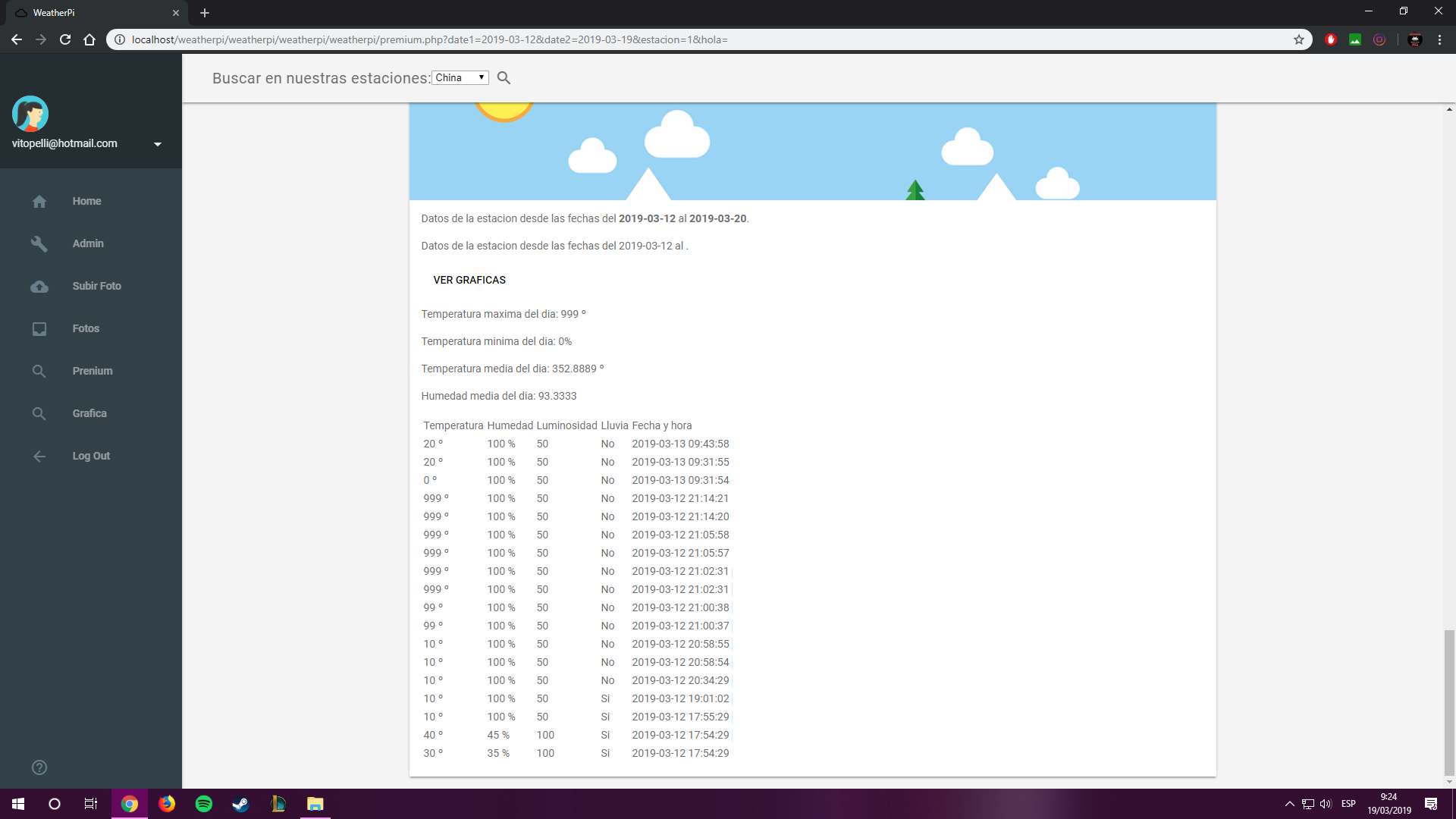


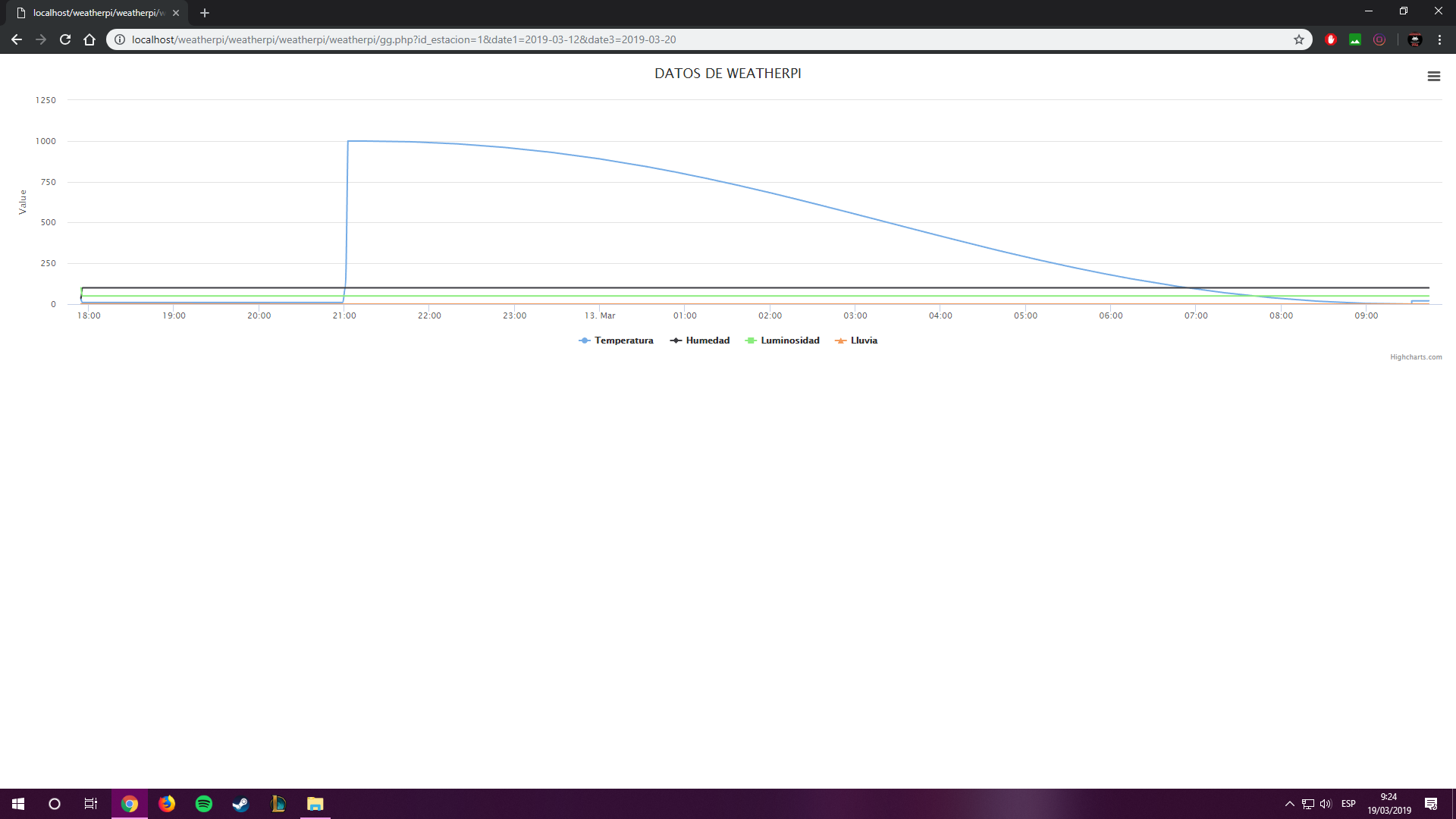


Una vez iniciada sesión, se puede ver que hay más opciones disponibles, como son subir foto y fotos, que son unos menús donde los usuarios pueden subir fotos a que hayan hecho ese día, y luego el resto de usuarios pueden ver y votar las fotos, la foto más votada es la que aparece en el index de la página, cada día se renueva.

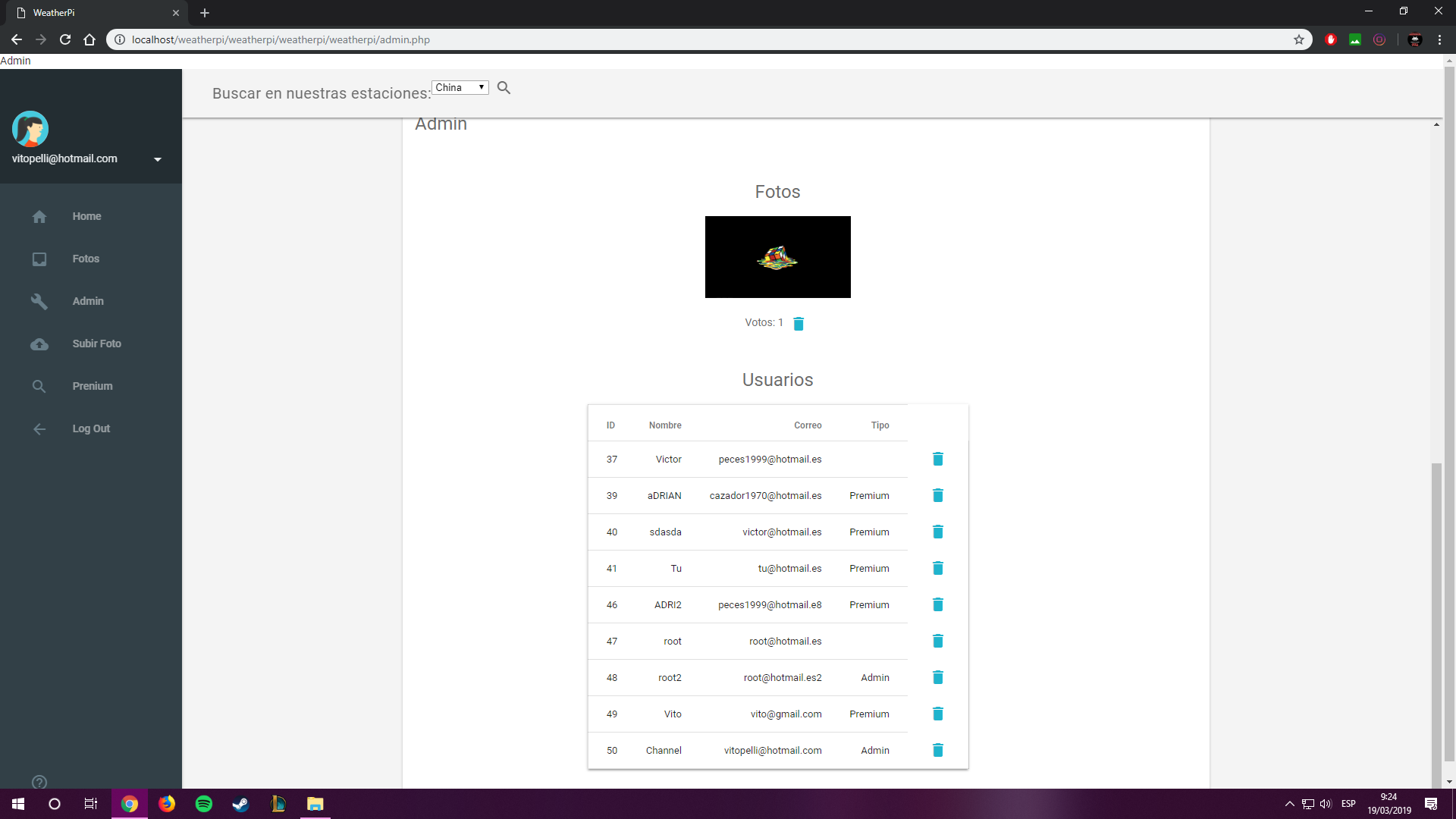


Los usuarios premium se diferencian de los normales en que tienen la opción de poder elegir entre dos fechas que quieran y les mostrara todos los datos de esos días, y también hay una opción de ver en una gráfica todos los datos que se han recogido, pudiendo elegir los datos que quieres ver, así como descargar el grafico en pdf.





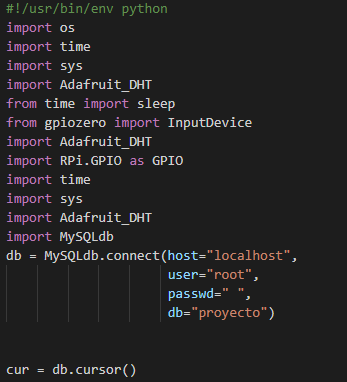
Luego finalmente tenemos la parte de backend donde el usuario administrador de la pagina puede crear nuevos usuarios administradores, borrar fotos que los usuarios hallan subido, pr ejemplo cuando no son apropiadas para la página.



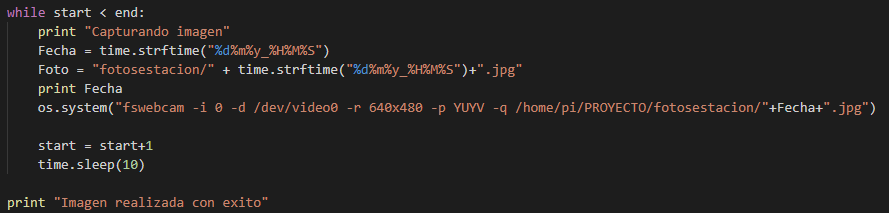
Con respescto al diseño en forma de codigo de la web, hemos utilizado un diseño estilo matrial desing, ya que es un diseño que desde que google saco android la mayoria de personas estamos acostumbradas a ver y es muy sencillo e intuitivo.

Hemos utilizado bootstrap para el diseño de la web además de un css para la colocacion de algunos de los elementos de la web

**6.Creación del codigo Python.**



Este es el código para la conexión de los sensores con la base de datos en python



Con este código ejecutamos la cámara para que realice una foto

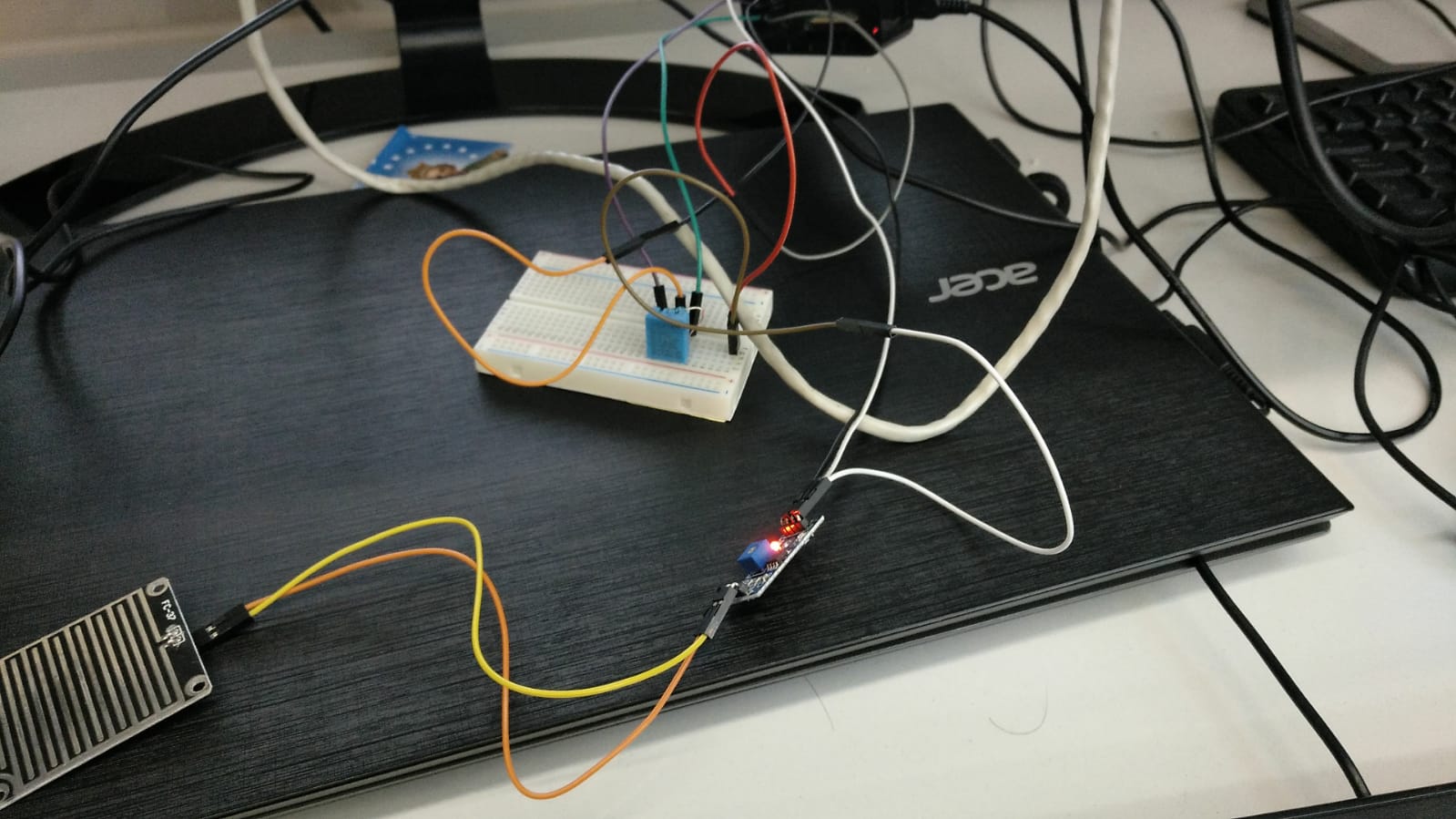


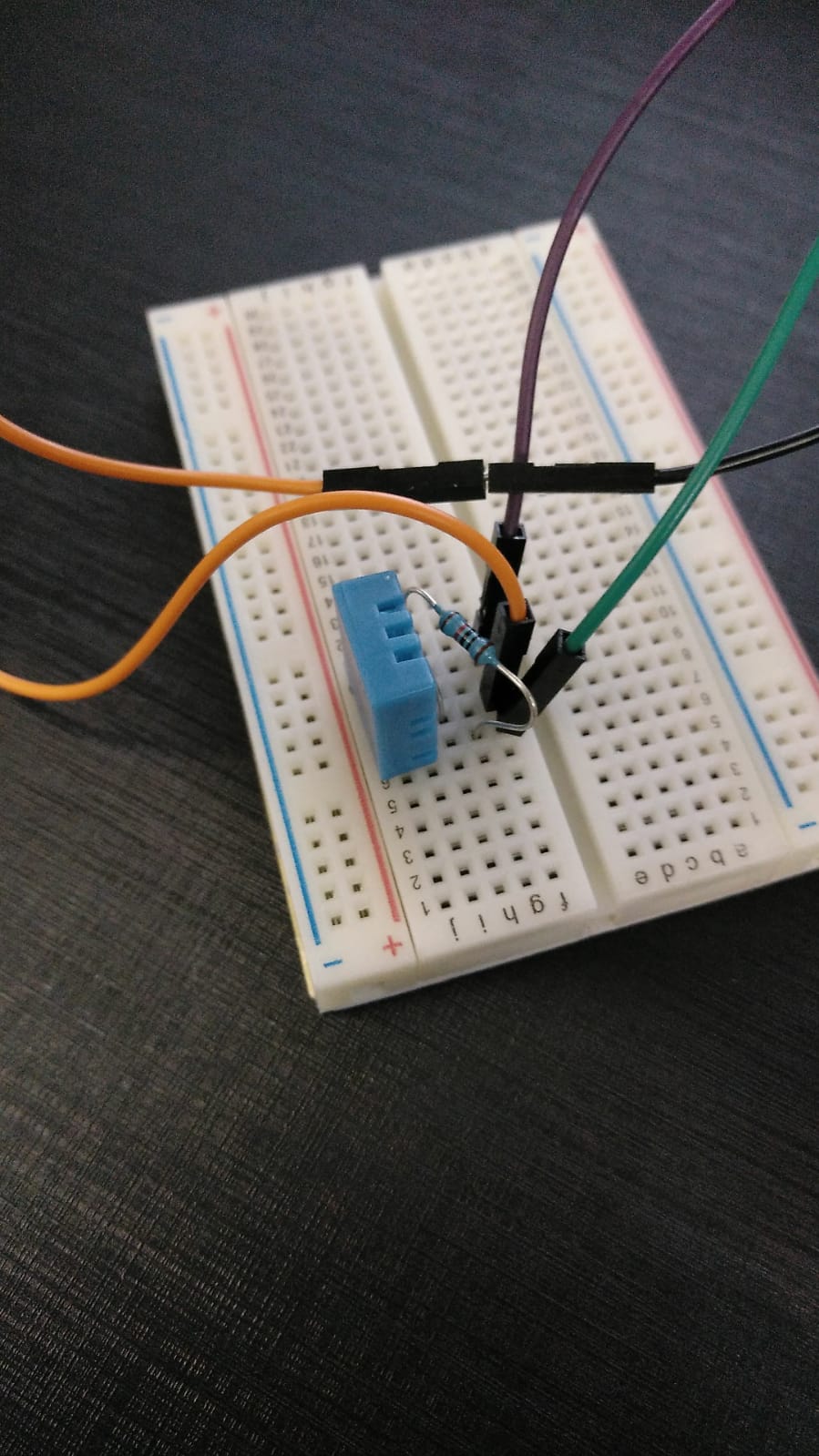
Aquí ejecutamos el sensor de temperatura y humedad, y se necesitan la biblioteca Adafruit para ejecutarse

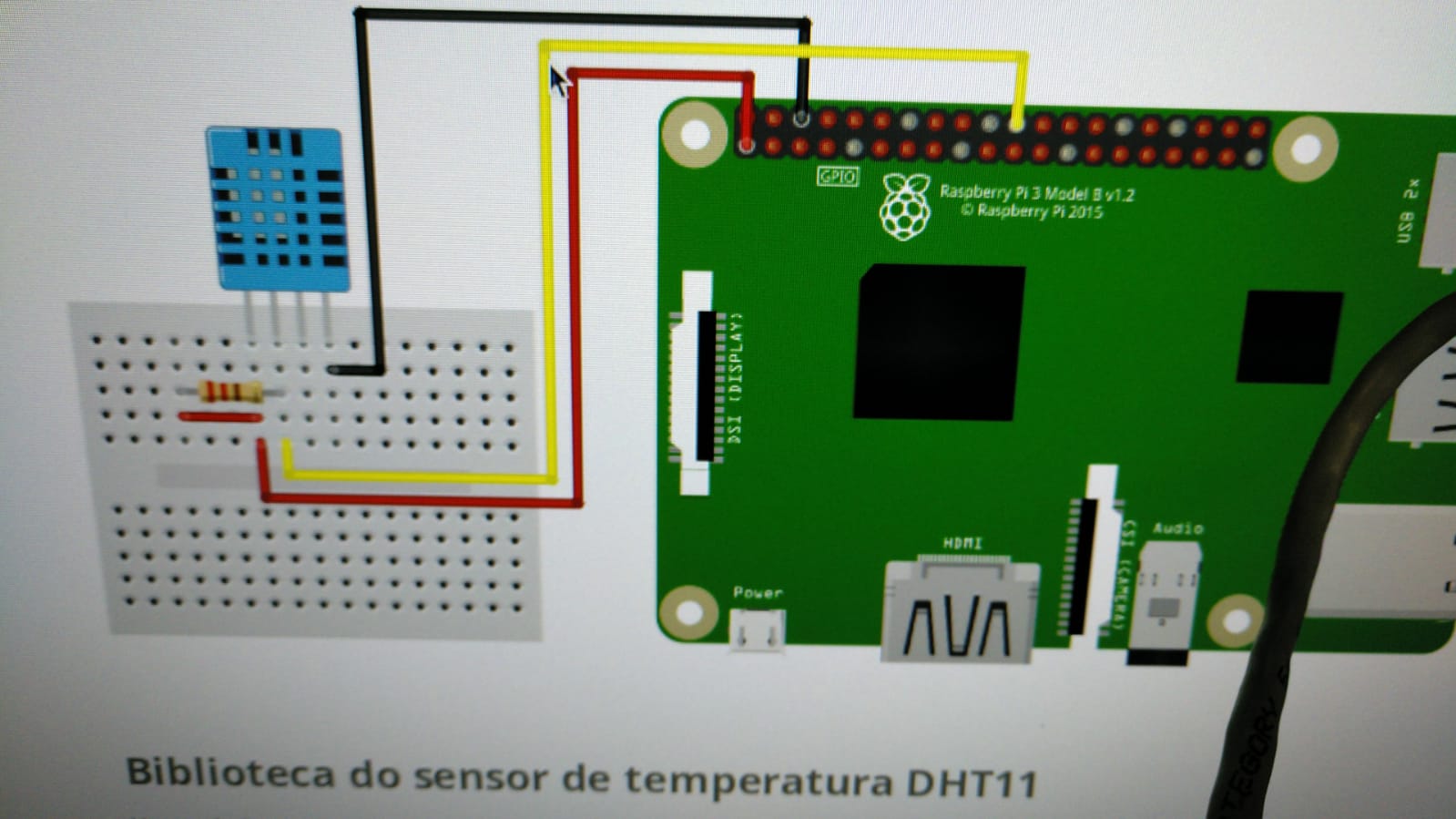


Y finalmente imprimimos todos los datos y los insertamos en la base de datos que después sincronizaremos con la base de datos de windows y se verán reflejados esos datos en la web

Conexión de los sensores a la raspberry







**7.Compartir la carpeta mediante el servicio Samba.**

Optamos por Samba debido a que este conjunto de aplicaciones implementan un protocolo de comunicación SMB mediante el cual podemos compartir ficheros, archivos... a través de carpetas compartidas.

La diferencia con otros protocolos de carpetas compartidas de Linux es que este hace posible que se pueda pasar información entre un dispositivo Linux y otro Windows.

Pasos seguidos para la instalación de Samba:

1. *Sudo apt-get update*
2. *Sudo apt-get upgrade*
3. *Sudo apt-get install samba samba-common-bin*
4. Editamos el archivo smb.conf mediante *sudo nano /etc/samba/smb.conf*
5. Descomentamos la linea wins support y ponemos “yes” en el lugar de “no”.
6. Creamos la carpeta que queremos compartir *mkdir PROYECTO*
7. Volvemos a */etc/samba/smb.conf*
8. Ponemos lo siguiente al final del fichero:

*[PROYECTO]*

*comment = Carpeta para proyecto*

*path= /home/pi/PROYECTO*

*browseable = yes*

*writeable = yes*

*only guest = no*

*create mask = 0777*

*directory mask = 0777*

*public = no*

1. Creamos el usuario para samba con *sudo smbpasswd –a pi*
2. Rellenamos.
3. Ponemos los permisos al usuario en el fichero *sudo chmod 777 /home/pi/PROYECTO*
4. Reiniciamos el servicio con *sudo service smbd restart*

Con esto ya tendríamos creada la carpeta , para acceder a ella solo hay que poner //IPRASP una vez entremos nos pedirá el usuario y contraseña , completaremos estos campos con los del usuario creado anteriormente.

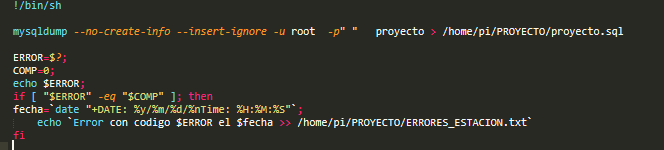
**Nota: Pusimos la ip fija para que no crease problemas con el script.**

**8.Scripts.**

Una vez en este punto tenemos que crear los 2 scripts que necesitaremos :

8.1.PROYECTO.sh

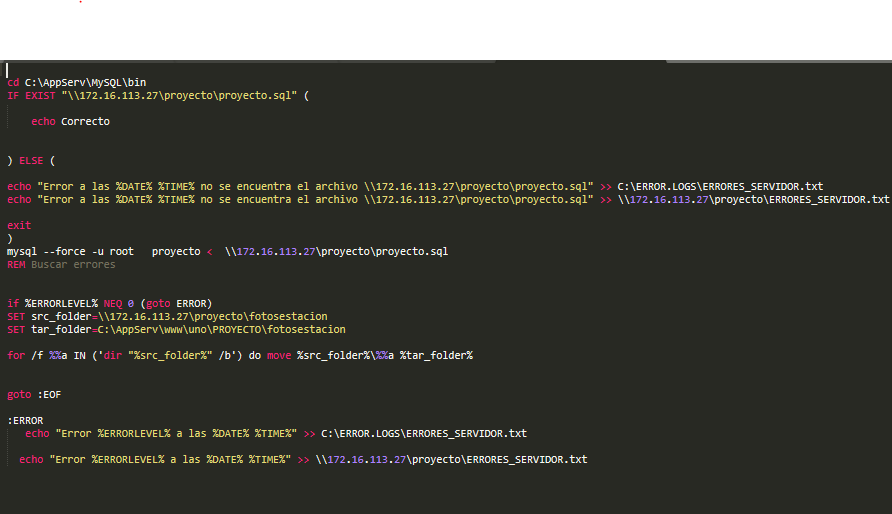
Este script lo único que hace es hacer una copia en la base de datos en la cual solo se meten los datos insertados no la estructura, de esa forma evitamos problemas a la hora de volcarla en la principal.



8.2.PROYECTO.bat

Este script sirve para mover las imágenes de las fotos de la cámara de la estación a

la web y para insertar todos los datos nuevos en la base de datos.

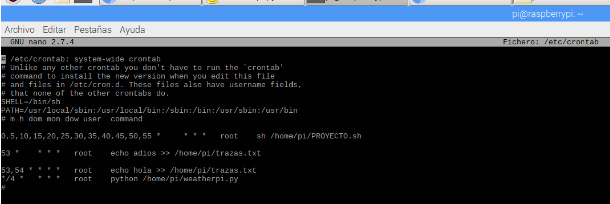


**9.Tareas.**

En estos puntos vamos a explicar como están la programación de tareas en caso de windows o crontab en caso de linux.

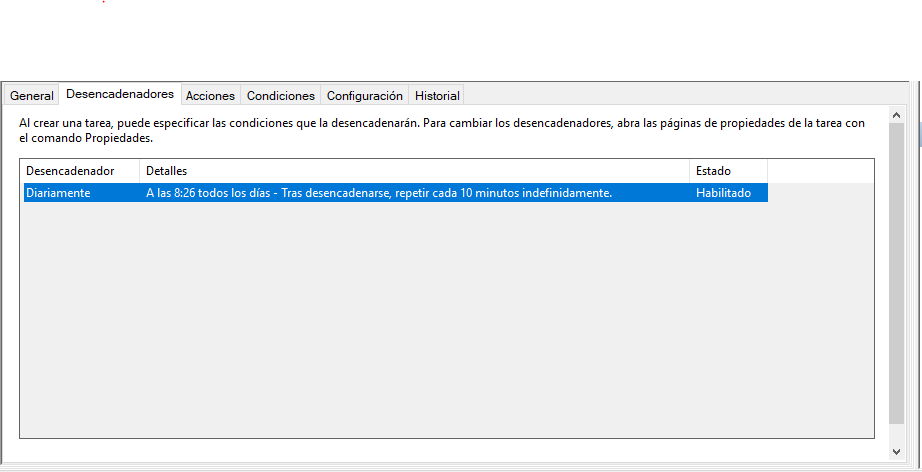
9.1.crontab(Linux).

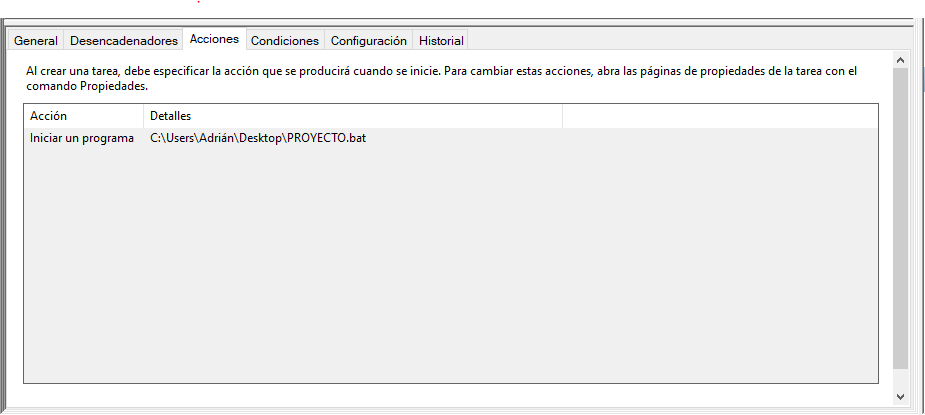
Debido a que la estación la idea es que este encendida continuamente hemos decidido ponerlo en el crontab. Este está programado para que el python weather.py se ejecute cada 4 minutos y cada 5 se subirán los nuevos datos a la base de datos. Esto es así ya que puede demorar algo de tiempo weatherpi por lo tanto de esa manera aseguramos que los datos se guarden correctamente.



9.2.Programador de tareas(Windows).

Hacemos que se active cada 10 minutos de forma ilimitada ya que la idea es que siempre este recibiendo datos, ponemos 10 minutos para que así tenga mas datos de una vez y optimizar los recursos del equipo Tras elegir las opciones, así es como queda:





**10.Mejoras para un futuro.**

Mientras avanzábamos en el proyecto se nos ocurrieron varias ideas para mejorar el proyecto y estas son las principales mejoras que hemos pensado para WeatherPi.

* + Crear una funda para la estación.
  + Utilizar una raspberry más pequeña.
  + Añadir más sensores y más sensibles por tanto mayor información en la web.
  + Obtener un dominio público.
  + Poner varias estaciones.
  + Mejorar el sistema de las graficas