**Reconocimiento de Emociones**

Este proyecto se realiza una aplicación la cual permite reconocer la emoción con tomarse una foto, el servidor está hecho con flask con la ayuda de opencv para el reconocimiento y el modelo FisherFaces.

**Comenzando 🚀**

Para tener el proyecto únicamente descárgalo desde mi repositorio o si tienes conocimiento en git usa git clone

**Pre-requisitos 📋**

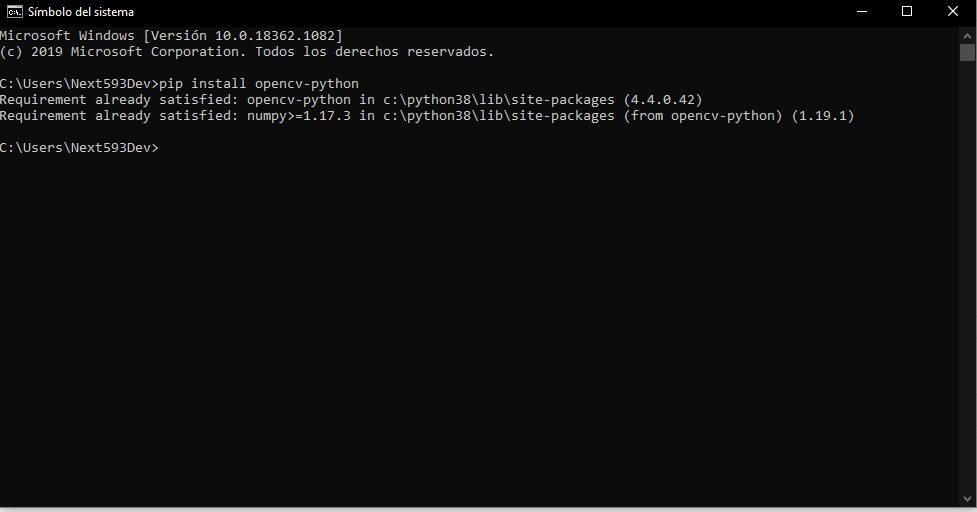
* Visual studio code
* Pyhton 3.8.5

**Instalación 🔧**

Para comenzar debemos instalar opencv para ello vamos al terminal de consola en este caso tengo windows

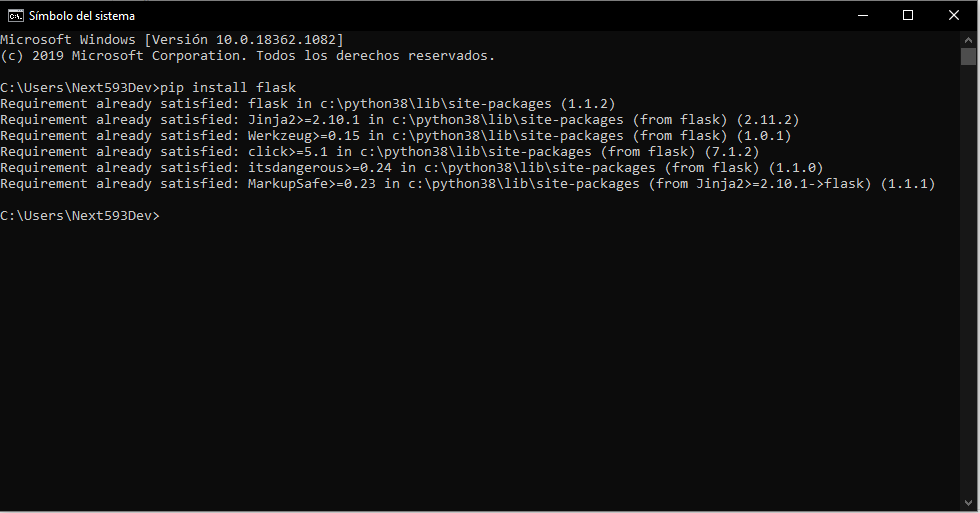
Usamos el comando *Pip install* "Libreria que vamos a instalar"

* *pip install open-cv-python*

[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/image.png)

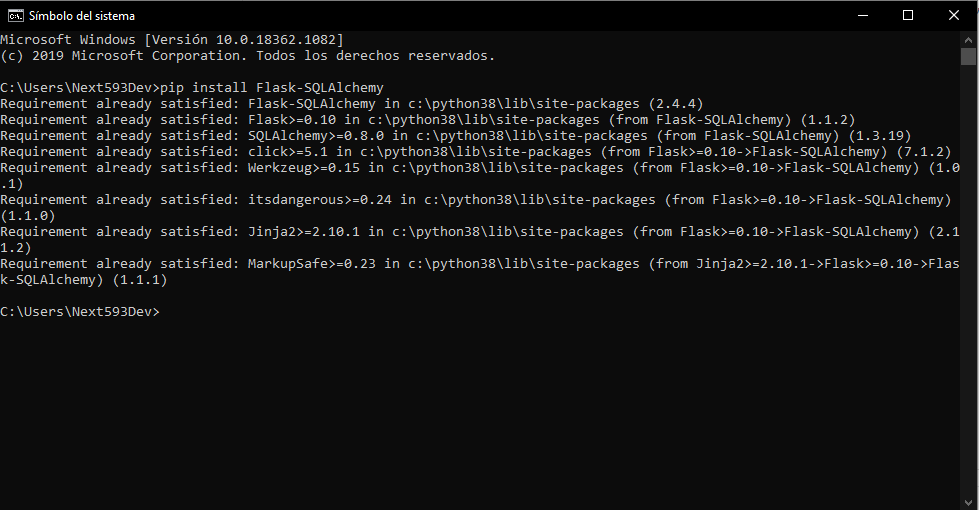
**Ahora instalamos Flask**

* *pip install Flask*

[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/installFlask.PNG)

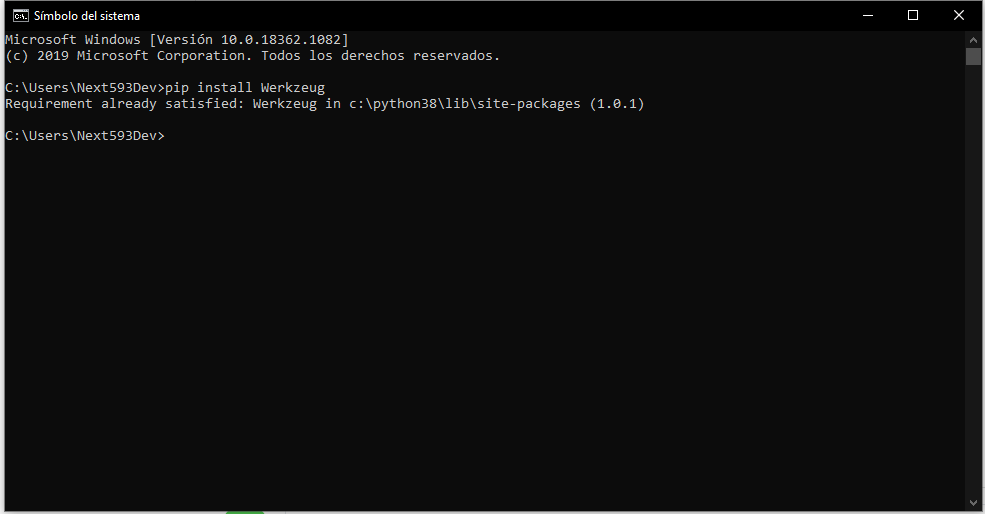
**Instalacion de flask\_sqlalchemy**

* *pip install Flask-SQLAlchemy*

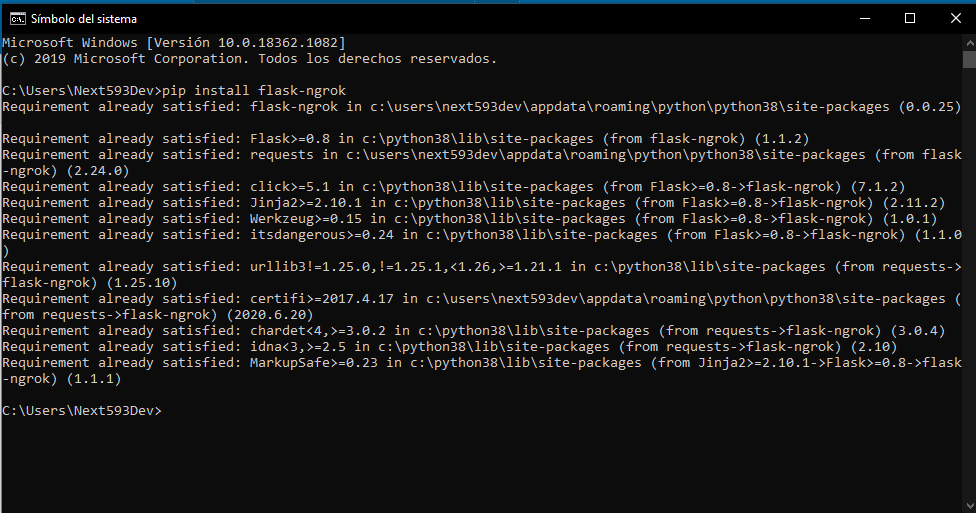
[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/FlaskAchemy.PNG)

**Instalación de Werkzeug**

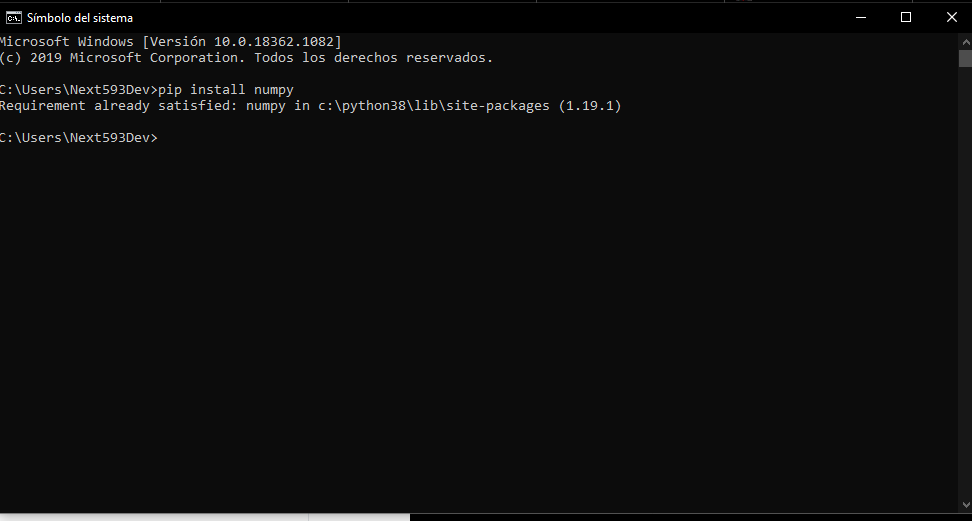
* *pip install Werkzeug*

[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/Werkzeug.PNG)

**Instalación de Flask-ngrok**

[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/flaskngrok.PNG)

**Instalación de numpy**

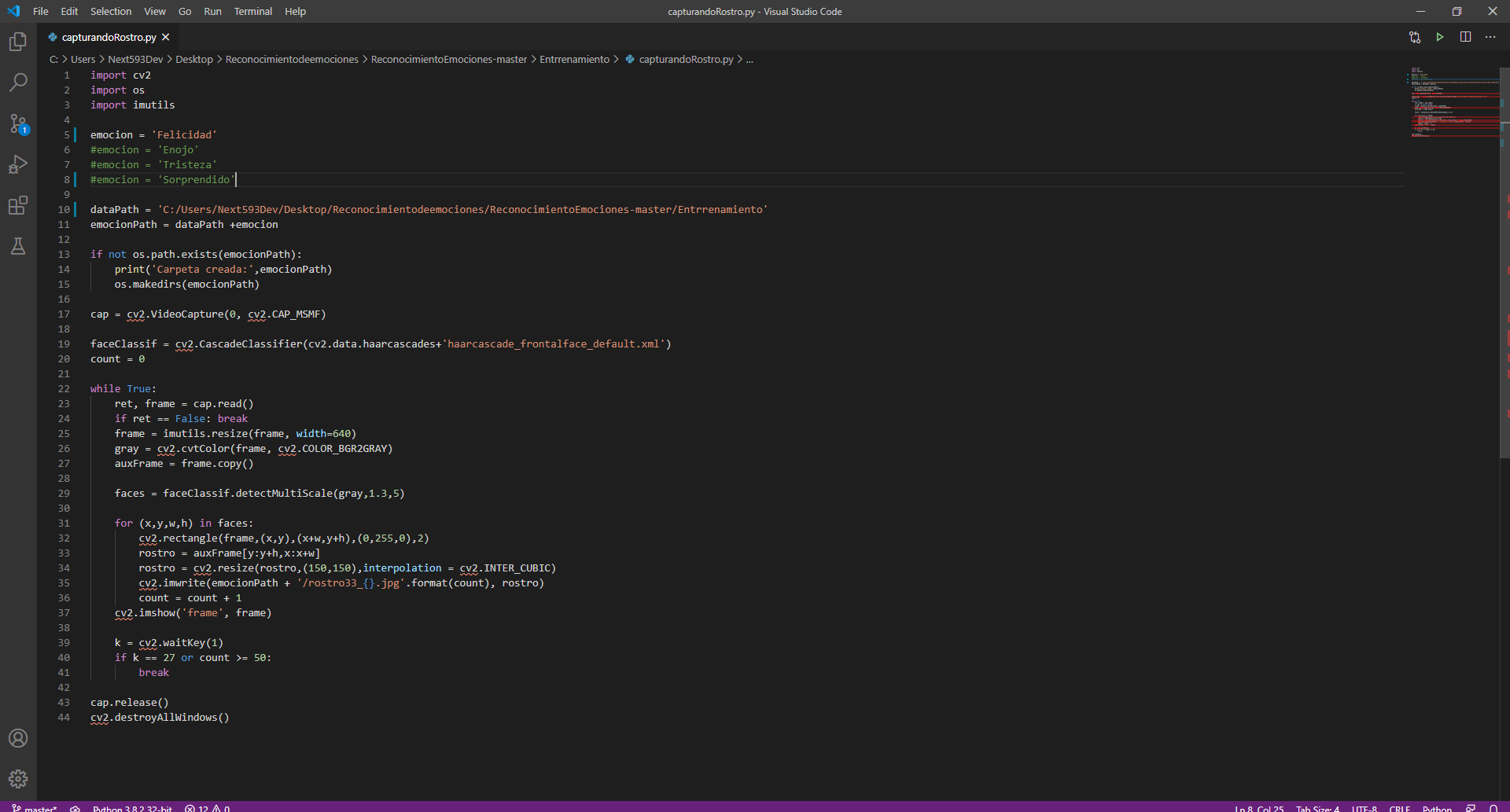
[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/Numpy.PNG)

**Creando la carpeta data 🔧**

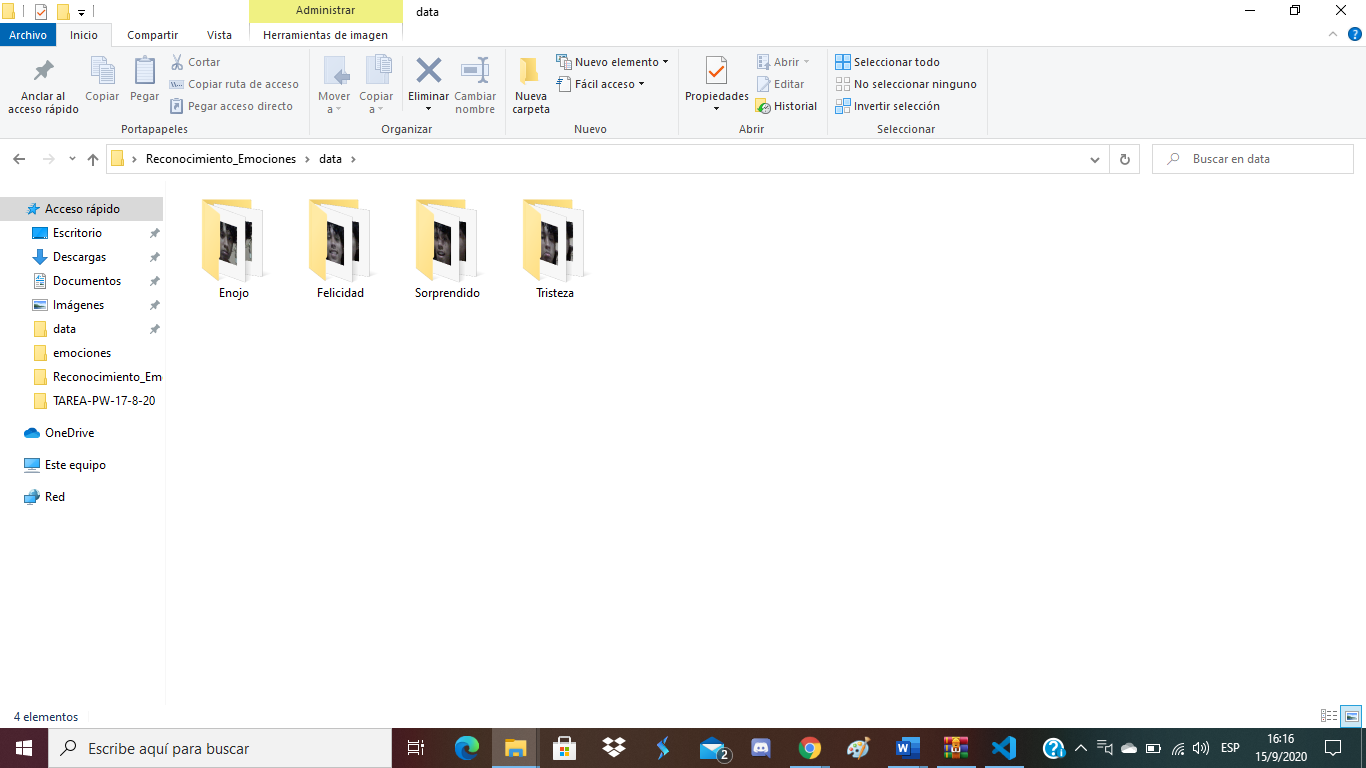
En este punto nos tomaremos foto usando python para esto nos vamos a la carpeta entrenamiento, abrimos el archivo *capturandoRostro.py* con visual studio code.

En las líneas 5 6 7 8 están las emociones para comenzar des-comentamos la emoción que vamos a guardar(capturandoRostro.py toma fotos y las guarda en un carpeta según la variable emoción)

La variable *dataPath* es la ruta donde se guardará la carpeta data con las fotos (recomendación usar la ruta de la carpeta entrenamiento añadiéndole data como se muestra la imagen)

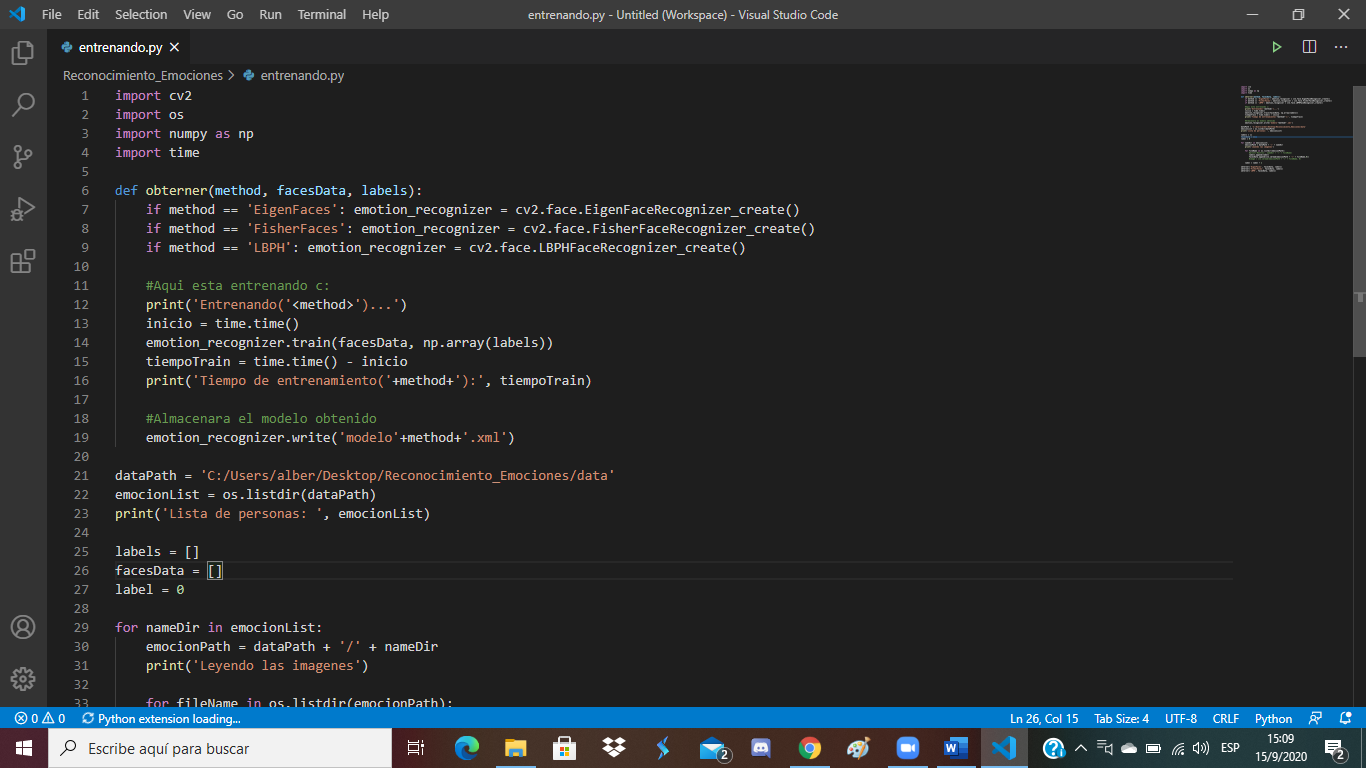
[](https://raw.githubusercontent.com/AngelC01/Imagenes/master/EmocionCaptura.PNG)

En la imagen que se ve a continuación se ve las carpetas que ya han sido creada con su debida emoción los cuales son Felicidad, Sorprendido, Tristeza y Enojo.

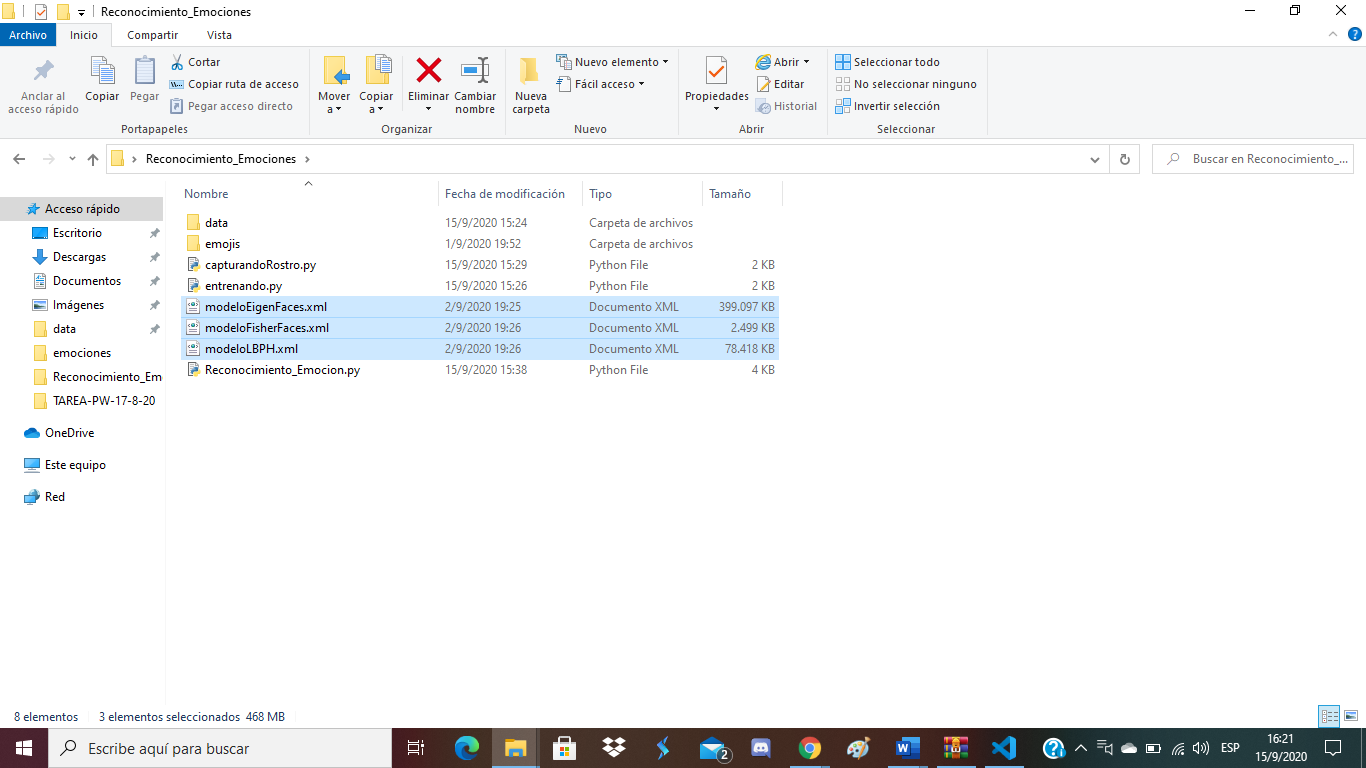


**Entrenando Data de Emociones**

Cuando ya tengamos las fotos necesarias, comenzaremos a entrenar en este caso se utilizó tres algoritmos los cuales son EigenFaces, FisherFaces y LBHP. Cuando se termine el entrenamiento se guardará en la respectiva ruta que hayamos colocado en el *DataPath(línea 21)* los tres modelos entrenados.

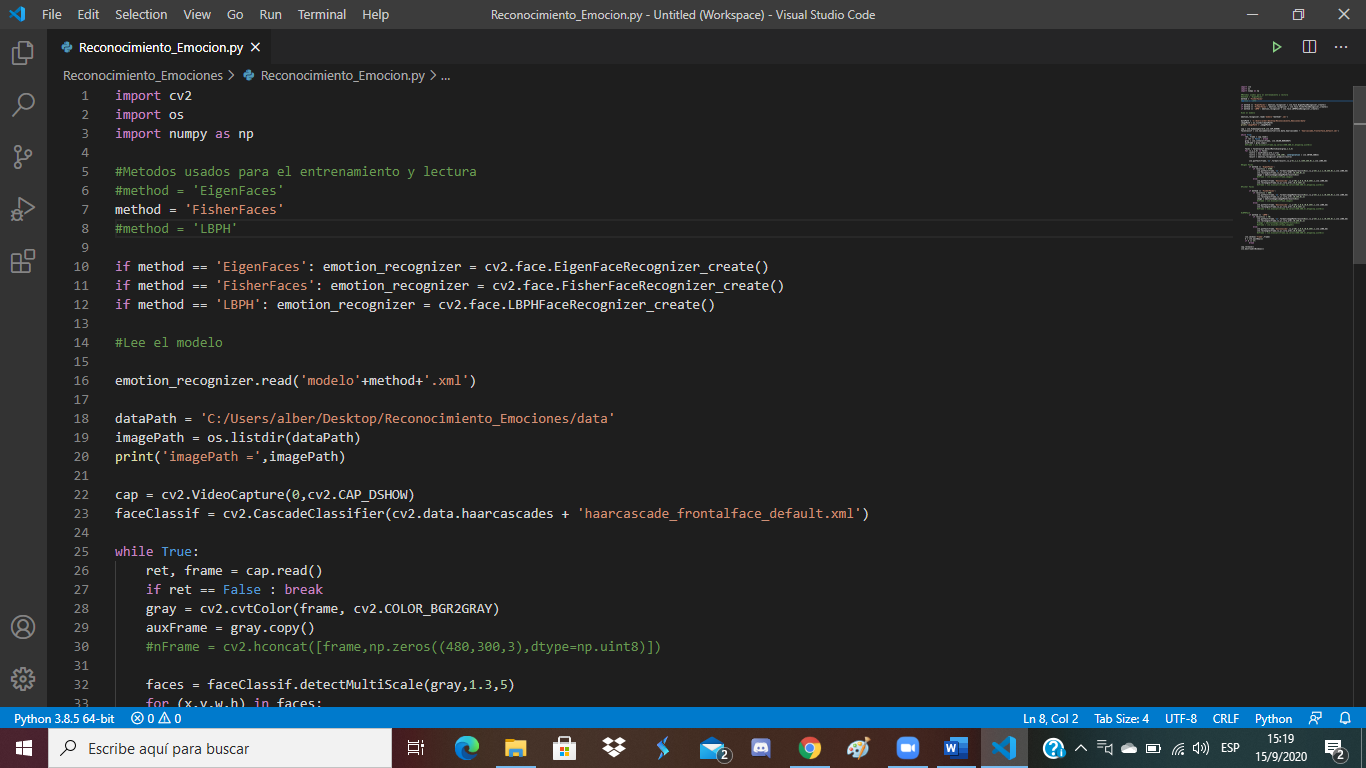


Se puede observar los tres modelos entrenados que han sido guardados como un archivo xml ya que se lo almaceno para que se guardara de esta manera como se ve claramente en la línea 19 de la imagen anterior.

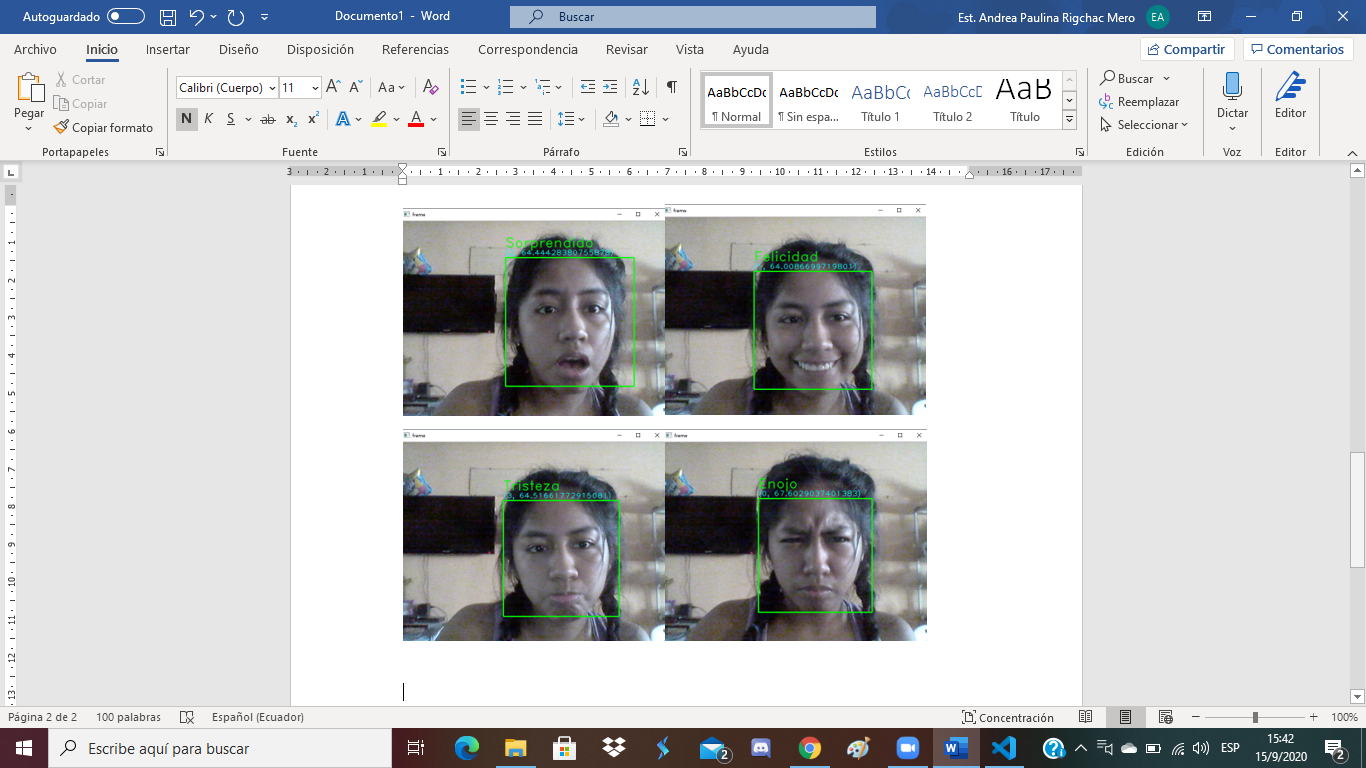


**Reconocer la Emoción**

Cuando ya tengamos los debidos modelos entrenados podemos probar con este código si nos reconoce las emociones, solamente debemos de elegir el modelo que queramos probar en este caso se usó FisherFaces y se comentó la línea 6 y 8. Solamente se debe de ejecutar para verificar si el entrenamiento funciona correctamente.



Cuando se haya ejecutado nos aparecerá un frame en el cual vamos a realizar expresiones en donde nos indicara nuestro estado de ánimo.

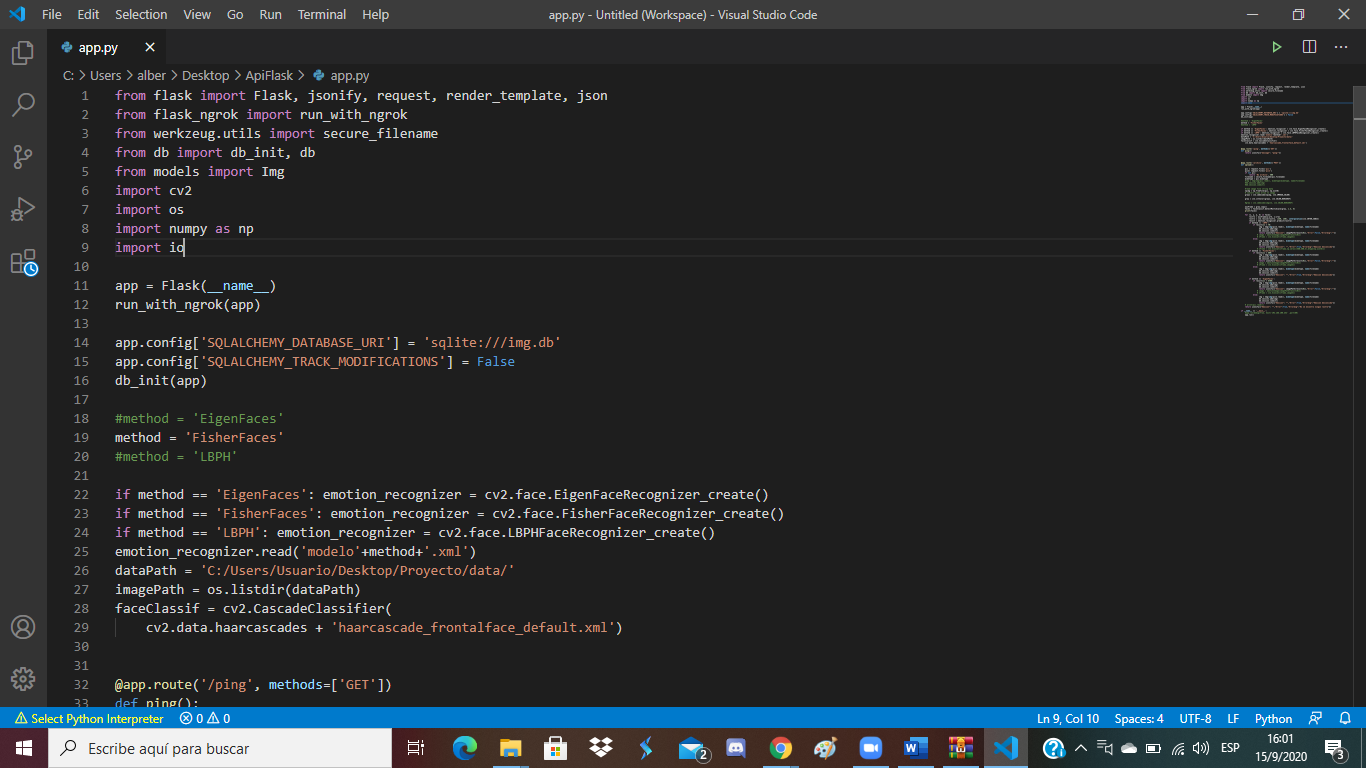


**Importar librerías o módulos para el Api**

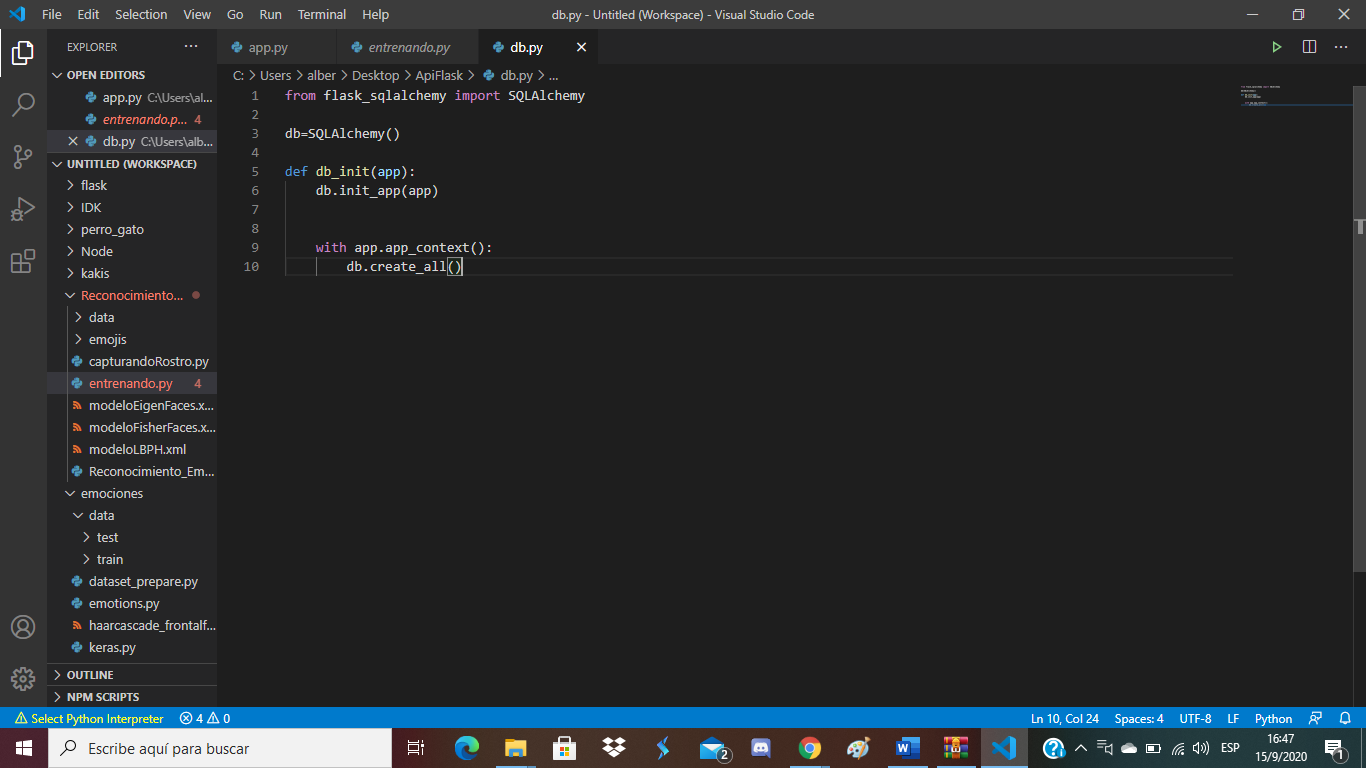
Creamos nuestro respectivo api con flask así que importamos las librerías o módulos necesarios para que funcione y se ejecute sin ningún inconveniente.

En la línea 1 esta librería nos ayudara lo que es para el servidor y a su vez para poder retornar json en nuestro api y también para poder realizar peticiones. En la línea 2 esta librería me ayudara en desplegar un servidor a una dirección publica, pero esta solamente tiene un tiempo de duración que es de 7 horas, la tercera línea me ayudara en acceder al nombre de la imagen que llegara al api. La cuarta es en donde llamaremos nuestro archivo *db.py* (Imagen 2) como se dará cuenta este archivo contiene un from de flask\_sql y este nos permite usar la base de datos con nuestro api y la línea 9 y 10 nos creara una base nueva siempre y cuando no haya una base creada.

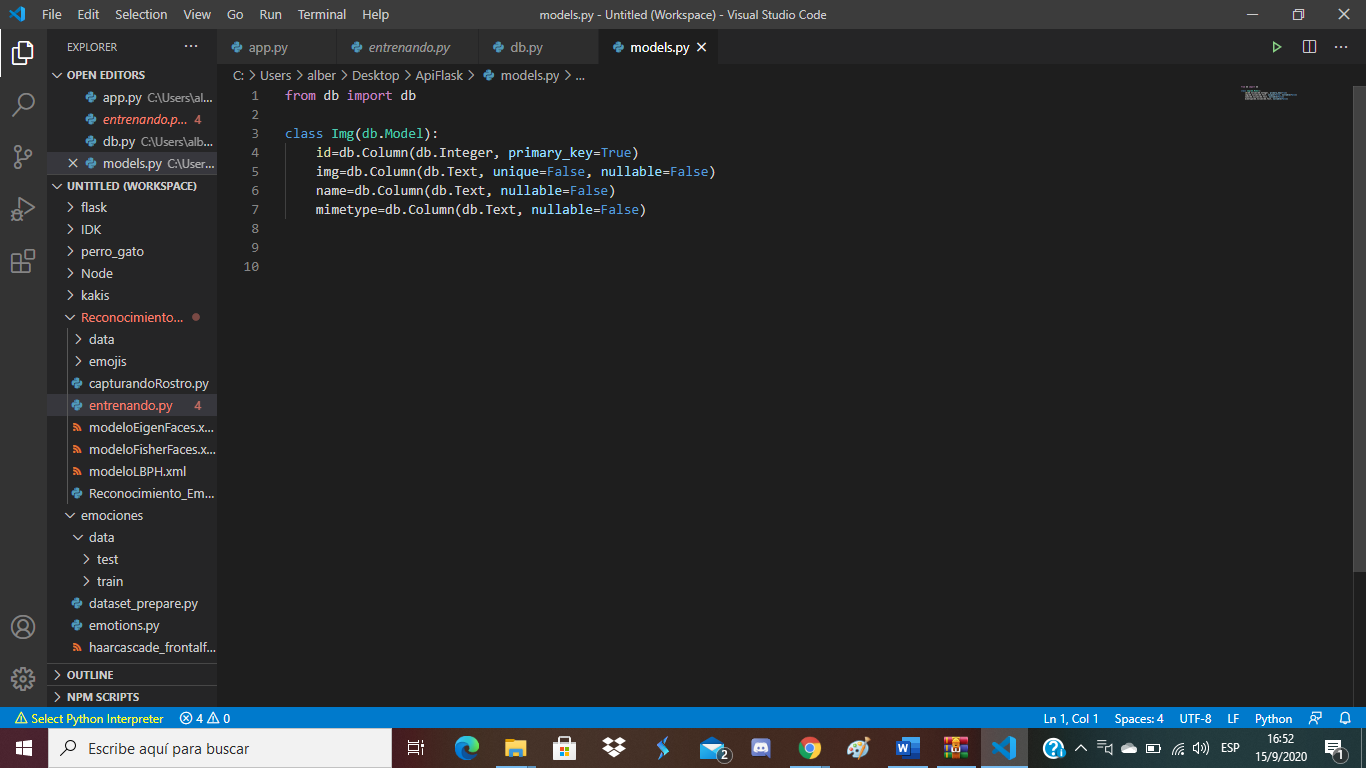
La quinta se importa *Img* del módulo *models.py*, este *Img* es en donde se guardará la imagen que llega del api con su respectivo nombre y tipo de archivo. Y por ultimo se importa cv2, os, numpy e io que son librerías que ya no conocemos.



***Imagen 1***



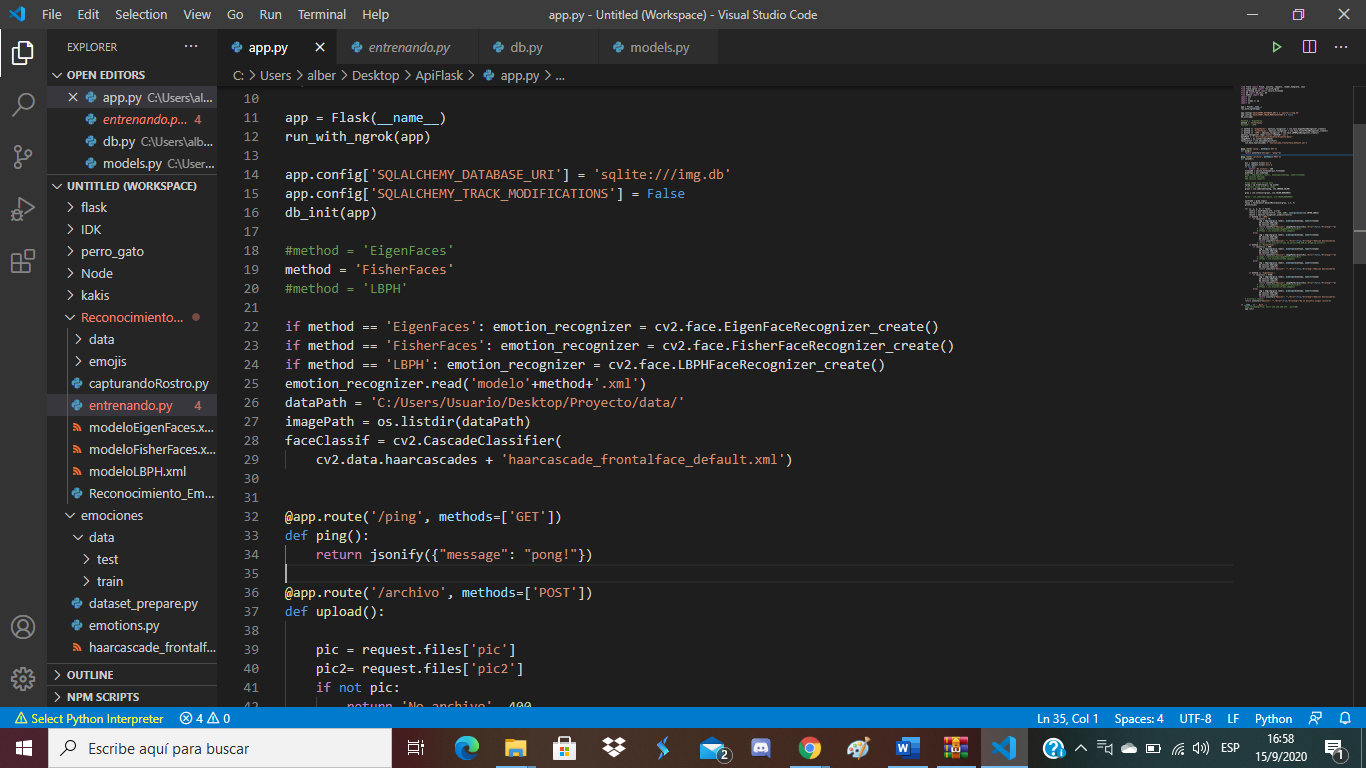
***Imagen 2***



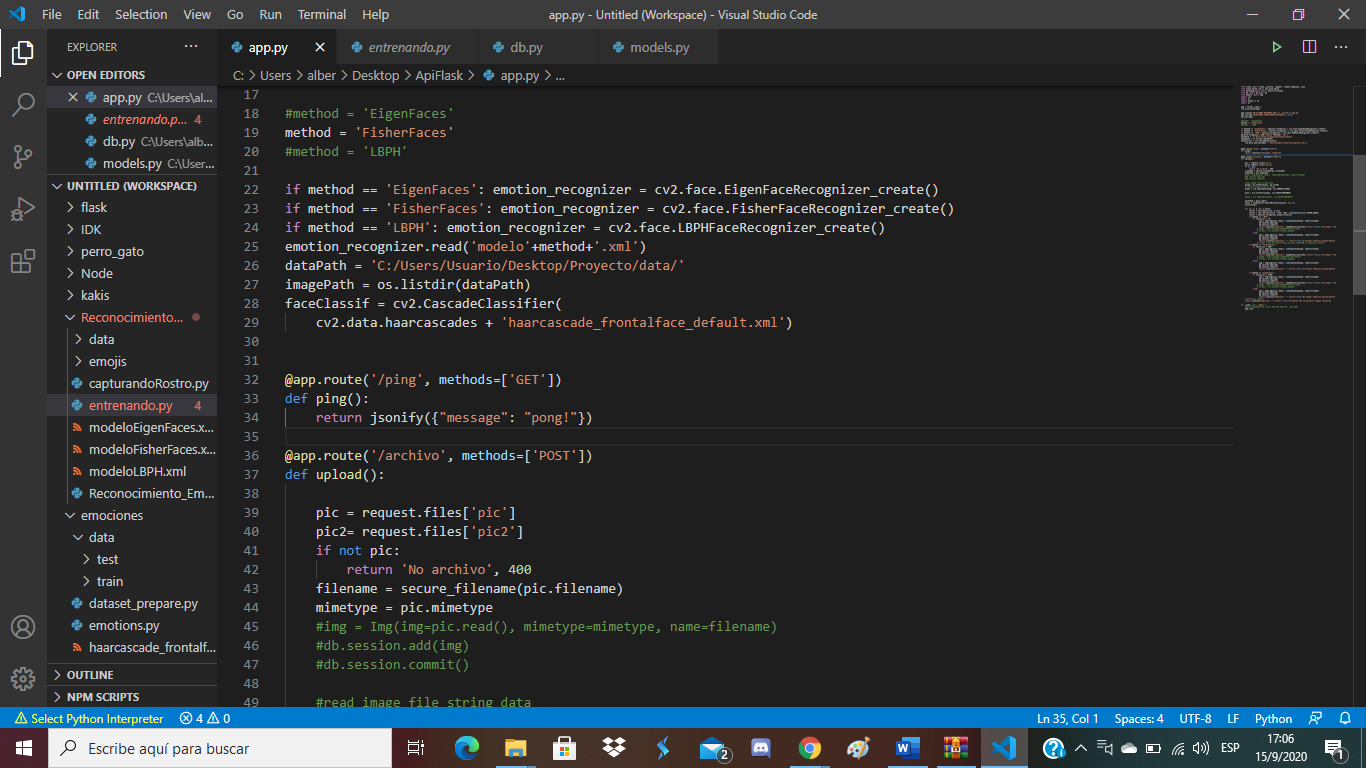
***Imagen 3***

**Crear Api con Flask**

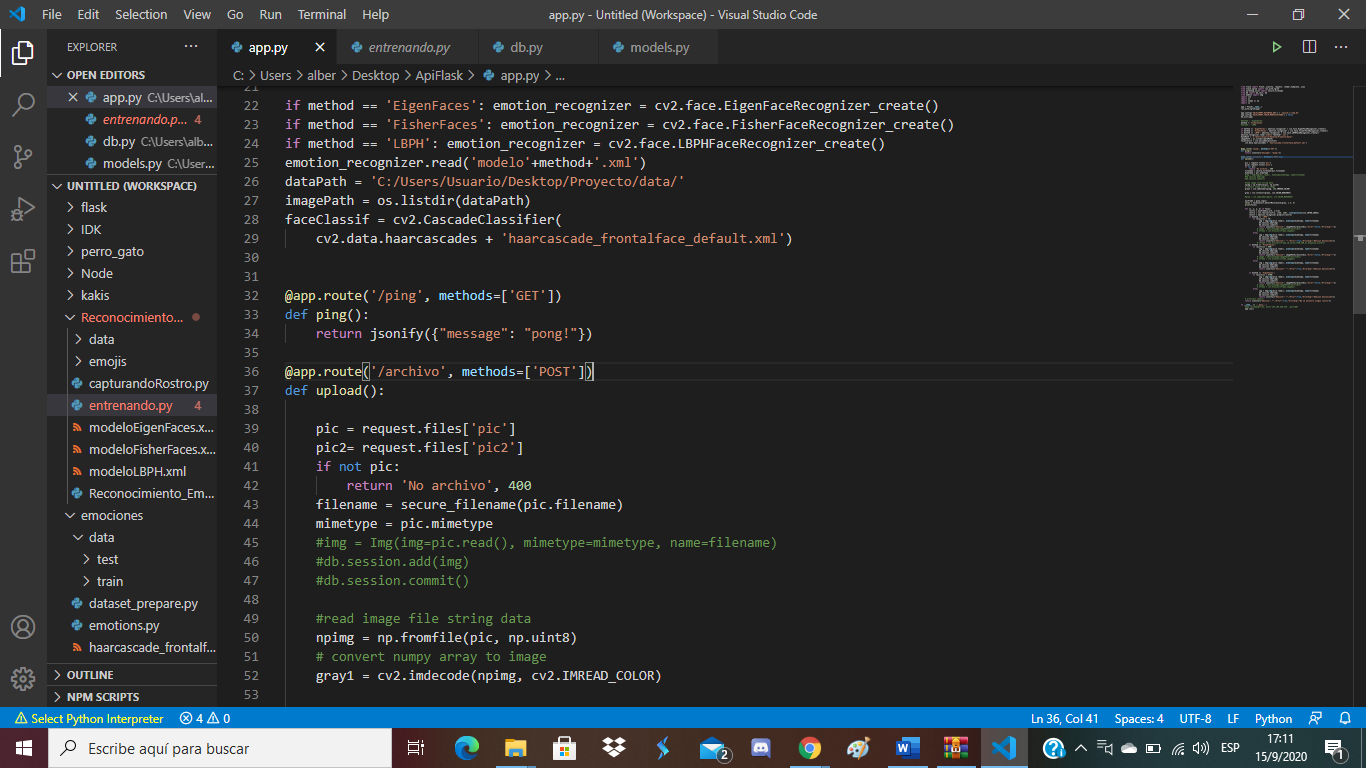
Comenzamos escribiendo lo que se ve en la línea 11, después colocaremos *run\_with\_ngrok(app)* como ya se menciono es para que se publique en una dirección publica y poder acceder a ella desde cualquier url, entre la línea 14 y 15 lo que se hace es que se configure el nombre de la base de datos sqlite y a partir de la línea 16 que es el *db\_init(app)* esta es para que la app se inicialice con la base de datos pero también antes mencionado si no hay alguna base por defecto se creara una.



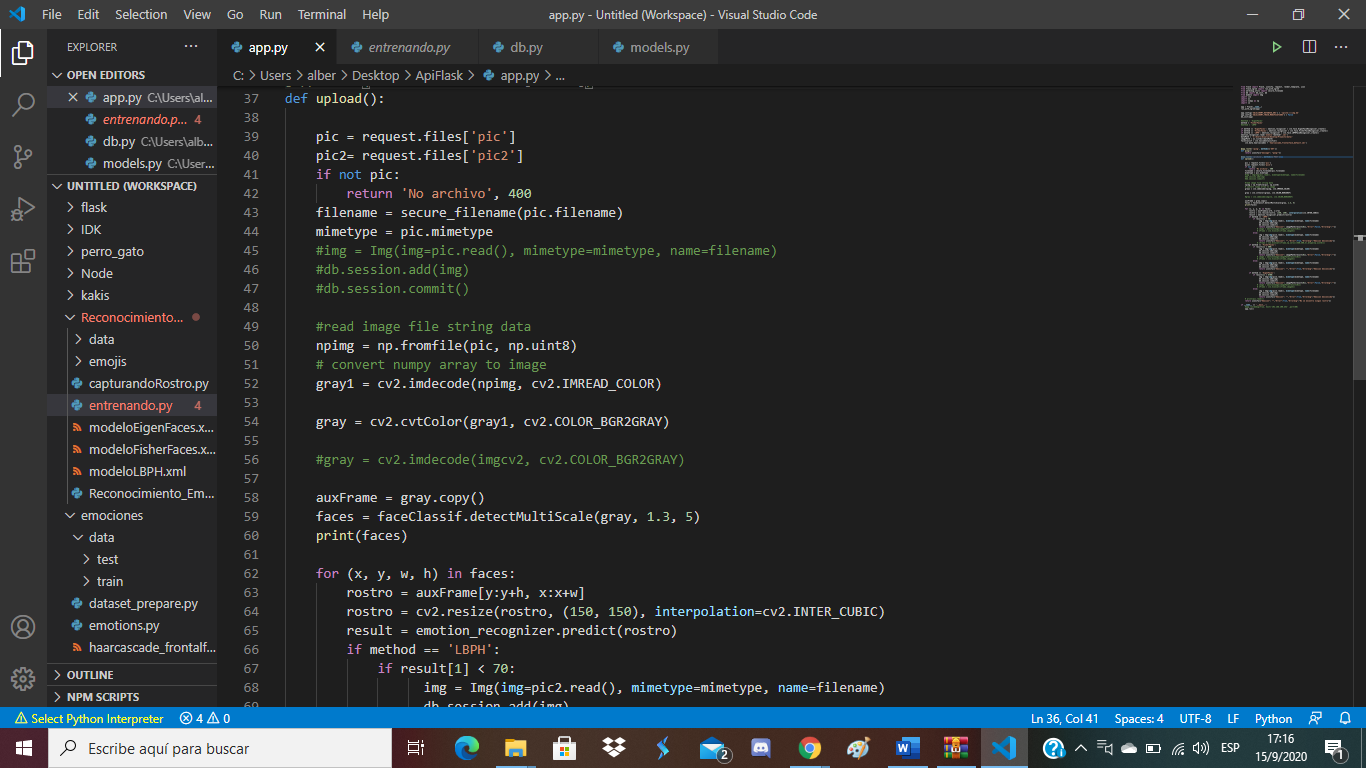
En esta parte del código se encuentra los modelos o métodos para usar en el reconocimiento de emociones pero nosotros elegimos el FisherFaces por que consideramos que es el mejor, obviamente este debe de tener la misma ruta que en el modelo entrenado como se puede ver en la linea 26 y se sabe claramente que en la carpeta data se encuentra nuestras fotos.



Como se puede observar en la imagen lo que hará la linea 32 es crear una ruta para el api en este utilizamos el método POST ya que se enviara fotos y obviamente con el método GET es para obtener.



En la línea 37 se crea una función la cual es donde llegara la imagen y retorna el json con la emoción respectiva, es decir en la línea 39 y 40 tenemos lo que es pic y pic2 es la foto que enviamos desde la aplicación a la api con secure\_filename (línea 43) es donde obtenemos el nombre del archivo, con mimetype (línea 44) obtenemos el tipo de archivo, con np.fromfile (línea 50) es donde se creara un array numpy de la imagen, en gray1 (línea 52) el arreglo se transformara en un tipo de dato cv2, en gray (línea 54) le cambiamos a escala de grises, con auxFrame (linea 58) obtendremos una copia igual que gray y finalmente con faceClassif (línea 59) es para detectar un rostro dentro del gray.



Dentro de este for lo que hacemos es recortar el auxFrame, luego redimensionamos el tamaño del rostro y se hará un predict del rostro según el método que hayamos escogido en este caso fue FisherFaces, creamos un objeto de clase *img*  y el *db.session.add*guardara un registro y luego hacemos el commit y retonarmos un json con la función *jsonify*  ya sea con la emocion, error o un mensaje de error.

