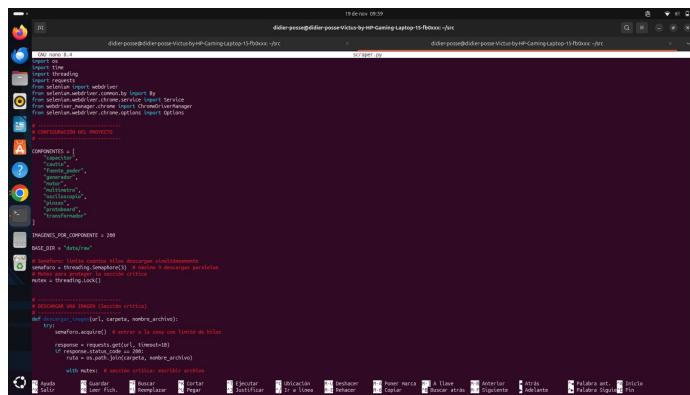


# Proyecto Tercer Corte

Didier Posse, Oscar Grande, Juan Pablo Pedraza, Marco Gomez.

November 2025

## 1 Desarrollo del primer punto.



```
didier-posse@didier-posse-Victor:~/H-Gaming/Lapto-15/Foloxo$ ./src/descarga.py
[  0%] [  1%] [  2%] [  3%] [  4%] [  5%] [  6%] [  7%] [  8%] [  9%] [ 10%] [ 11%] [ 12%] [ 13%] [ 14%] [ 15%] [ 16%] [ 17%] [ 18%] [ 19%] [ 20%] [ 21%] [ 22%] [ 23%] [ 24%] [ 25%] [ 26%] [ 27%] [ 28%] [ 29%] [ 30%] [ 31%] [ 32%] [ 33%] [ 34%] [ 35%] [ 36%] [ 37%] [ 38%] [ 39%] [ 40%] [ 41%] [ 42%] [ 43%] [ 44%] [ 45%] [ 46%] [ 47%] [ 48%] [ 49%] [ 50%] [ 51%] [ 52%] [ 53%] [ 54%] [ 55%] [ 56%] [ 57%] [ 58%] [ 59%] [ 60%] [ 61%] [ 62%] [ 63%] [ 64%] [ 65%] [ 66%] [ 67%] [ 68%] [ 69%] [ 70%] [ 71%] [ 72%] [ 73%] [ 74%] [ 75%] [ 76%] [ 77%] [ 78%] [ 79%] [ 80%] [ 81%] [ 82%] [ 83%] [ 84%] [ 85%] [ 86%] [ 87%] [ 88%] [ 89%] [ 90%] [ 91%] [ 92%] [ 93%] [ 94%] [ 95%] [ 96%] [ 97%] [ 98%] [ 99%] [ 100%]
[  0%] [  1%] [  2%] [  3%] [  4%] [  5%] [  6%] [  7%] [  8%] [  9%] [ 10%] [ 11%] [ 12%] [ 13%] [ 14%] [ 15%] [ 16%] [ 17%] [ 18%] [ 19%] [ 20%] [ 21%] [ 22%] [ 23%] [ 24%] [ 25%] [ 26%] [ 27%] [ 28%] [ 29%] [ 30%] [ 31%] [ 32%] [ 33%] [ 34%] [ 35%] [ 36%] [ 37%] [ 38%] [ 39%] [ 40%] [ 41%] [ 42%] [ 43%] [ 44%] [ 45%] [ 46%] [ 47%] [ 48%] [ 49%] [ 50%] [ 51%] [ 52%] [ 53%] [ 54%] [ 55%] [ 56%] [ 57%] [ 58%] [ 59%] [ 60%] [ 61%] [ 62%] [ 63%] [ 64%] [ 65%] [ 66%] [ 67%] [ 68%] [ 69%] [ 70%] [ 71%] [ 72%] [ 73%] [ 74%] [ 75%] [ 76%] [ 77%] [ 78%] [ 79%] [ 80%] [ 81%] [ 82%] [ 83%] [ 84%] [ 85%] [ 86%] [ 87%] [ 88%] [ 89%] [ 90%] [ 91%] [ 92%] [ 93%] [ 94%] [ 95%] [ 96%] [ 97%] [ 98%] [ 99%] [ 100%]
```

Figure 1: Proceso.

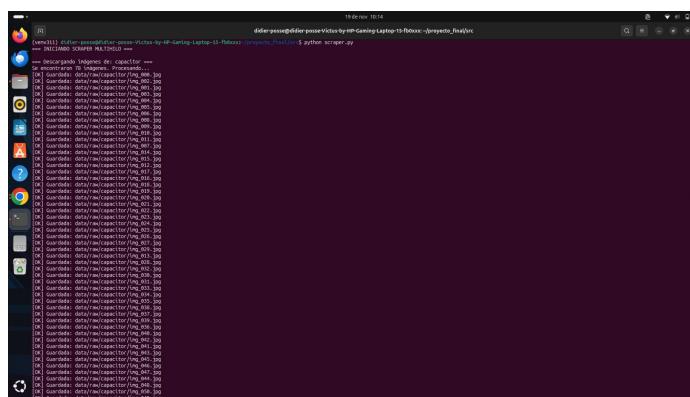


Figure 2: Imagenes

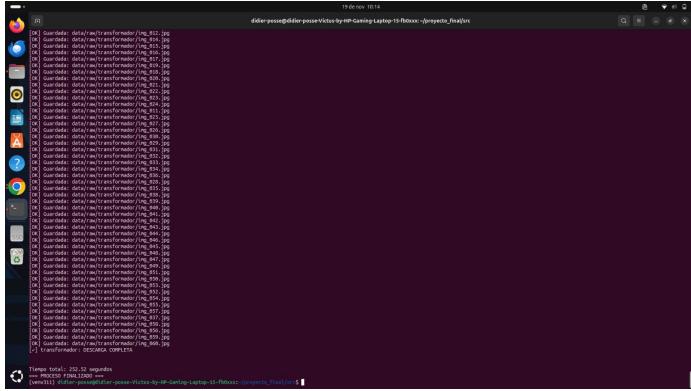


Figure 3: Imagenes

## 2 Desarrollo segundo punto.

- Como primer paso se creo una carpeta llamada "segundo-punto", con el objetivo de crear un proceso que me perminta identificar objetos electronicos y para ella se crearon archivos que se evidencian en la imagen anterior.

```
● (venv) PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> ls
Directory: C:\Users\marqu\OneDrive\Documentos\Segundo-Punto

Mode                LastWriteTime         Length Name
----                -----        -----
lar--           21/11/2025 5:53 p. m.          0 database
lar--           18/11/2025 6:37 p. m.          0 src
l----

| Mode  | LastWriteTime         | Length | Name                 |
|-------|-----------------------|--------|----------------------|
| ----  | -----                 | -----  | -----                |
| lar-- | 21/11/2025 5:53 p. m. | 0      | database             |
| lar-- | 18/11/2025 6:37 p. m. | 0      | src                  |
| l---- | 18/11/2025 6:13 p. m. | 0      | venv                 |
| la--- | 21/11/2025 6:27 p. m. | 4039   | arreglar_carpetas.py |
| la--- | 18/11/2025 6:34 p. m. | 7161   | main.py              |
| la--- | 18/11/2025 7:07 p. m. | 115    | requirements.txt     |


○ (venv) PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto>
```

Figure 4: Archivos.

En la imagen se observa que previamente se creo un entorno virtual con el comando python -m venv venv y luego usando ., lo cual se confirma porque en la terminal aparece el prefijo (venv). A continuación, se ejecuto una instrucción para instalar varias librerías de Python mediante pip install opencv-python mediapipe numpy pillow requests beautifulsoup4. Por eso la consola muestra el proceso de descarga e instalación de cada paquete, junto con todas las dependencias adicionales que estos requieren, como opencv-contrib-python, matplotlib, flatbuffers, protobuf, jax y otras. En resumen, antes se creó y activó el entorno virtual, y ahora se están instalando las librerías necesarias dentro de ese entorno, mostrando en pantalla el progreso de cada descarga e instalación.

```

PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> python -m venv venv
PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> venv\Scripts\activate
(venv) PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> pip install opencv-python mediapipe numpy pillow requests beautifulsoup4
Collecting opencv-python
  Downloading opencv_python-4.12.0.88-cp37abi3-win_amd64.whl (39.0 MB)
    ██████████ | 39.0 MB 1.1 MB/s
Collecting mediapipe
  Downloading mediapipe-0.10.11-cp38-cp38-win_amd64.whl (50.8 MB)
    ██████████ | 50.8 MB 1.1 MB/s
Collecting numpy
  Downloading numpy-1.24.4-cp38-cp38-win_amd64.whl (14.9 MB)
    ██████████ | 14.9 MB 1.1 MB/s
Collecting pillow
  Downloading pillow-10.4.0-cp38-cp38-win_amd64.whl (2.6 MB)
    ██████████ | 2.6 MB 819 kB/s
Collecting requests
  Downloading requests-2.32.4-py3-none-any.whl (64 kB)
    ██████████ | 64 kB 2.2 MB/s
Collecting beautifulsoup4
  Downloading beautifulsoup4-4.14.2-py3-none-any.whl (106 kB)
    ██████████ | 106 kB 3.3 MB/s
Collecting opencv-contrib-python
  Downloading opencv_contrib-python-4.12.0.88-cp37abi3-win_amd64.whl (45.3 MB)
    ██████████ | 45.3 MB 930 kB/s
Collecting flatbuffers>=2.0
  Downloading flatbuffers-25.0.23-py2.py3-none-any.whl (30 kB)
Collecting matplotlib
  Downloading matplotlib-3.7.5-cp38-cp38-win_amd64.whl (7.5 MB)
    ██████████ | 7.5 MB 595 kB/s
Collecting absolv
  Downloading absolv_2.3.1-py3-none-any.whl (135 kB)
    ██████████ | 135 kB 1.7 MB/s
Collecting attrs>=19.1.0
  Downloading attrs-25.3.0-py3-none-any.whl (63 kB)
    ██████████ | 63 kB 1.6 MB/s
Collecting jax
  Downloading jax-0.4.13.tar.gz (1.3 MB)
    ██████████ | 1.3 MB 1.3 MB/s
Installing build dependencies ... done
Getting requirements to build wheel ... done
  Preparing wheel metadata ... done
Collecting sounddevice>=0.4.4
  Downloading sounddevice-0.5.3-py3-none-win_amd64.whl (364 kB)
    ██████████ | 364 kB 1.6 MB/s
Collecting protobuf<4,>=3.11
  Downloading protobuf-3.20.3-cp38-cp38-win_amd64.whl (904 kB)

```

Figure 5: Entorno virtual / Descargas.

El entorno virtual (venv) ya estaba activado, por lo que todas las instalaciones se realizaron dentro de el. El sistema primero revisó cada paquete listado en requirements.txt y detecto cuáles ya estaban instalados y cuáles no. Para varios paquetes —como numpy, pillow, requests, opencv-python, mediapipe y beautifulsoup4— encontró versiones anteriores, por lo que procedió a desinstalar esas versiones viejas y luego instalar las nuevas versiones especificadas en el archivo. El proceso terminó exitosamente con la actualización e instalación de las librerías solicitadas.

```

(venv) PS E:\Users\marqu\OneDrive\Documentos\Segundo-Punto> pip install -r requirements.txt
Requirement already satisfied: processing==2.3.1 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (3.14)
Requirement already satisfied: python-dateutil<2.7 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (2.8.0.post1)
Requirement already satisfied: pytz==2020.1 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (25.0)
Requirement already satisfied: pillow==8.0.0 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (25.0)
Requirement already satisfied: mediapipe<1.0.1 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (1.4.7)
Requirement already satisfied: cyvcv==0.1.0 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (0.12.1)
Requirement already satisfied: opencv-python<4.2.0 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (4.2.0)
Requirement already satisfied: contourpy<1.0.1 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>requirements.txt (line 2)) (0.1.1)
Requirement already satisfied: importlib-resources<3.2.0 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from matplotlib->mediapipe==0.10.8>r requirements.txt (line 2)) (0.4.5)
Requirement already satisfied: zipp==3.1.0 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from importlib-resources<3.2.0>requirements.txt (line 2)) (3.1.0)
Requirement already satisfied: six<1.5 in c:\users\marqu\onedrive\documentos\segundo-punto\venv\lib\site-packages (from python-dateutil<2.7>matplotlib->mediapipe==0.10.8>r requirements.txt (line 2)) (1.17.0)
Installing collected packages: numpy, pillow, requests, opencv-python, mediapipe, beautifulsoup4
  Attempting uninstall: numpy
    Found existing installation: numpy 1.24.4
    Uninstalling numpy-1.24.4:
      Successfully uninstalled numpy-1.24.4
  Attempting uninstall: pillow
    Found existing installation: pillow 10.4.0
    Uninstalling pillow-10.4.0:
      Successfully uninstalled pillow-10.4.0
  Attempting uninstall: requests
    Found existing installation: requests 2.32.4
    Uninstalling requests-2.32.4:
      Successfully uninstalled requests-2.32.4
  Attempting uninstall: opencv-python
    Found existing installation: opencv-python 4.12.0.88
    Uninstalling opencv-python 4.12.0.88:
      Successfully uninstalled opencv-python-4.12.0.88
  Attempting uninstall: mediapipe
    Found existing installation: mediapipe 0.10.11
    Uninstalling mediapipe-0.10.11:
      Successfully uninstalled mediapipe-0.10.11
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.14.2
    Uninstalling beautifulsoup4-4.14.2:
      Successfully uninstalled beautifulsoup4-4.14.2
Successfully installed beautifulsoup4-4.14.2 mediapipe-0.10.8 numpy-1.24.4 opencv-python-4.12.0.88 pillow-10.4.0 requests-2.31.0
WARNING: You are using pip version 21.1.1; however, version 25.0.1 is available.
You should consider upgrading via the C:\Users\marqu\OneDrive\Documentos\Segundo-Punto\venv\scripts\python.exe -m pip install --upgrade pip command.

(venv) PS E:\Users\marqu\OneDrive\Documentos\Segundo-Punto>

```

Figure 6: Instalación de librerías.

La imagen muestra la ejecución exitosa del script arreglar-carpetas.py dentro de un entorno virtual de Python. El programa verifica la estructura de carpetas necesarias para un proyecto de clasificación de imágenes, confirmando la existencia de diez categorías como capacitor, cautín, fuente-poder, generador, motor, multímetro, osciloscopio, pinzas, protoboard y transformador. Para cada una, se indica la cantidad de imágenes encontradas y si cumplen con el mínimo requerido. En la sección Estado Actual, se listan nuevamente las carpetas junto con el número de imágenes en cada una, destacando que fuente-poder y generador aún necesitan mas archivos. Finalmente, el script concluye con un mensaje indicando que la estructura ha sido corregida y que ya se puede ejecutar el archivo principal con el comando python main.py.

```

● (venv) PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> python arreglar_carpetas.py
=====
` ARREGLANDO ESTRUCTURA DE CARPETAS
=====

■ Verificando carpetas necesarias...
✓ capacitor          (7 imágenes)
✓ cautín              (9 imágenes)
✓ fuente_poder        (4 imágenes)
✓ generador           (4 imágenes)
✓ motor               (9 imágenes)
✓ multímetro          (8 imágenes)
✓ osciloscopio         (11 imágenes)
✓ pinzas              (9 imágenes)
✓ protoboard          (12 imágenes)
✓ transformador        (9 imágenes)

=====
ESTADO ACTUAL
=====
 7 imágenes          Capacitor
 9 imágenes          Cautín
 4 imágenes (necesitas más) Fuente Poder
 4 imágenes (necesitas más) Generador
 9 imágenes          Motor
 Multímetro
 Osciloscopio
 Pinzas
 Protoboard
 Transformador

Total: 82 imágenes
 ¡Todas las carpetas tienen imágenes!

=====
⚠ Estrutura corregida. Ahora puedes ejecutar:
python main.py
=====
```

Figure 7: Ejecución de la carpeta arreglo.

La imagen muestra la ejecución del archivo main.py, el cual despliega el menú principal del Sistema Clasificador de Equipos de Laboratorio. El programa presenta una interfaz en consola organizada y clara, donde se ofrecen distintas opciones numeradas para interactuar con el sistema. Entre las funciones disponibles se encuentran: configurar la base de datos mediante web scraping, entrenar o cargar el clasificador, clasificar una imagen, y visualizar el estado actual del sistema. Además, incluye una opción futura para clasificar imágenes directamente desde la webcam, indicada como “Próximamente”. Finalmente, también se muestra la opción para salir del sistema. Todo el menú está rodeado por líneas decorativas que resaltan el título y la organización del contenido.

```

● (venv) PS C:\Users\marqu\OneDrive\Documentos\Segundo-Punto> python main.py
=====
 SISTEMA CLASIFICADOR DE EQUIPOS DE LABORATORIO
=====

1.  Configurar Base de Datos (Web Scraping)
2.  Entrenar/Cargar Clasificador
3.  Clasificar una Imagen
4.  Ver Estado del Sistema
5.  Clasificar desde Webcam (Próximamente)
6.  Salir
```

Figure 8: clasificador

```
Opción 1: Buscar imágenes en Google
1. Busca cada equipo en Google Imágenes
2. Descarga 5-10 imágenes de cada uno
3. Guárdalas en: database/images/[nombre_equipo]/
```

Equipos a buscar:

- Capacitor  
Carpeta: database/images/capacitor/
- Cautín  
Carpeta: database/images/cautin/
- Fuente Poder  
Carpeta: database/images/fuente\_poder/
- Generador  
Carpeta: database/images/generador/
- Motor  
Carpeta: database/images/motor/
- Multímetro  
Carpeta: database/images/multimetro/
- Osciloscopio  
Carpeta: database/images/osciloscopio/
- Pinzas  
Carpeta: database/images/pinzas/
- Protoboard  
Carpeta: database/images/protoboard/
- Transformador  
Carpeta: database/images/transformador/

Opción 2: Tomar fotos propias

1. Ve al laboratorio de tu universidad
2. Toma fotos de los equipos disponibles
3. Organízalas en las carpetas correspondientes

```
=====
 Estado de la Base de Datos:
-----
✓ Capacitor: 7 imágenes
✓ Cautín: 9 imágenes
✓ Fuente Poder: 4 imágenes
✓ Generador: 4 imágenes
✓ Motor: 9 imágenes
✓ Multímetro: 8 imágenes
✓ Osciloscopio: 11 imágenes
✓ Pinzas: 9 imágenes
✓ Protoboard: 12 imágenes
✓ Transformador: 9 imágenes
-----
Total: 82 imágenes
```

Figure 9: creacion base de datos

el sistema ofrece dos formas de crear la base de datos de imágenes: buscarlas en Google y guardarlas en sus carpetas correspondientes, o tomar fotos propias en el laboratorio y organizarlas igual. Se muestran las carpetas creadas para cada equipo (capacitor, cautín, fuente de poder, motor, multímetro, etc.) junto con la cantidad de imágenes disponibles, sumando un total de 82 imágenes actualmente en la base de datos.

Entrenar/Cargar Clasificador; El sistema muestra el proceso de carga del clasificador después de seleccionar la opción 2 del menú. Primero se inicia la carga del modelo y luego se inicializa el clasificador. A continuación, el programa procede a cargar la base de datos de imágenes, mostrando una confirmación por cada categoría detectada, como capacitor, cautín, fuente de poder,

generador, motor, multímetro, oscilloscopio, pinzas, protoboard y transformador. Una vez completado este proceso, el sistema confirma que las 10 categorías fueron cargadas correctamente y que el clasificador quedó listo para usarse.

```
=====
Elige una opción (1-6): 2
🕒 Cargando clasificador...
🕒 Inicializando clasificador...

📁 Cargando base de datos...
✓ Cargada categoría: Capacitor
✓ Cargada categoría: Cautín
✓ Cargada categoría: fuente_poder
✓ Cargada categoría: generador
✓ Cargada categoría: motor
✓ Cargada categoría: multímetro
✓ Cargada categoría: oscilloscopio
✓ Cargada categoría: pinzas
✓ Cargada categoría: Protoboard
✓ Cargada categoría: Transformador

✓ Base de datos cargada: 10 categorías
✓ Clasificador cargado exitosamente
    Categorías disponibles: 10

Presiona Enter para continuar...
=====
```

Figure 10: opción 2

Reconocimiento de objetos; La imagen muestra el proceso de clasificación de una herramienta seleccionada desde la base de datos, donde el usuario elige la categoría “Generador” y selecciona una de las imágenes disponibles. Cabe resaltar que se hizo la prueba con los otros objetos como evidencia y se compartirán posteriormente.

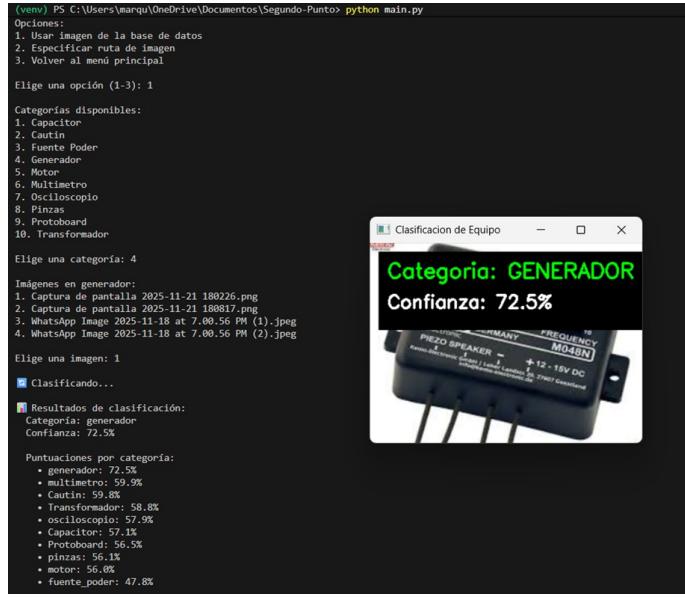


Figure 11: Reconocimiento.



Figure 12: objeto detectado.

Estado del sistema; En esta imagen se muestra un resumen del estado completo del sistema: primero aparece el conteo de imágenes por cada categoría dentro de la base de datos, indicando que el sistema tiene un total de 82 imágenes distribuidas en 10 clases. Luego se observa el estado del clasificador, que comienza su inicialización y procede a cargar nuevamente todas las categorías, confirmando que las 10 se han cargado correctamente. Finalmente, el sistema reporta que todo funciona bien y muestra la lista de categorías disponibles, verificando que el clasificador está operativo y listo para usar.

```
Elige una opción (1-6): 4
[ESTADO DEL SISTEMA
=====
[ Estado de la Base de Datos:
-----
✓ Capacitor: 7 imágenes
✓ Cautin: 9 imágenes
✓ Fuente Poder: 4 imágenes
✓ Generador: 4 imágenes
✓ Motor: 9 imágenes
✓ Multímetro: 8 imágenes
✓ Osciloscopio: 11 imágenes
✓ Pinzas: 9 imágenes
✓ Protoboard: 12 imágenes
✓ Transformador: 9 imágenes
-----
Total: 82 imágenes
[ Estado del Clasificador:
[ Inicializando clasificador...
[ Cargando base de datos...
✓ Cargada categoría: Capacitor
✓ Cargada categoría: Cautin
✓ Cargada categoría: fuente_poder
✓ Cargada categoría: generador
✓ Cargada categoría: motor
✓ Cargada categoría: multímetro
✓ Cargada categoría: osciloscopio
✓ Cargada categoría: pinzas
✓ Cargada categoría: Protoboard
✓ Cargada categoría: Transformador
-----
✓ Base de datos cargada: 10 categorías
✓ Funcionando correctamente
✓ Categorías cargadas: 10
• Capacitor
• Cautin
• Fuente Poder
• Generador
• Motor
• Multímetro
• Osciloscopio
• Pinzas
• Protoboard
• Transformador
```

Figure 13: Funcionamiento.

### 3 Desarrollo punto 3.

Creación del entorno con mediapipe.

```
(deteccción) juan-pablo-pedraza@juan-pablo-pedraza-Vivobook-ASUSLaptop-X1605VA:~/detección_vocales$ conda create -n detección python3.11 -y
channel Terms of Service accepted
Current channel: defaults
Platform: linux-64
Fetching package metadata (repodata.json): done
Solving environment: done
## Package Plan ##

environment locations: /home/juan-pablo-pedraza/miniconda3/envs/detección

added / updated specs:
  - python3.11

The following packages will be downloaded:
  packages | build
  libgcc-11.2.0 | h969729_7    896 kB
  libgcc-11.2.0 | h8994729_7    85 kB
  libgcc-11.2.0 | h475f7c2_7    436 kB
  libatexx-1.2.0 | h3973987_7    3.7 MB
  libyaml-0.2.0 | h8994729_7    83 kB
  pip-21.3       | pybc87213_0    1.1 MB
  pip-21.3       | pybc87213_0    23 kB
  setuptools-61.14 | py311h66a4398_0    1.9 MB
  setuptools-61.14 | py311h66a4398_0    151 kB
  wheel-0.45.1   | py311h66a4398_0    151 kB
Total:          37.9 MB

The following NEW packages will be INSTALLED:
  _libgcc_mutex  pkg/main/linux-64/_libgcc_mutex-0.1.mzn
  _openmp_mutex  pkg/main/linux-64/_openmp_mutex-5.1.1.mzn
  libgcc         pkg/main/linux-64/libgcc-11.2.0-h969729_7
  ca-certificates  pkg/main/linux-64/ca-certificates-2025.11.4-h6aa4398_0
  libgcc-11.2.0 | h969729_7    896 kB
  libgcc-11.2.0 | h8994729_7    85 kB
  libgcc-11.2.0 | h475f7c2_7    436 kB
  libatexx-1.2.0 | h3973987_7    3.7 MB
  libyaml-0.2.0 | h8994729_7    83 kB
  pip-21.3       | pybc87213_0    1.1 MB
  pip-21.3       | pybc87213_0    23 kB
  setuptools-61.14 | py311h66a4398_0    1.9 MB
  setuptools-61.14 | py311h66a4398_0    151 kB
  wheel-0.45.1   | py311h66a4398_0    151 kB
  ncurses        pkg/main/linux-64/ncurses-8.1.2-h337f514_2
```

Figure 14: Creación de entorno.

Instalación de dependencias necesarias para el funcionamiento de mediapipe.

```
(detección) juan-pablo-pedraza@juan-pablo-pedraza-Vivobook-ASUSLaptop-X1605VA-X1605VA:~/detección_vocales$ python -c "import mediapipe as mp; print('MediaPipe OK')"
MediaPipe OK
OpenCV: 4.12.0
```

Figure 15: Instalar dependencias.

Creación del código para el aprendizaje de detección con las imágenes descargadas y seleccionadas en los anteriores pasos.

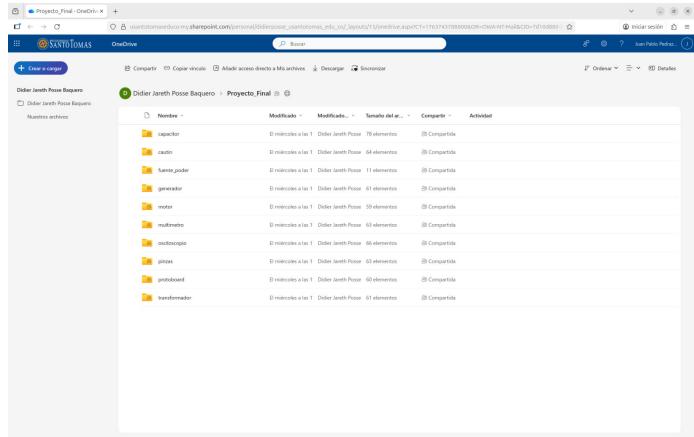


Figure 16: imágenes

A continuación se mostraran los directorios con los archivos necesarios y ya ubicados. Para la visualización de estos puede hacerlo directamente el el repositorio donde están separados por puntos a desarrollar.

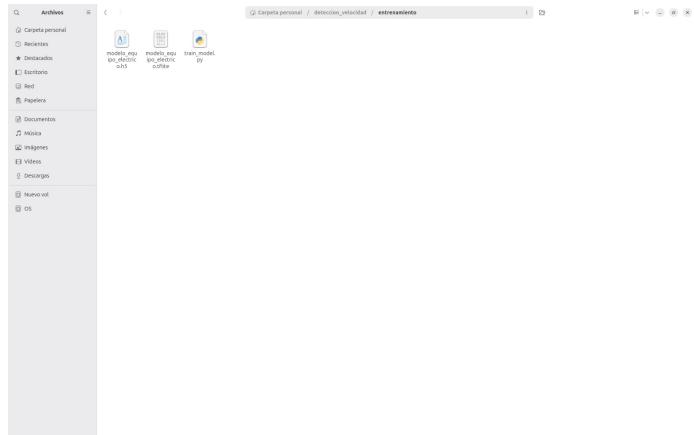


Figure 17: Archivos.

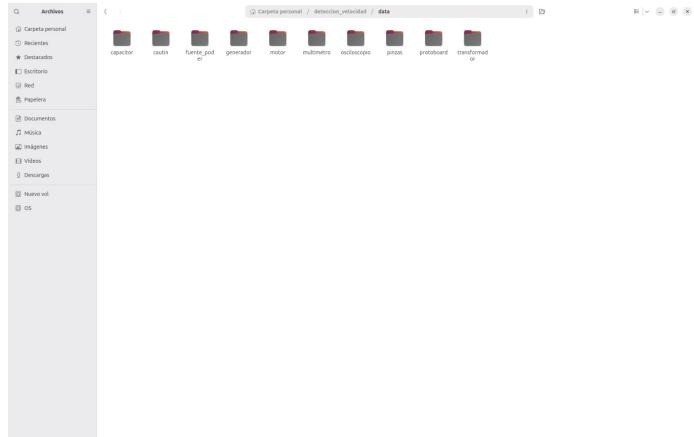


Figure 18: Directories.



Figure 19: Archivos.py

Evidencia de funcionamiento.

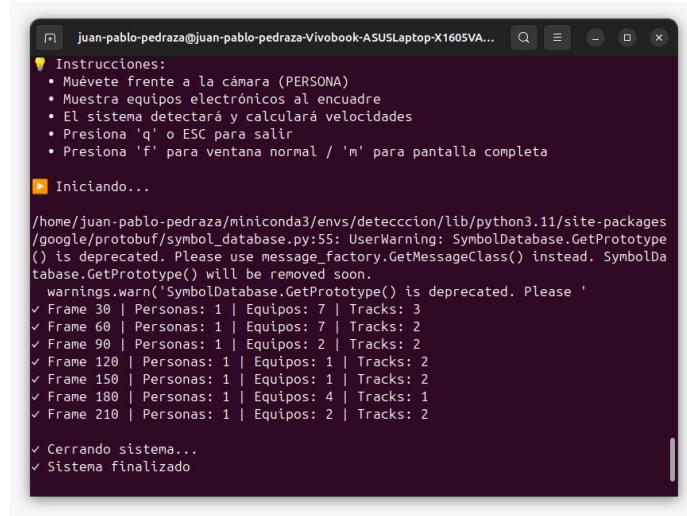
```
✓ MediaPipe inicializado
  SpeedTracker inicializado (escala: 40 px/m)

✓ Sistema inicializado correctamente

=====
INICIANDO DETECCION
=====

  Abriendo cámara: 0
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
W0000 00:00:1763742161.777674    35239 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1763742161.794898    35255 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
✓ Cámara abierta
```

Figure 20: Funcionamiento



The screenshot shows a terminal window titled 'juan-pablo-pedraza@juan-pablo-pedraza-Vivobook-ASUSLaptop-X1605VA...'. It displays the following text:

- Yellow lightbulb icon: Instrucciones:
  - Muévete frente a la cámara (PERSONA)
  - Muestra equipos electrónicos al encuadre
  - El sistema detectará y calculará velocidades
  - Presiona 'q' o ESC para salir
  - Presiona 'f' para ventana normal / 'm' para pantalla completa
- Yellow play button icon: Iniciando...
- Terminal output:

```
/home/juan-pablo-pedraza/miniconda3/envs/deteccion/lib/python3.11/site-packages/google/protobuf/symbol_database.py:55: UserWarning: SymbolDatabase.GetPrototype() is deprecated. Please use message_factory.GetMessageClass() instead. SymbolDatabase.GetPrototype() will be removed soon.  
  warnings.warn('SymbolDatabase.GetPrototype() is deprecated. Please '  
✓ Frame 30 | Personas: 1 | Equipos: 7 | Tracks: 3  
✓ Frame 60 | Personas: 1 | Equipos: 7 | Tracks: 2  
✓ Frame 90 | Personas: 1 | Equipos: 2 | Tracks: 2  
✓ Frame 120 | Personas: 1 | Equipos: 1 | Tracks: 2  
✓ Frame 150 | Personas: 1 | Equipos: 1 | Tracks: 2  
✓ Frame 180 | Personas: 1 | Equipos: 4 | Tracks: 1  
✓ Frame 210 | Personas: 1 | Equipos: 2 | Tracks: 1  
  
✓ Cerrando sistema...  
✓ Sistema finalizado
```

Figure 21: Funcionamiento.



Figure 22: Juan Pablo

### 3.1 Despliegue Streamlit.

Para el despliegue del proyecto en streamlit, primero optamos por tener orden y conocer la lógica del programa, una vez eso crearemos el archivo Dockerfile listo para construir la imagen, acompañado de el archivo requirements.txt con todas las librerías necesarias para el funcionamiento del programa y el codigo del dashboard con streamlit.

Primero creamo el entorno virtual de la siguiente manera:

```
[interno] user@grandesocar-grande-MacBook-Air:~/Proyecto$ python3 -m venv entorno
[entorno] user@grandesocar-grande-MacBook-Air:~/Proyecto$ source entorno/bin/activate
```

Figure 23: Creacion del entorno.

luego lo que haremos es crear la imagen.

```
(entorno) user@grandesocar-grande-MacBook-Air:~/Proyecto$ docker build -t tu_usuario/lab-detector:latest .
[+] Building 214.6s (3/5)
   .> [internal] transfer dockerfile: 40B
   .> [internal] transfer context: 4.99MB
   .> [internal] load dangling
   .> [internal] load reference
   .> [internal] load manifest
   .> [internal] push blob: 3.09GB
   .> [internal] push manifest: 3.09GB
   .> [internal] push tag: lab-detector:latest
[entorno] user@grandesocar-grande-MacBook-Air:~/Proyecto$ docker run -it tu_usuario/lab-detector:latest
[entorno] user@grandesocar-grande-MacBook-Air:~/Proyecto$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
327f79e33f4d        tu_usuario/lab-detector:latest   "streamlit run app.py"   21 seconds ago     Up 21 seconds          0.0.0.0:8501->8501/tcp   lab-detector
[entorno] user@grandesocar-grande-MacBook-Air:~/Proyecto$ curl http://localhost:8501
[entorno] user@grandesocar-grande-MacBook-Air:~/Proyecto$
```

Figure 24: Construccion de la imagen

Construida la imagen, podremos ingresar al URL del streamlit el cual se ejecuta en el puerto 8501, pruebas:

Figure 25: Imagen construida

Para correr la imagen

```
(Info) oscar-gradebebaco-grande-Nitro-ANV15-01z ->/projectofiles S docker run -it -rm -p 8501:8501 ts_usuario/lab-detector
Collecting usage statistics. To deactivate, set browser.getUsageStats to False.

You can now view your Streamlit app in your browser:
  Networks (URL): http://172.17.0.1:8501
  External URL: http://201.234.142.194:8501
```

Figure 26: ejecutar la imagen

Ingresamos al link que nos da directamente la terminal para que abra el dashboard directamente en su navegador.

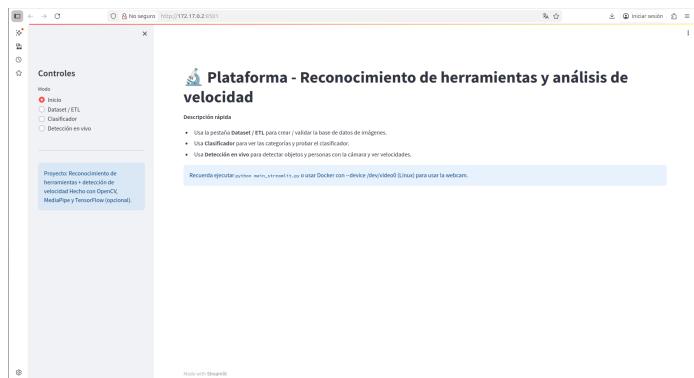


Figure 27: Dashboard.

Acá directamente en el dashboard puede seguir las instrucciones para el funcionamiento del dashboard.



Figure 28: Dataset/ ETL.

Como se puede evidenciar en la imagen donde puede confirmar el estado de la base de datos o descargar imágenes ejemplos de esta misma base de datos.

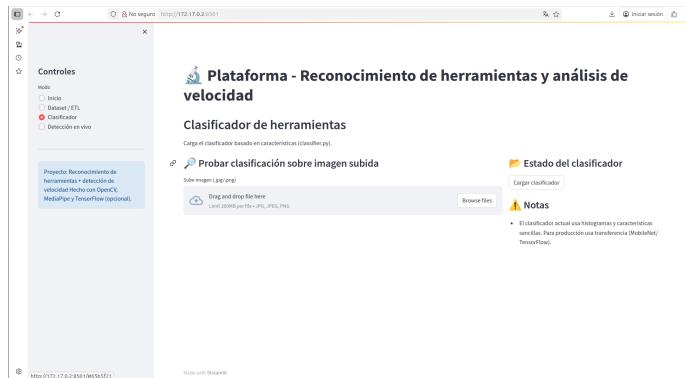


Figure 29: Reconocimiento de herramientas

En esta sección podrá subir alguna imagen compatible para desarrollar su respectivo reconocimiento de la herramienta.



Figure 30: Detección en vivo.

Acá debe seguir las recomendaciones del dashboard para poder ejecutar la cámara y poder hacer la detección en vivo mediante la cámara directamente de su equipo.

Comprobado el funcionamiento del programa, lo que se procede hacer es subir la imagen construida a Docker-hub por ello se comparte la imagen del procedimiento para realizar el respectivo proceso:

Figure 31: Proceso Docker-hub

Una vez completado el proceso podemos validar la presencia del proyecto en docker-hub lo cual quiere decir que el proceso de subir el proyecto a docker-hub se ha culminado y fue acertado.

```
[Enter] user@oscar:~/Desktop$ docker push oscargrande08/lab-detector:latest
The push refers to repository [docker.io/oscargrande08/lab-detector]
4294967295: Pushed
7add68152962: Pushed
e496c995d0c: Pushed
4294967295: Pushed
00a17959f180: Pushed
00a17959f180: Pushed
381653e33245: Mounted from library/python
a1e9b0a847bc: Mounted from library/python
a4243a2a2300: Mounted from library/python
c266bc8af06: Mounted from library/python
a3333a2a2300: Mounted from library/python
4c8d11a7ff00: Mounted from library/python
4c8d11a7ff00: Mounted from library/python
latest: digest: sha256:c2aeb5fc0d540fe20857216ab0d3020fb522515872a43221730ec6df1773f5 size: 3058
[Enter] user@oscar:~/Desktop$
```

Figure 32: Proceso culminado.

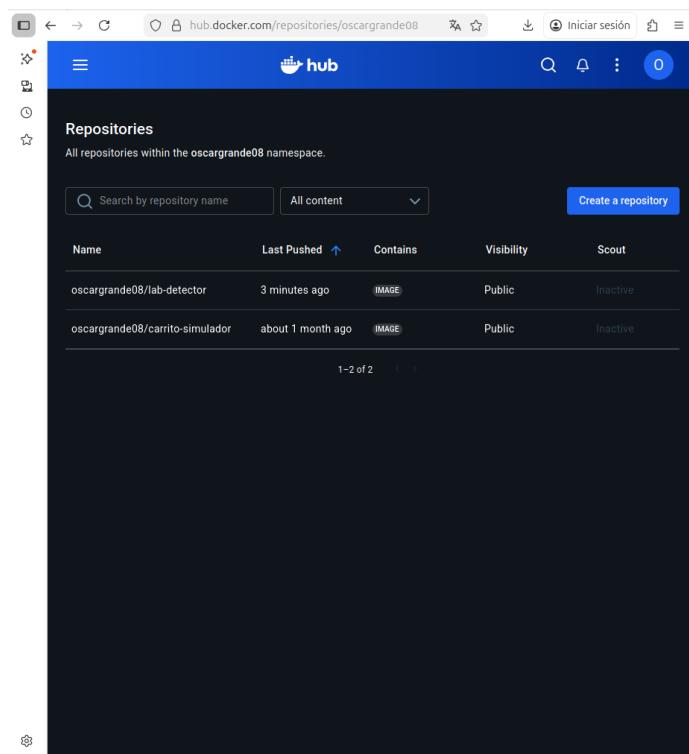


Figure 33: Docker-hub.