

# Dataset annotation

---

## Dataset annotation

1. Dataset collection
  - 1.1. Extract images
  - 1.2. Dataset path
2. Label Studio usage
  - 2.1. Label Studio installation
  - 2.2. Launch Label Studio
    - 2.2.1. Shared folder permissions
    - 2.2.2. Start Label Studio
    - 2.2.3. Access Label Studio
      - Host
      - Same LAN
      - First time use
  3. Dataset labeling
    - 3.1. Create a project
    - 3.2. Project name
    - 3.3. Import training set
    - 3.4. Label setting
    - 3.5. Dataset Annotation
    - 3.6. Export dataset
    - 3.7. Dataset directory framework
    - 3.8. Dataset configuration file

References

To identify specific objects or improve the accuracy of object recognition, users need to train the model themselves, and the collection and annotation of data sets are an indispensable part.

## 1. Dataset collection

---

Description: For recognition in specific situations, users can use a camera to record videos of objects from various angles, and then capture pictures from the video as training sets

### 1.1. Extract images

Provide a simple example: extract a frame of image from the video every 15 frames as a data set

```
import cv2
import os

# Path to the input video file
video_file =
'/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_video/orange.mkv'

# Path to the output folder to save the frames
output_folder =
'/home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_pic'

# Check if the output folder exists, if not, create it
if not os.path.exists(output_folder):
    os.makedirs(output_folder)
```

```

# Open the video file using OpenCV
cap = cv2.VideoCapture(video_file)

# Get the total number of frames in the video
total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

# Get the frames per second (fps) of the video
fps = cap.get(cv2.CAP_PROP_FPS)

# Frame counter to keep track of the current frame number
frame_number = 0

# Define the interval for extracting frames (every 15th frame)
frame_interval = 15

# Initialize a new counter for saved frames (starting from 1)
saved_frame_number = 1

while True:
    ret, frame = cap.read()

    # If the frame is not successfully read, exit the loop
    if not ret:
        break

    # Save every 15th frame
    if frame_number % frame_interval == 0:
        # Format the frame number starting from 1 (e.g., 1.png, 2.png)
        image_name = f'{saved_frame_number}.png'
        image_path = os.path.join(output_folder, image_name)

        # Save the current frame as a PNG image
        cv2.imwrite(image_path, frame)
        print(f'Saving frame {frame_number}/{total_frames} as {image_name}')

    # Increment the saved frame counter
    saved_frame_number += 1

    # Increment the frame counter for the video
    frame_number += 1

# Release the video capture object
cap.release()

# Print message when processing is complete
print("Video processing complete! Every 15th frame saved as an image!")

```

## 1.2. Dataset path

Sample video: /home/jetson/ultralytics/ultralytics/data/yahboom\_data/orange\_video

Sample image: /home/jetson/ultralytics/ultralytics/data/yahboom\_data/orange\_pic

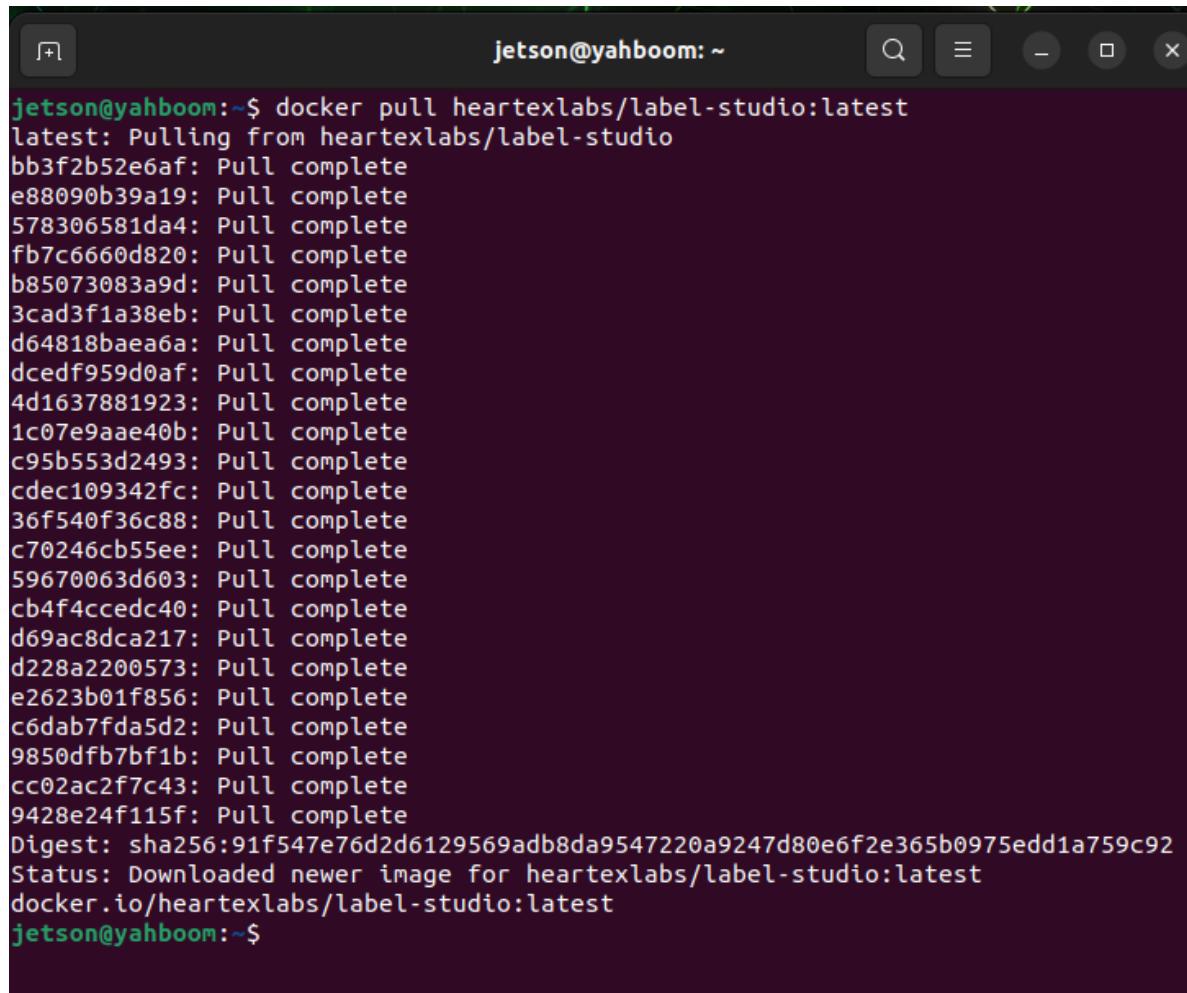
## 2. Label Studio usage

---

Label Studio is an open source data annotation platform for data annotation and annotation task management, supporting various types of data input and annotation formats.

## 2.1. Label Studio installation

```
docker pull heartexlabs/label-studio:latest
```

A screenshot of a terminal window titled "jetson@yahboom: ~". The window shows the command "docker pull heartexlabs/label-studio:latest" being run. The output lists numerous Docker image IDs followed by "Pull complete" messages. It concludes with "Status: Downloaded newer image for heartexlabs/label-studio:latest" and ends with the prompt "jetson@yahboom:~\$".

```
jetson@yahboom:~$ docker pull heartexlabs/label-studio:latest
latest: Pulling from heartexlabs/label-studio
bb3f2b52e6af: Pull complete
e88090b39a19: Pull complete
578306581da4: Pull complete
fb7c6660d820: Pull complete
b85073083a9d: Pull complete
3cad3f1a38eb: Pull complete
d64818baea6a: Pull complete
dcdef959d0af: Pull complete
4d1637881923: Pull complete
1c07e9aae40b: Pull complete
c95b553d2493: Pull complete
cdec109342fc: Pull complete
36f540f36c88: Pull complete
c70246cb55ee: Pull complete
59670063d603: Pull complete
cb4f4ccedc40: Pull complete
d69ac8dca217: Pull complete
d228a2200573: Pull complete
e2623b01f856: Pull complete
c6dab7fd45d2: Pull complete
9850dfb7bf1b: Pull complete
cc02ac2f7c43: Pull complete
9428e24f115f: Pull complete
Digest: sha256:91f547e76d2d6129569adb8da9547220a9247d80e6f2e365b0975edd1a759c92
Status: Downloaded newer image for heartexlabs/label-studio:latest
docker.io/heartexlabs/label-studio:latest
jetson@yahboom:~$
```

## 2.2. Launch Label Studio

### 2.2.1. Shared folder permissions

Share the prepared dataset to the folder corresponding to Label Studio:

```
sudo chmod 777 /home/jetson/ultralytics/ultralytics/data/
```

### 2.2.2. Start Label Studio

```
sudo docker run -it -p 8080:8080 -v
/home/jetson/ultralytics/ultralytics/data:/label-studio/data heartexlabs/label-
studio:latest label-studio --log-level DEBUG
```

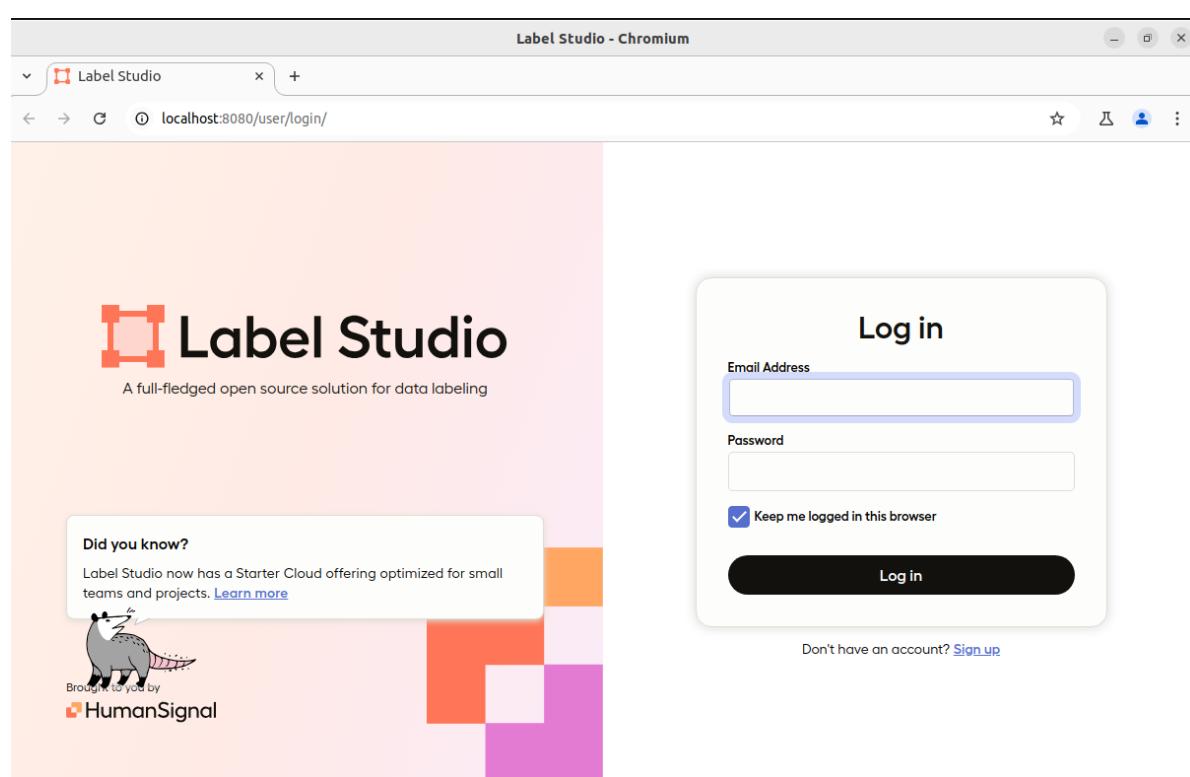
```
jetson@yahboom:~$ docker run -it -p 8080:8080 -v /home/jetson/ultralytics/ultralytics/data:/label-studio/data heartexlabs/label-studio:latest label-studio --log-level DEBUG
=> Database and media directory: /label-studio/data
=> Static URL is set to: /static/
=> Database and media directory: /label-studio/data
=> Static URL is set to: /static/
Read environment variables from: /label-studio/data/.env
get 'SECRET_KEY' casted as '<class 'str'>' with default ''
/lable-studio/.venv/lib/python3.12/site-packages/label_studio_sdk/_extensions/label_studio_tools/core/label_config.py:137: SyntaxWarning: invalid escape sequence '\$'
    expression = "^\$[A-Za-z_]+\$"
Starting new HTTPS connection (1): pypi.org:443
https://pypi.org:443 "GET /pypi/label-studio/json HTTP/1.1" 200 33651
[2024-12-31 09:12:15,565] [core.feature_flags.base::<module>::40] [INFO] Read flags from file /label-studio/label_studio/feature_flags.json
[2024-12-31 09:12:16,078] [core.redis::<module>::22] [DEBUG] => Redis is not connected.
[2024-12-31 09:12:16,502] [faker.factory::<module>::20] [DEBUG] Not in REPL -> leaving logger event level as is.
[2024-12-31 09:12:18,045] [faker.factory::_find_provider_class::78] [DEBUG] Looking for locale `en_US` in provider `faker.providers.address`.
[2024-12-31 09:12:18,048] [faker.factory::_find_provider_class::97] [DEBUG] Prov
```

## 2.2.3. Access Label Studio

### Host

Access method for Jetson Orin series development board. You need to enter your username and password for the first time.

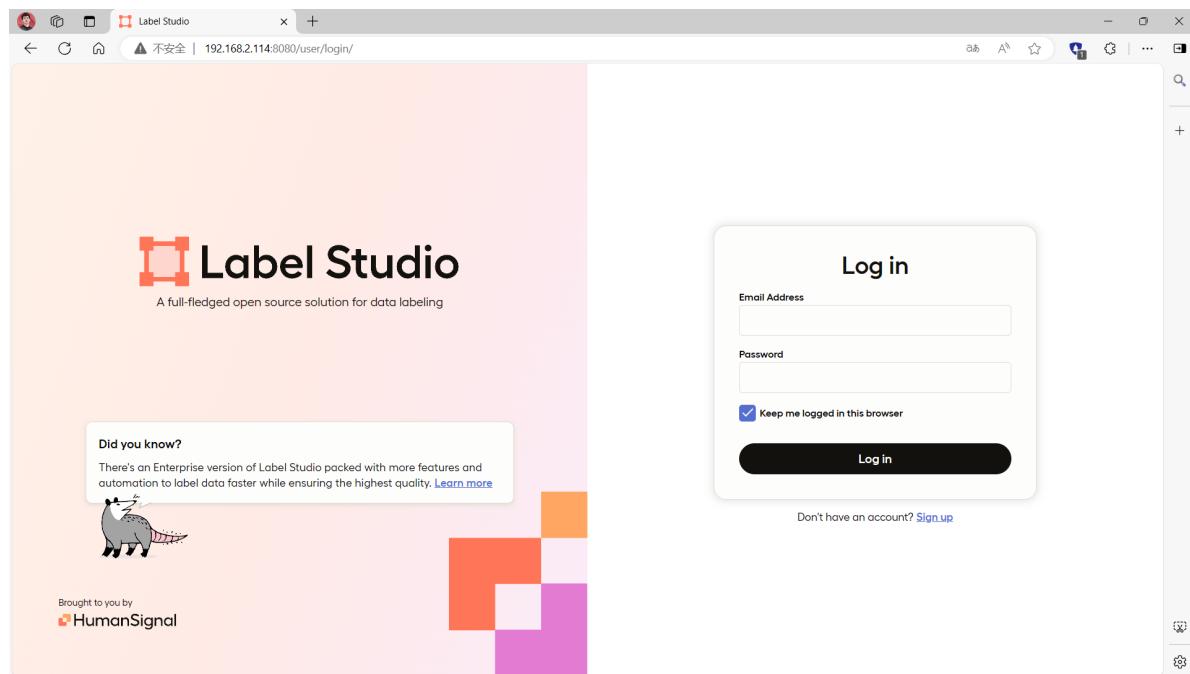
<http://localhost:8080/>



## Same LAN

Devices in the same LAN can be accessed based on the motherboard IP:8080

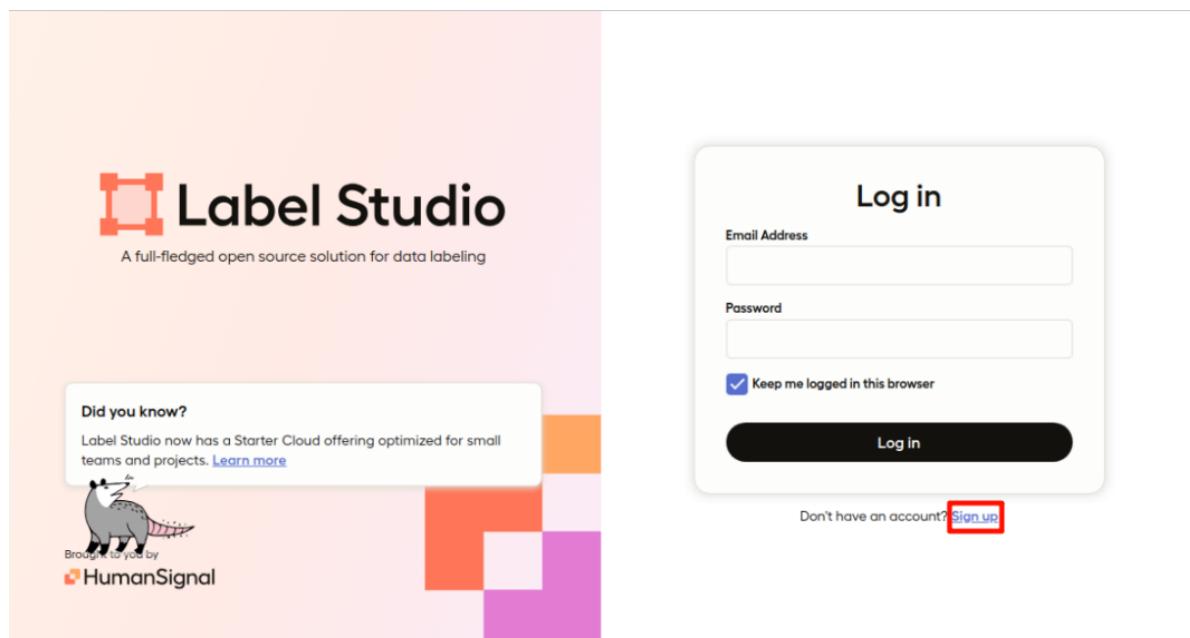
`http://motherboard IP:8080/ # For example: http://192.168.2.114:8080/`

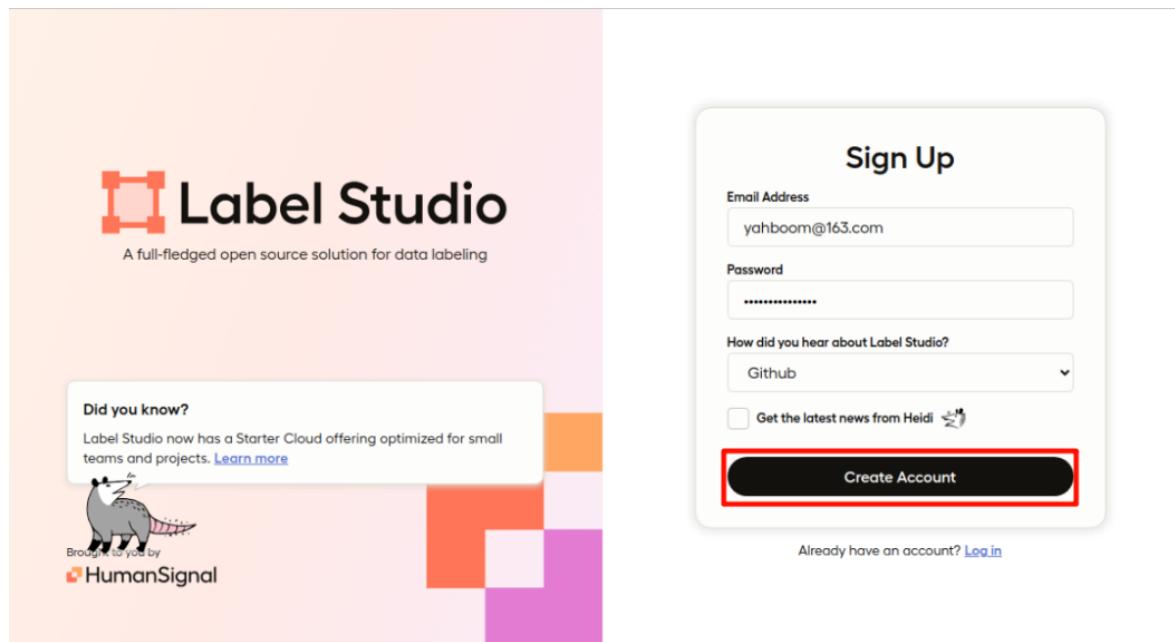


## First time use

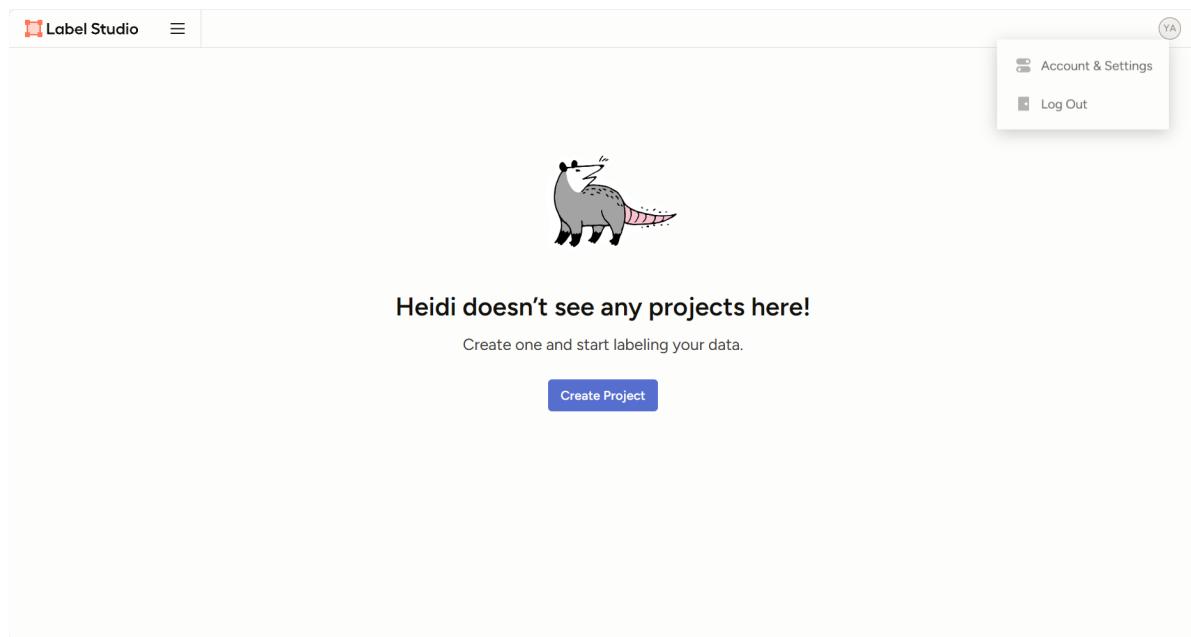
For the first time use, you need to register and log in to your account, and the account information is set by factory settings (the account information is false):

Account: `yahboom@163.com`  
Password: `yahboom@163.com`





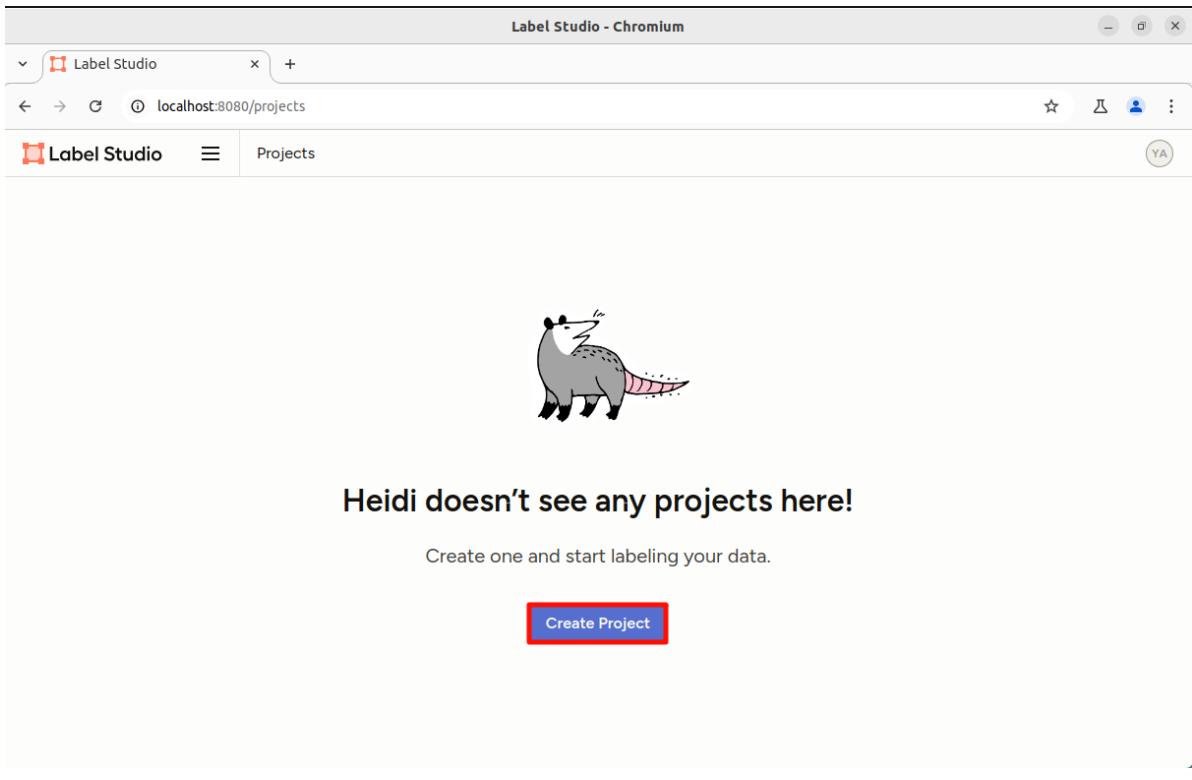
After registration is completed, the website will automatically log in:



## 3. Dataset labeling

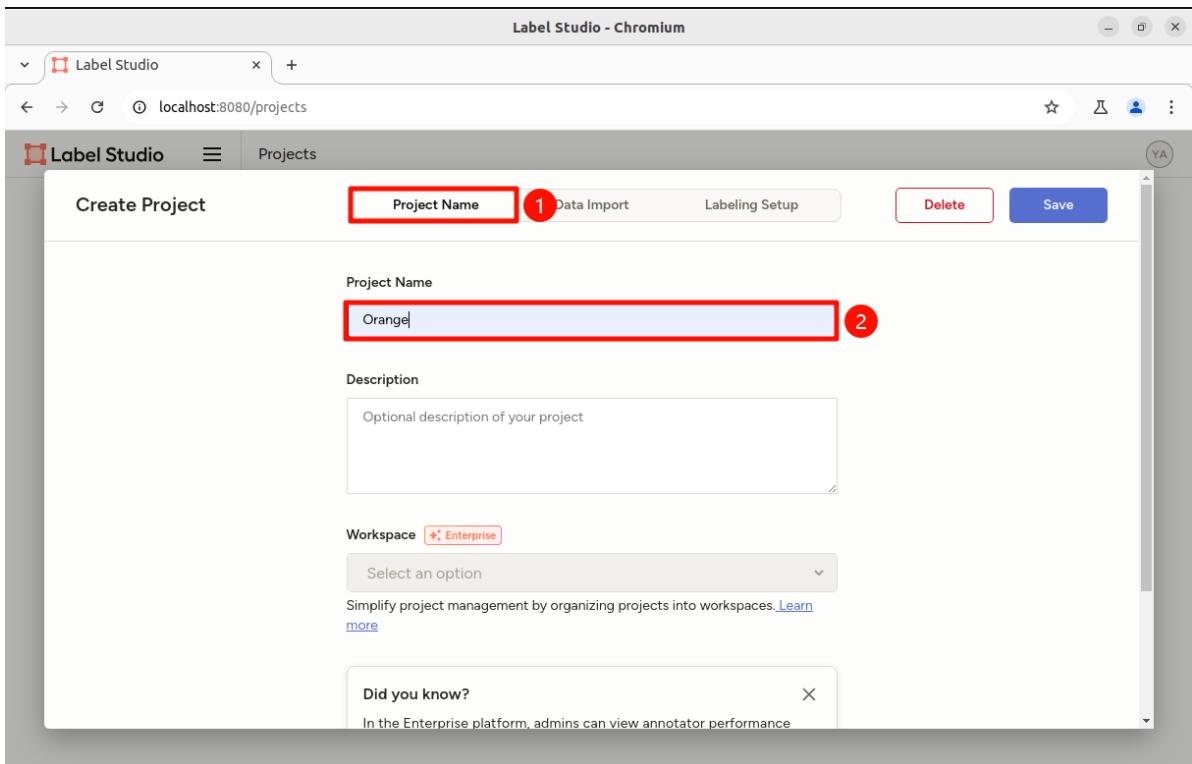
The demonstration dataset is on the main board, so access Label Studio on the main board.

### 3.1. Create a project



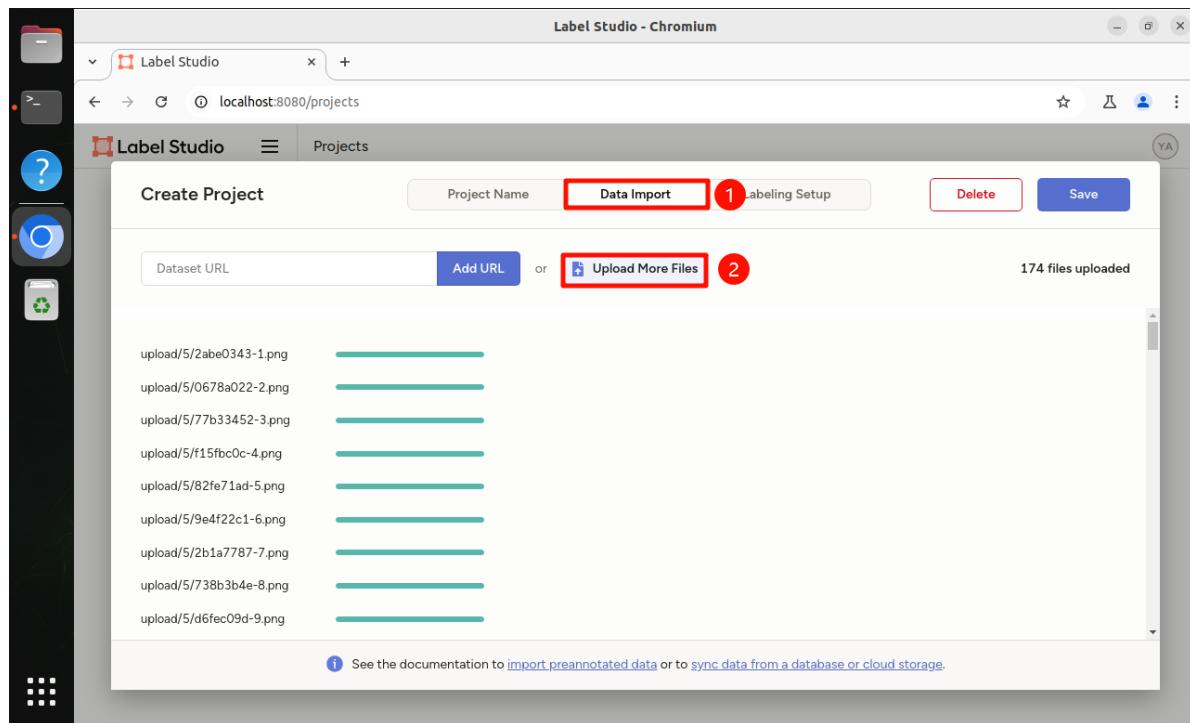
## 3.2. Project name

You can name the project as you like, based on your own training set:



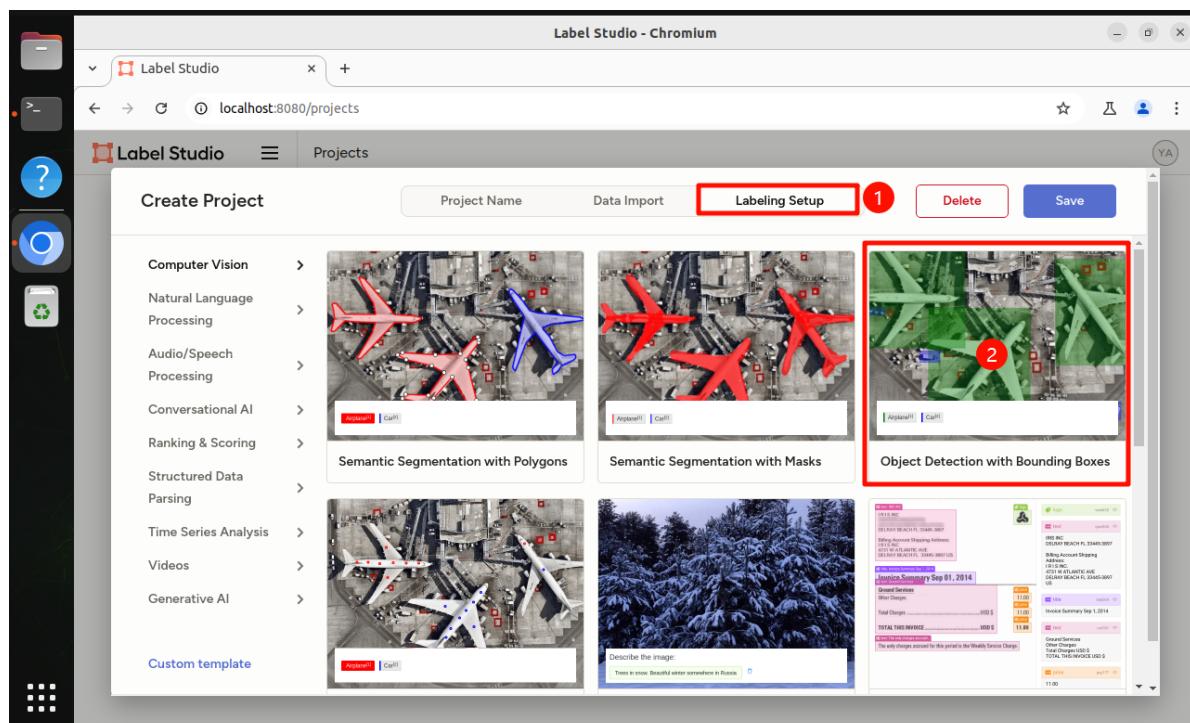
## 3.3. Import training set

Manually import the prepared dataset: Do not import too many images at a time. It is recommended to select 50-100 images each time and then import them multiple times.

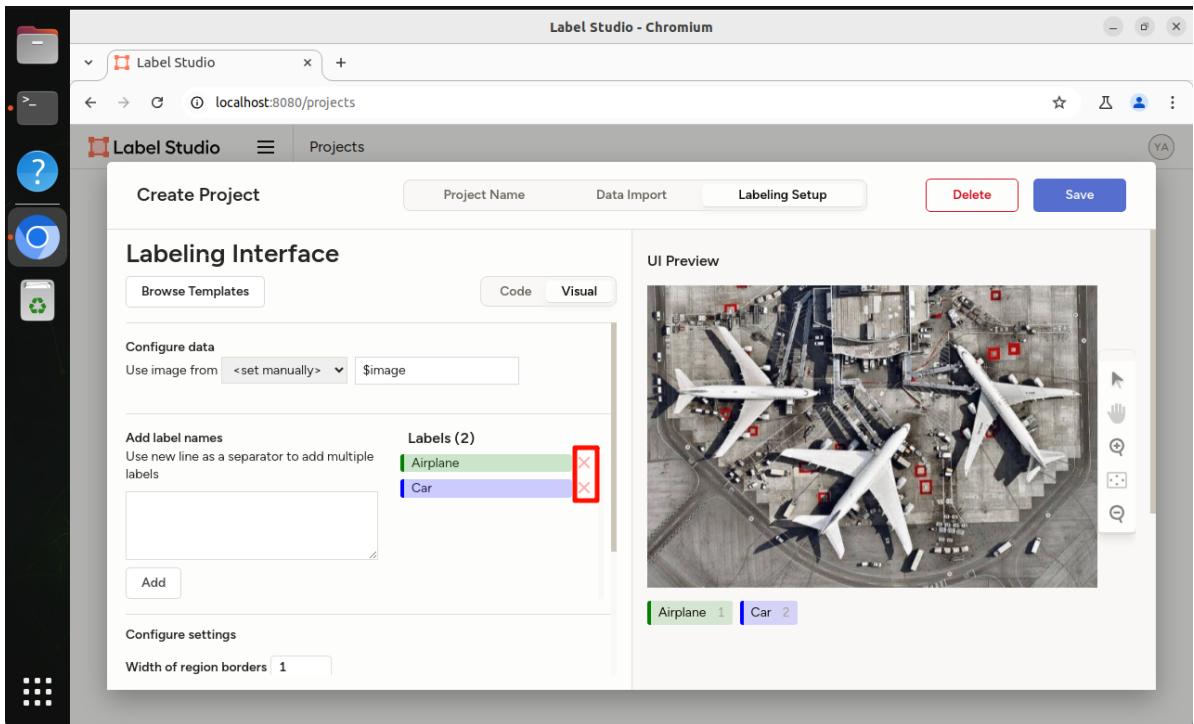


## 3.4, Label setting

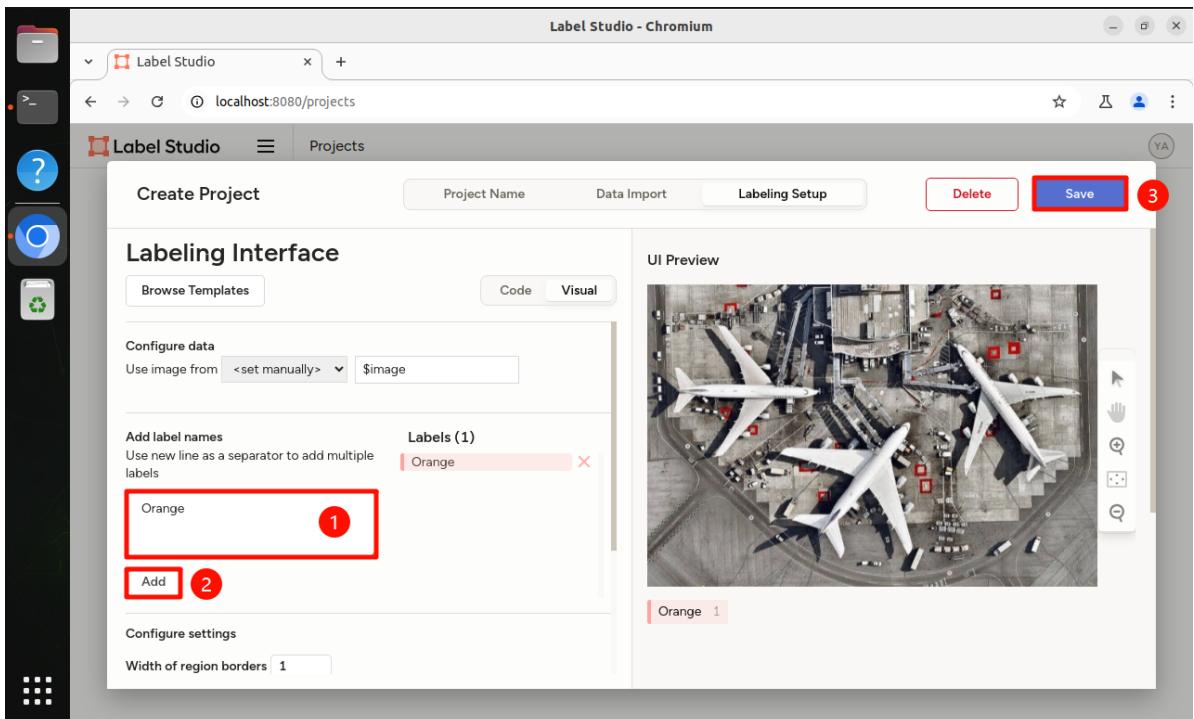
We are demonstrating the recognition of oranges, so select "Object Detection"



Delete built-in labels:



Add new labels:

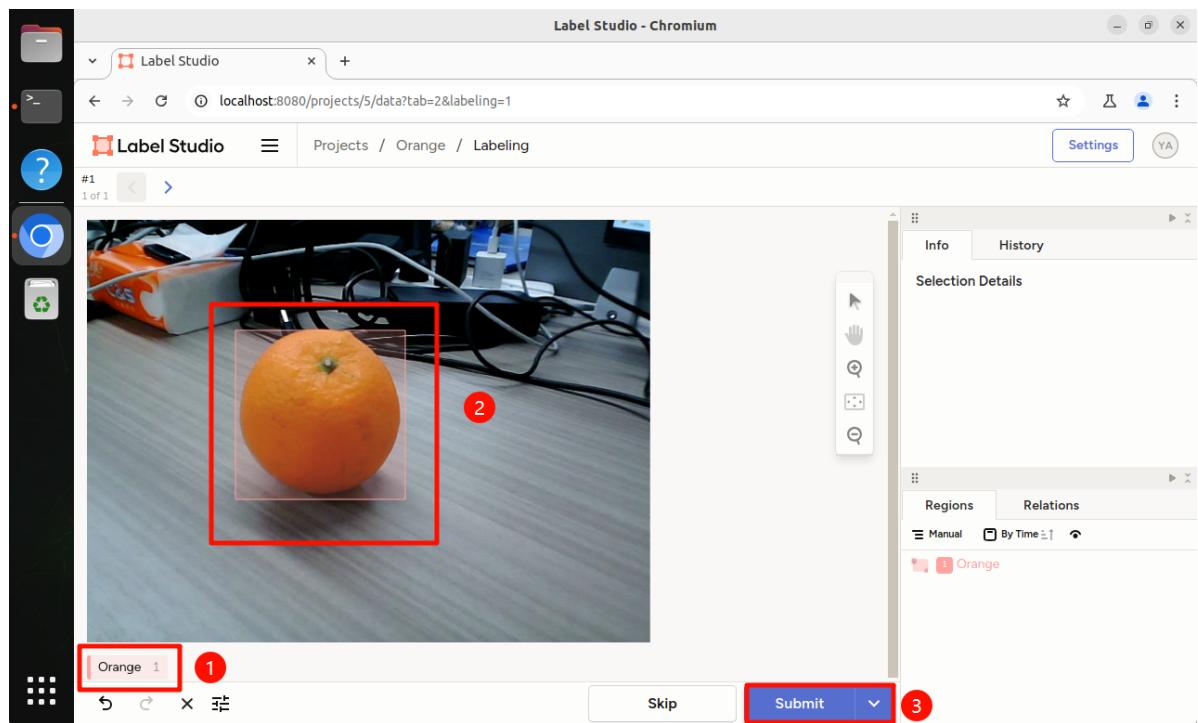


## 3.5. Dataset Annotation

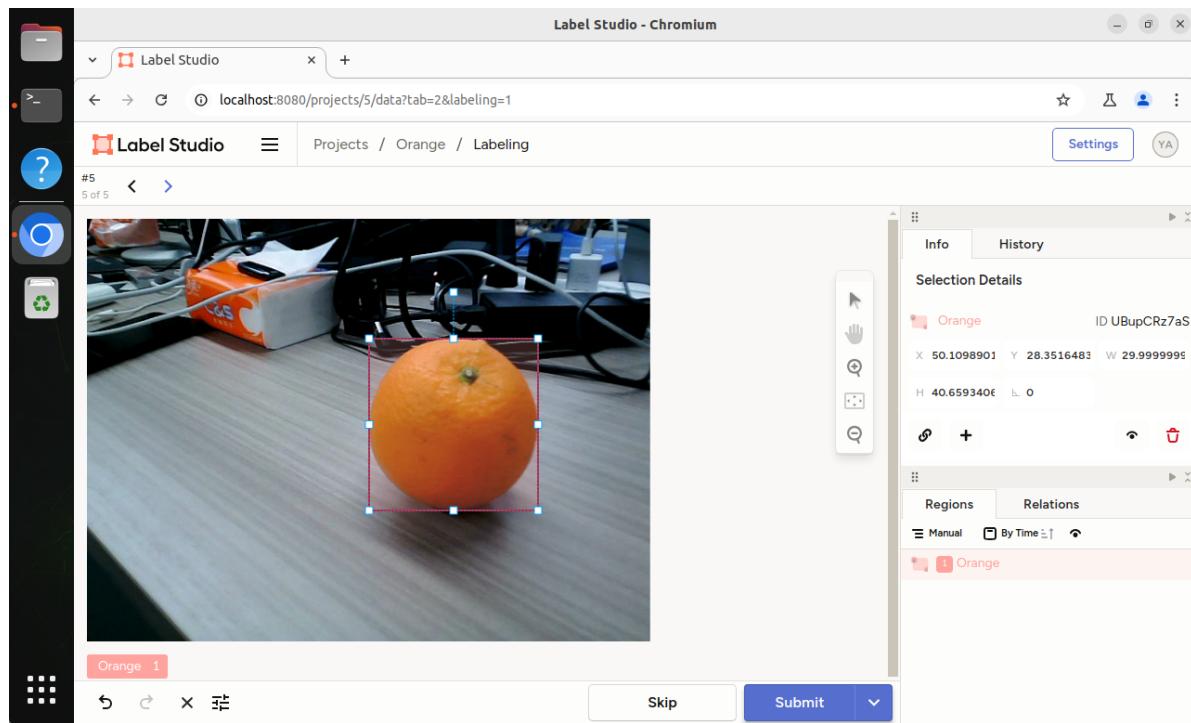
Click to annotate all tasks:

Label Studio - Chromium								
Projects / Orange								
Default								
Actions	Columns	Filters	Order	not set	Label All Tasks	Import	Export	List Grid
<input type="checkbox"/> ID	Completed	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Annotated by	<input type="checkbox"/>	Image	<input type="checkbox"/> Img
1	0	0	0					<a href="#">/p</a>
2	0	0	0					<a href="#">/p</a>
3	0	0	0					<a href="#">/p</a>
4	0	0	0					<a href="#">/p</a>
5	0	0	0					<a href="#">/p</a>
6	0	0	0					<a href="#">/p</a>
7	0	0	0					<a href="#">/p</a>

Annotation steps: First click the label "Orange" and then use the mouse to select the corresponding orange in the picture. After completion, click Submit to enter the next annotation



If the selected object is not the completed orange, you can click the selected box to readjust the position of each edge:

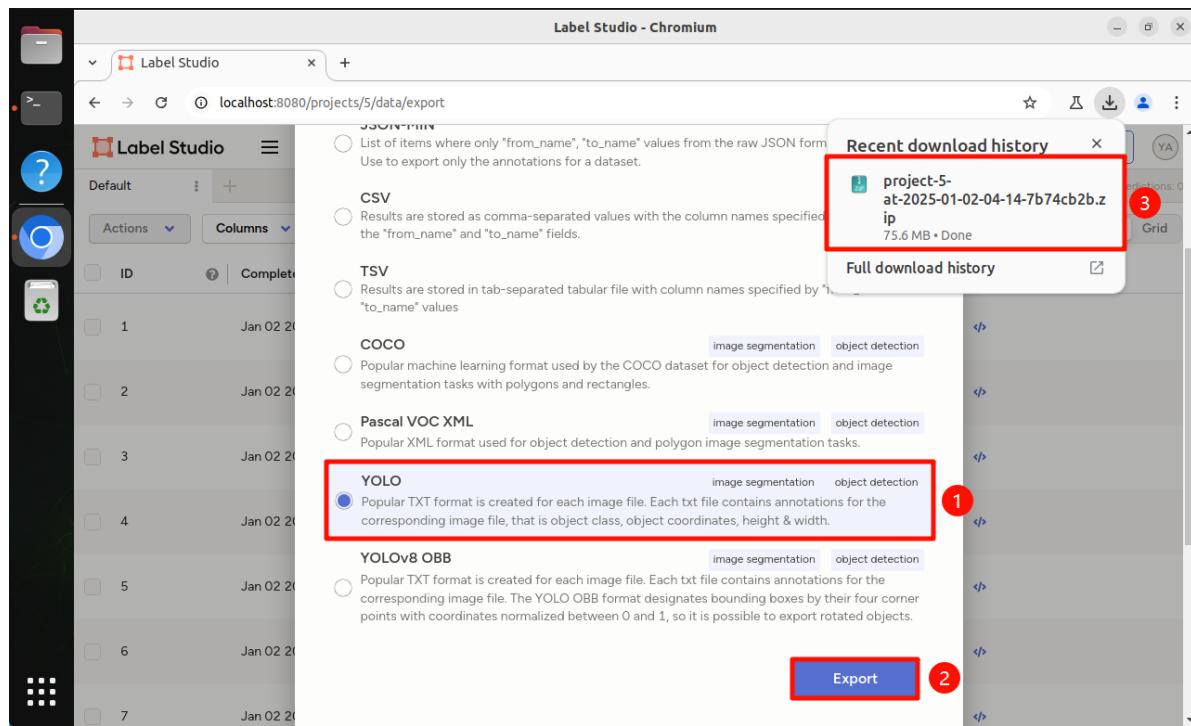


## 3.6. Export dataset

Select YOLO format for export, and the browser will automatically download a compressed package

A screenshot of the Label Studio interface showing a list of annotated tasks. The top navigation bar includes 'Label Studio - Chromium', a tab for 'Label Studio', and a URL 'localhost:8080/projects/5/data?tab=2'. Below the navigation is a toolbar with 'Actions', 'Columns', 'Filters', 'Order', 'not set', 'Label All Tasks', 'Import', 'Export' (which is highlighted with a red box), 'List', and 'Grid'. The main area is a table with columns: ID, Completed (with a timestamp), and three numerical columns (likely width, height, and depth). Each row contains a checkbox, a thumbnail image of the orange, and a download icon. The table has 7 rows, numbered 1 to 7. The status bar at the bottom shows 'Tasks: 174 / 174 Annotations: 174 Predictions: 0'.

ID	Completed				Annotated by	image	
1	Jan 02 2025, 11:13:39	1	0	0	YA		<a href="#">🔗</a>
2	Jan 02 2025, 11:14:40	1	0	0	YA		<a href="#">🔗</a>
3	Jan 02 2025, 11:14:55	1	0	0	YA		<a href="#">🔗</a>
4	Jan 02 2025, 11:15:21	1	0	0	YA		<a href="#">🔗</a>
5	Jan 02 2025, 11:29:30	1	0	0	YA		<a href="#">🔗</a>
6	Jan 02 2025, 11:29:51	1	0	0	YA		<a href="#">🔗</a>
7	Jan 02 2025, 11:30:05	1	0	0	YA		<a href="#">🔗</a>



### 3.7. Dataset directory framework

The directory framework corresponding to the compressed package exported using Label Studio:

```
.
├── images
├── labels
├── classes.txt
└── notes.json
```

We need to export images and labels from Label Studio into train (training set) and val (validation set), with a ratio of about 8:2: the images and labels in train and val need to have one-to-one correspondence in name.

```
.
├── classes.txt
├── notes.json
├── train
│   ├── images
│   └── labels
└── val
    ├── images
    └── labels
```

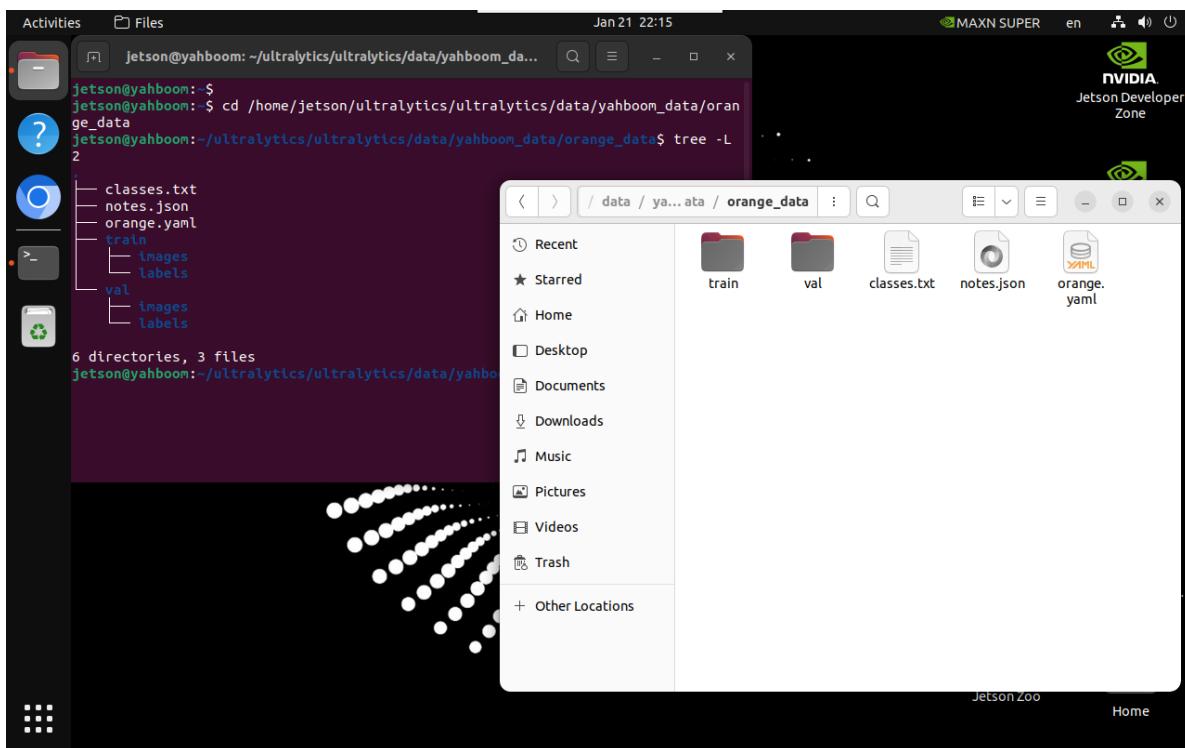
### 3.8. Dataset configuration file

Contains the paths of the training set and validation set, as well as the number and names of categories.

```
path: /home/jetson/ultralytics/ultralytics/data/yahboom_data/orange_data
train: train/images
val: val/images

nc: 1

# Classes
names:
0: Orange
```



## References

---

<https://github.com/HumanSignal/label-studio>