

BÁO CÁO:

IOT - LAB 3

Ngô Quang Minh - N22DCCN053

Nguyễn Khắc Tùng Dương - N22DCCN018

1. Tóm tắt Giới thiệu

Việc triển khai (deploy) một mô hình Machine Learning không chỉ dừng lại ở việc huấn luyện mô hình. Để mô hình có thể được sử dụng trong một ứng dụng thực tế, nó cần được "gói" trong một API và vận hành như một dịch vụ (service) ổn định, có khả năng mở rộng.

Quy trình này mô tả kiến trúc hiện đại sử dụng **Docker** để đóng gói ứng dụng, **Amazon Elastic Container Registry (ECR)** để lưu trữ các image, và **Amazon Elastic Container Service (ECS)** để tự động hóa việc triển khai, vận hành và nhân rộng dịch vụ.

Kiến trúc tổng quan:

- Local/CI/CD:** Mã nguồn ứng dụng (ví dụ: Flask/Python) và mô hình đóng gói vào một Docker image.
- Amazon ECR:** Docker image này được đẩy (push) và lưu trữ an toàn trong một kho lưu trữ (repository) riêng tư.
- Amazon ECS:** Dịch vụ sẽ đọc image từ ECR, sau đó tự động khởi chạy và duy trì các container.
- Client:** Người dùng cuối truy cập vào dịch vụ thông qua một địa chỉ IP công cộng hoặc một Bộ cân bằng tải (Load Balancer).

2. Các Công nghệ Chính

- Docker:** Nền tảng để đóng gói ứng dụng (mã nguồn, thư viện, mô hình) vào một đơn vị độc lập gọi là "container image".
- Amazon ECR (Elastic Container Registry):** Dịch vụ lưu trữ Docker image được quản lý hoàn toàn, có độ an toàn và tin cậy cao, tích hợp chặt chẽ với hệ sinh thái AWS.

- **Amazon ECS (Elastic Container Service):** Dịch vụ điều phối container (container orchestration) của AWS. ECS cho phép bạn chạy, dừng và quản lý các Docker container trên một cụm (cluster) máy chủ. Chúng ta sẽ tập trung vào chế độ **Fargate**, là chế độ "serverless" (không cần quản lý máy chủ).

3. Quy trình Triển khai Chi tiết

Đây là các bước kỹ thuật để triển khai từ máy tính cá nhân lên đám mây AWS.

Bước 1: Docker hóa Ứng dụng Machine Learning

Trước tiên, ứng dụng Python (máy chủ Flask) và các mô hình của bạn cần được đóng gói.

1. Cấu trúc thư mục (Ví dụ):

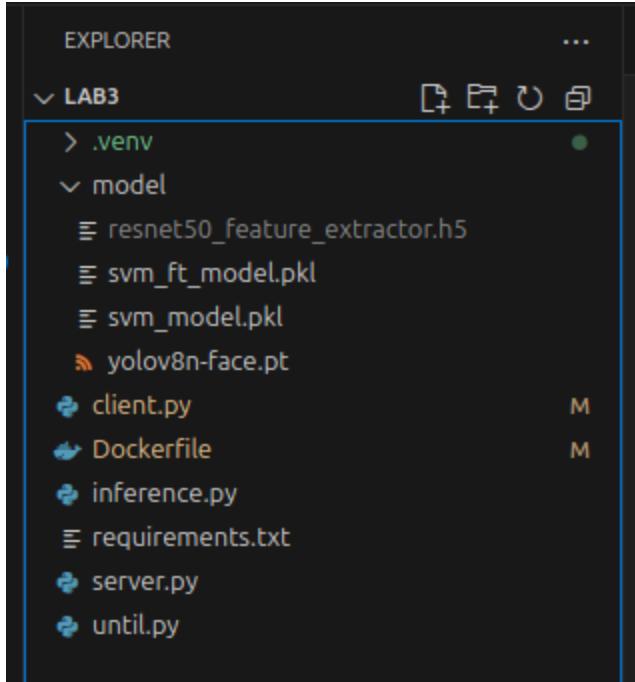
```
/warning-mask-detection
|-- /models
|   |-- restnet_50.h5
|   |-- svm.pkl
|   |-- yolo.pt
|-- server.py
|-- requirements.txt
|-- Dockerfile
|-- utils
```

2. Viết Dockerfile: Đây là tệp hướng dẫn Docker cách xây dựng image.

Lưu ý quan trọng: Log của bạn cho thấy bạn đang dùng máy chủ phát triển (dev server) của Flask. Điều này **không** an toàn và không hiệu quả cho production. Bạn nên dùng một máy chủ WSGI như **Gunicorn**.

Đây là một Dockerfile mẫu cho production:

3. Dockerfile



Bước 2: Đẩy Image lên Amazon ECR

Sau khi có **Dockerfile**, bạn cần xây dựng và đẩy image lên "kho" ECR.

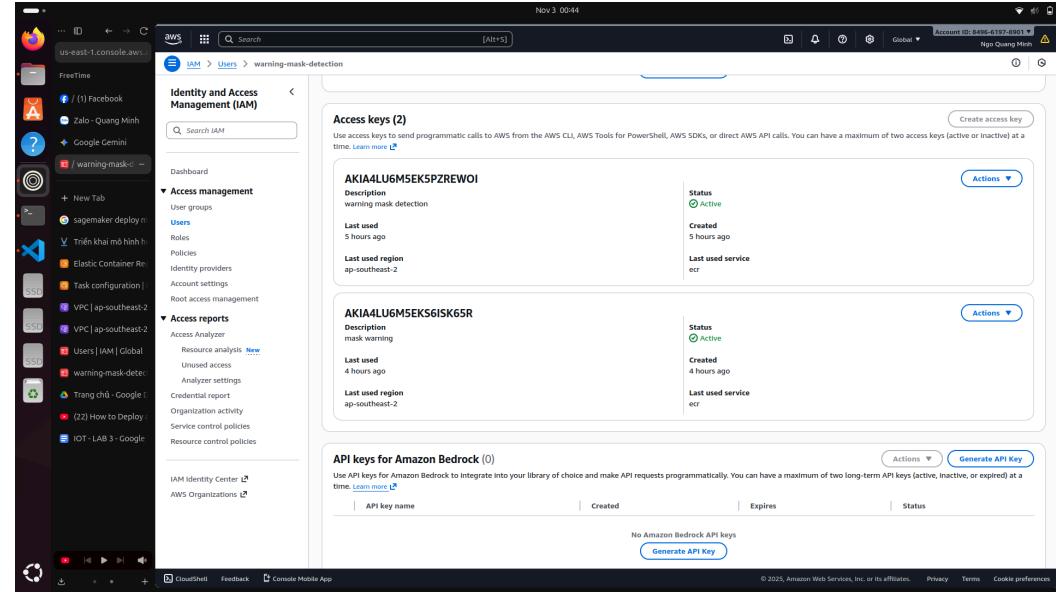
1. Tạo một Repository trên ECR:

- Truy cập Bảng điều khiển (Console) AWS > **ECR**.
- Nhập "Create repository" (Tạo kho).
- Đặt tên (ví dụ: **warning-mask-detection**) và giữ các cài đặt khác ở chế độ riêng tư (private).

2. Xác thực Docker với ECR:

- Cấp phép cho Docker CLI trên máy của mình để được đẩy image lên ECR.

Access key:

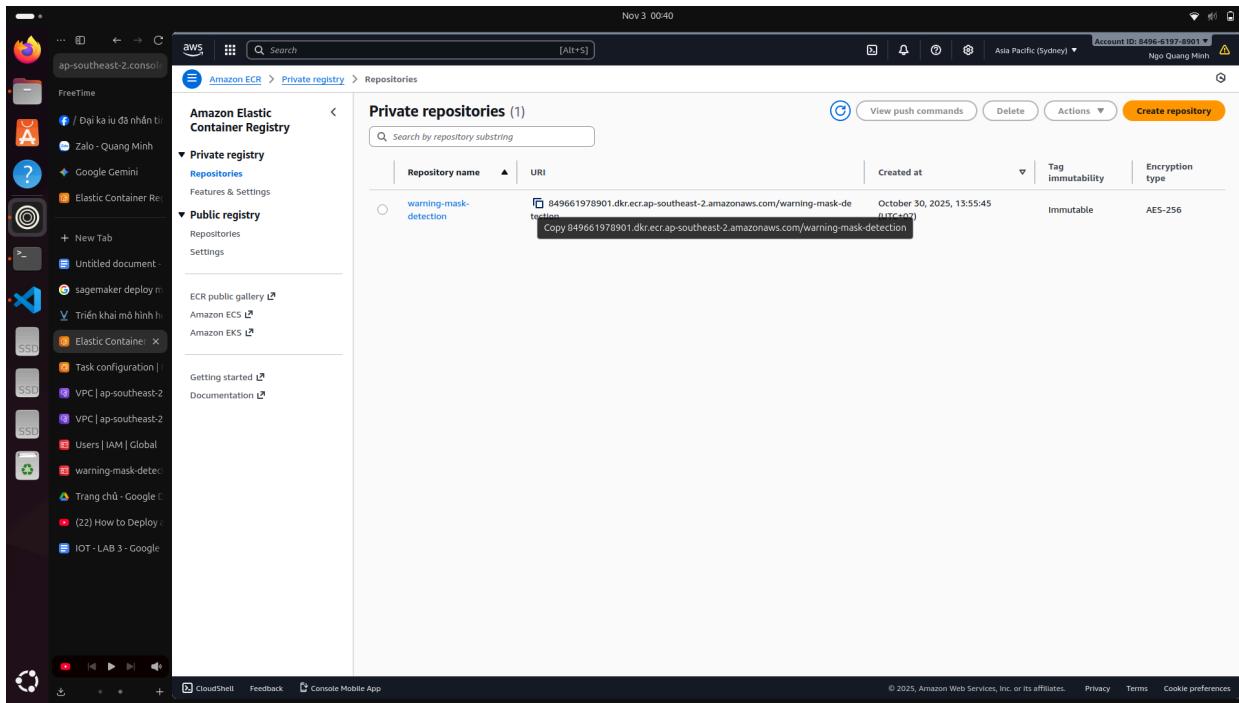


Bước 3: Build, Tag, và Push Image:

- **Build:** Xây dựng image từ Dockerfile.

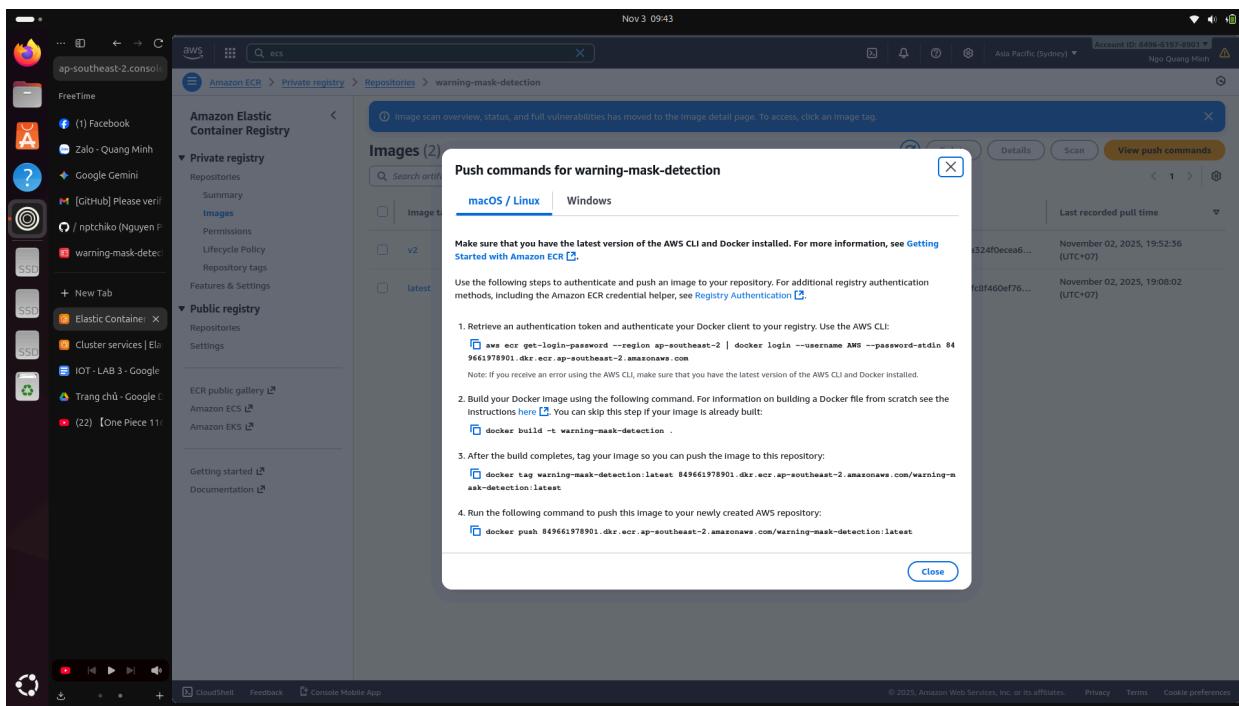
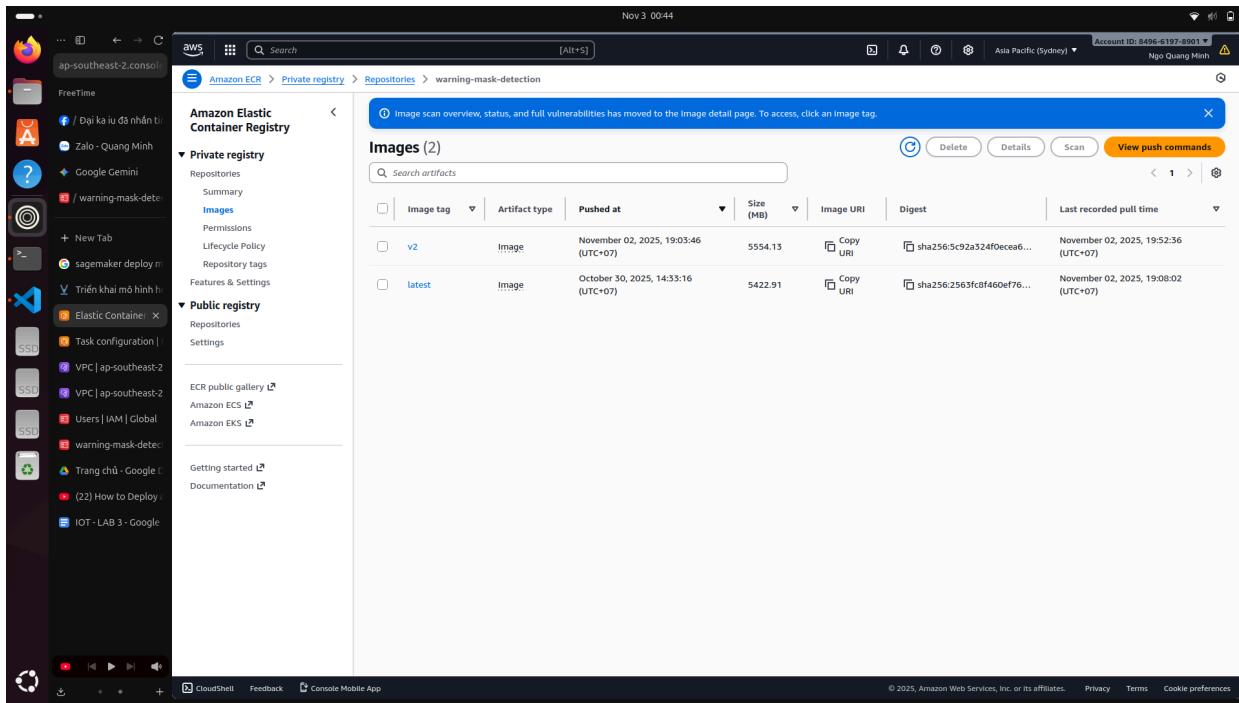
```
Nov 3 00:38
minhminh@minhminh-ThinkBook-14-G2-ITL:~/workspace/PTIT/IOT/LAB3$ docker images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
849661978901.dkr.ecr.ap-southeast-2.amazonaws.com/warning-mask-detection   v2      c447e34e71a4   6 hours ago   10.9GB
                                         v2      c447e34e71a4   6 hours ago   10.9GB
                                         latest  8890de499e99  7 hours ago   10.6GB
```

- **Tag:** Gắn thẻ image theo định dạng ECR yêu cầu.



```
minhminh@minhminh-ThinkBook-14-G2-ITL:~/workspace/PTIT/IOT/LAB3$ docker images
REPOSITORY          TAG      IMAGE ID   CREATED     SIZE
849661978901.dkr.ecr.ap-southeast-2.amazonaws.com/warning-mask-detection   v2       c447e34e71a4  6 hours ago  10.9GB
warning-wearing-mask           v2       c447e34e71a4  6 hours ago  10.9GB
warning-wearing-mask           latest   8890de499e99  7 hours ago  10.6GB
minhminh@minhminh-ThinkBook-14-G2-ITL:~/workspace/PTIT/IOT/LAB3$ docker tag warning-wearing-mask:v2 849661978901.dkr.ecr.ap-southeast-2.amazonaws.com/warning-mask-detection
```

- **Push:** Đẩy image lên kho lưu trữ ECR.



Bước 4. Vận hành Dịch vụ trên Amazon ECS

Đây là bước đang thực hiện. Chúng ta sẽ ECS lấy image từ ECR và chạy nó.

Understanding Services and Tasks



1. Tạo ECS Cluster (Cụm):

- Truy cập Bảng điều khiển (Console) AWS > **ECS**.
- Nhập "Create cluster" (Tạo cụm).
- Chọn "Networking only" (chỉ Mạng) để sử dụng **AWS Fargate**
- Đặt tên cho Cluster (ví dụ: `hosipitable-penguin-rfkjpf`).

2. Tạo Task Definition (Định nghĩa Tác vụ):

- Đây là "bản thiết kế" cho container của bạn.
- Trong ECS, vào "Task Definitions" > "Create new Task Definition".
- Chọn "Fargate" làm launch type.
- **Task role:** (Tùy chọn) Nếu ứng dụng của bạn cần gọi các dịch vụ AWS khác (như S3), hãy gán quyền ở đây.
- **Operating system:** Linux.
- **Task size:** Cung cấp CPU và Bộ nhớ (ví dụ: 1 vCPU, 4GB Memory - các mô hình ML thường cần nhiều RAM).
- **Container Definitions (Quan trọng):**
 - Nhập "Add container".
 - **Image (Quan trọng):** Dán đường dẫn URI của image ECR (ví dụ:
`<account-id>.dkr.ecr.ap-southeast-2.amazonaws.com/warning-mask-detection:v2`).
 - **Port mappings (Quan trọng):** Thêm một mục: `Container port: 5000` (đây là cổng EXPOSE trong Dockerfile của bạn).

3. Tạo Service (Dịch vụ):

- Đây là bước khởi chạy và duy trì Tác vụ của bạn.
- Đến Cluster của bạn, nhấp vào tab "Services" > "Create".
- **Launch type:** Fargate.
- **Task Definition:** Chọn Task Definition bạn vừa tạo.
- **Service name:** Đặt tên (ví dụ: `warning-mask-v2-service-gku889s1`).
- **Desired tasks:** 1 (để chạy 1 bản sao của container).
- **Networking (Bước bạn đang gặp sự cố):**
 - **VPC & Subnets:** Chọn VPC và các Subnet (thường là mặc định).
 - **Security Groups (Rất quan trọng):**
 - Tạo một Security Group (nhóm bảo mật) MỚI.
 - Thêm một "Inbound rule" (Quy tắc đi vào).
 - Loại (Type): `Custom TCP`.
 - Port Range (Dải cổng): `5000` (đây là cổng container).
 - Nguồn (Source): `Anywhere (0.0.0.0/0)` (để cho phép truy cập từ Internet).
 - **Auto-assign public IP (Rất quan trọng):** PHẢI đặt thành **ENABLED** (ĐÃ BẬT). Nếu không, container sẽ không có IP công cộng và không thể truy cập nó từ bên ngoài.
- **Load Balancing (Tùy chọn nhưng được khuyến nghị):**
 - Để sử dụng cho production (và truy cập qua cổng 80/443 thân thiện), bạn nên chọn "Application Load Balancer" (ALB).
 - ALB sẽ lắng nghe trên cổng `80` (HTTP) và chuyển tiếp lưu lượng truy cập đến "Target Group" (Nhóm mục tiêu).
 - Nhóm mục tiêu (Target Group) sẽ được cấu hình để gửi lưu lượng truy cập đến cổng `5000` của Tác vụ (Task).

4. Truy cập Dịch vụ:

- Sau khi Dịch vụ (Service) chuyển sang trạng thái "RUNNING" (Đang chạy), hãy đi tới tab "Tasks" (Tác vụ) và nhấp vào Tác vụ (Task) đang chạy.
- Trong phần "Network", tìm "**Public IP**" (IP công cộng) của nó.
- Truy cập dịch vụ của bạn trong trình duyệt bằng:
`http://<Public-IP-cua-ban>:5000`

4. Kết luận và Khắc phục sự cố

Kiến trúc ECR + ECS Fargate là một phương pháp mạnh mẽ, hiện đại và có khả năng mở rộng cao để triển khai các ứng dụng, đặc biệt là các mô hình ML.

Vấn đề phổ biến nhất: "Không thể truy cập URL."

Nguyên nhân luôn là do **Networking (Mạng)**:

- Security Group:** Đã chặn cổng. Đảm bảo cổng (5000) được mở cho 0.0.0.0/0 trong Inbound Rules.
- Public IP:** Tác vụ (Task) không được gán IP công cộng. Đảm bảo "Auto-assign public IP" được **ENABLED** khi tạo Dịch vụ.
- Port Mapping:** Cổng trong Task Definition (5000) không khớp với cổng mà ứng dụng Flask/Gunicorn của bạn đang chạy (5000).

The screenshot shows the AWS VPC Security Groups console. The main view displays the details of a security group named 'sg-063e6616cbf3e0c4d - default'. It shows the security group ID, owner, and various counts of rules. The 'Inbound rules' tab is selected, showing three rules:

Name	Security group rule ID	Type	Protocol	Port range	Source
-	sgr-0cad840b741b5a25b	All traffic	All	5000	0.0.0.0/0
-	sgr-033a969897722b4cb	MySQL	TCP	1433	0.0.0.0/0
-	sgr-0f9a89fbe8cc2ff89	Custom TCP	TCP	5000	0.0.0.0/0

5. DEMO

Public server ip: <http://54.206.102.233:5000/predict>

1. Mask:

Nov 3 00:53

```

Dockerfile M client.py M server.py

1 import cv2
2 import requests # <-- THAY THẾ 'boto3'
3 import numpy as np
4 import json
5 import time
6
7 # --- CẤU HÌNH BẮT BUỘC CHỖ
8 # 1. Thay thế bằng URL của
9 LOCAL_API_URL = "http://54.162.206.102:5000/predict"
10
11 # --- KHỞI TẠO CAMERA ---
12 cap = cv2.VideoCapture(0)
13
14 if not cap.isOpened():
15     print("Lỗi: Không thể mở camera")
16     exit()
17
18 print("Đang kết nối tới Server...")
19 print("Nhấn 'S' để chụp ảnh")
20
21 # --- VÒNG LẶP CHỤP VÀ GỬI ---
22 while True:
23     ret, frame = cap.read()
24
25     if not ret:
26         print("Lỗi: Không đọc được khung hình")
27         break
28
29     display_frame = frame.copy()
30
31     # Duyệt qua từng khung hình
32     for face in faces:
33         x, y, w, h = face
34
35         # Kiểm tra xem có đeo khẩu trang không
36         if mask_classifier.predict(face) == 1:
37             # Đeo khẩu trang
38             color = (0, 255, 0) # Xanh lá cây
39             label = "with_mask"
40         else:
41             # Không đeo khẩu trang
42             color = (0, 0, 255) # Đỏ
43             label = "without_mask"
44
45         # Vẽ khung viền và nhãn
46         cv2.rectangle(display_frame, (x, y), (x+w, y+h), color, 2)
47         cv2.putText(display_frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
48
49     # Hiển thị khung hình kết quả
50     cv2.imshow('Face Mask Detector - Client (LocalServer)', display_frame)
51
52     # Nhấn 'q' để thoát
53     if cv2.waitKey(1) & 0xFF == ord('q'):
54         break
55
56     # Nhấn 'S' để chụp ảnh và gửi
57     if cv2.waitKey(1) & 0xFF == ord('S'):
58         # Lấy khung hình
59         frame = display_frame[y:y+h, x:x+w]
60
61         # Chuyển đổi sang grayscale
62         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
63
64         # Lưu ảnh
65         cv2.imwrite('image.jpg', frame)
66
67         # Gửi yêu cầu đến API
68         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
69
70         # In kết quả
71         print(response.json())
72
73     # Nhấn 'ESC' để thoát
74     if cv2.waitKey(1) & 0xFF == 27:
75         break
76
77     # Nhấn 'F' để tăng độ nhạy cảm
78     if cv2.waitKey(1) & 0xFF == ord('F'):
79         sensitivity += 0.05
80
81     # Nhấn 'B' để giảm độ nhạy cảm
82     if cv2.waitKey(1) & 0xFF == ord('B'):
83         sensitivity -= 0.05
84
85     # Nhấn 'P' để tăng độ chính xác
86     if cv2.waitKey(1) & 0xFF == ord('P'):
87         accuracy += 0.05
88
89     # Nhấn 'N' để giảm độ chính xác
90     if cv2.waitKey(1) & 0xFF == ord('N'):
91         accuracy -= 0.05
92
93     # Nhấn 'R' để reset
94     if cv2.waitKey(1) & 0xFF == ord('R'):
95         reset()
96
97     # Nhấn 'Q' để thoát
98     if cv2.waitKey(1) & 0xFF == ord('Q'):
99         break
100
101     # Nhấn 'M' để tắt/mở camera
102     if cv2.waitKey(1) & 0xFF == ord('M'):
103         cap.isOpened() = not cap.isOpened()
104
105     # Nhấn 'C' để chụp ảnh
106     if cv2.waitKey(1) & 0xFF == ord('C'):
107         # Lấy khung hình
108         frame = display_frame[y:y+h, x:x+w]
109
110         # Chuyển đổi sang grayscale
111         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
112
113         # Lưu ảnh
114         cv2.imwrite('image.jpg', frame)
115
116         # Gửi yêu cầu đến API
117         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
118
119         # In kết quả
120         print(response.json())
121
122     # Nhấn 'E' để tăng độ chính xác
123     if cv2.waitKey(1) & 0xFF == ord('E'):
124         accuracy += 0.05
125
126     # Nhấn 'D' để giảm độ chính xác
127     if cv2.waitKey(1) & 0xFF == ord('D'):
128         accuracy -= 0.05
129
130     # Nhấn 'A' để tăng độ nhạy cảm
131     if cv2.waitKey(1) & 0xFF == ord('A'):
132         sensitivity += 0.05
133
134     # Nhấn 'Z' để giảm độ nhạy cảm
135     if cv2.waitKey(1) & 0xFF == ord('Z'):
136         sensitivity -= 0.05
137
138     # Nhấn 'S' để chụp ảnh và gửi
139     if cv2.waitKey(1) & 0xFF == ord('S'):
140         # Lấy khung hình
141         frame = display_frame[y:y+h, x:x+w]
142
143         # Chuyển đổi sang grayscale
144         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
145
146         # Lưu ảnh
147         cv2.imwrite('image.jpg', frame)
148
149         # Gửi yêu cầu đến API
150         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
151
152         # In kết quả
153         print(response.json())
154
155     # Nhấn 'Q' để thoát
156     if cv2.waitKey(1) & 0xFF == ord('Q'):
157         break
158
159     # Nhấn 'F' để tăng độ nhạy cảm
160     if cv2.waitKey(1) & 0xFF == ord('F'):
161         sensitivity += 0.05
162
163     # Nhấn 'B' để giảm độ nhạy cảm
164     if cv2.waitKey(1) & 0xFF == ord('B'):
165         sensitivity -= 0.05
166
167     # Nhấn 'P' để tăng độ chính xác
168     if cv2.waitKey(1) & 0xFF == ord('P'):
169         accuracy += 0.05
170
171     # Nhấn 'N' để giảm độ chính xác
172     if cv2.waitKey(1) & 0xFF == ord('N'):
173         accuracy -= 0.05
174
175     # Nhấn 'R' để reset
176     if cv2.waitKey(1) & 0xFF == ord('R'):
177         reset()
178
179     # Nhấn 'Q' để thoát
180     if cv2.waitKey(1) & 0xFF == ord('Q'):
181         break
182
183     # Nhấn 'M' để tắt/mở camera
184     if cv2.waitKey(1) & 0xFF == ord('M'):
185         cap.isOpened() = not cap.isOpened()
186
187     # Nhấn 'C' để chụp ảnh
188     if cv2.waitKey(1) & 0xFF == ord('C'):
189         # Lấy khung hình
190         frame = display_frame[y:y+h, x:x+w]
191
192         # Chuyển đổi sang grayscale
193         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
194
195         # Lưu ảnh
196         cv2.imwrite('image.jpg', frame)
197
198         # Gửi yêu cầu đến API
199         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
200
201         # In kết quả
202         print(response.json())
203
204     # Nhấn 'E' để tăng độ chính xác
205     if cv2.waitKey(1) & 0xFF == ord('E'):
206         accuracy += 0.05
207
208     # Nhấn 'D' để giảm độ chính xác
209     if cv2.waitKey(1) & 0xFF == ord('D'):
210         accuracy -= 0.05
211
212     # Nhấn 'A' để tăng độ nhạy cảm
213     if cv2.waitKey(1) & 0xFF == ord('A'):
214         sensitivity += 0.05
215
216     # Nhấn 'Z' để giảm độ nhạy cảm
217     if cv2.waitKey(1) & 0xFF == ord('Z'):
218         sensitivity -= 0.05
219
220     # Nhấn 'S' để chụp ảnh và gửi
221     if cv2.waitKey(1) & 0xFF == ord('S'):
222         # Lấy khung hình
223         frame = display_frame[y:y+h, x:x+w]
224
225         # Chuyển đổi sang grayscale
226         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
227
228         # Lưu ảnh
229         cv2.imwrite('image.jpg', frame)
230
231         # Gửi yêu cầu đến API
232         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
233
234         # In kết quả
235         print(response.json())
236
237     # Nhấn 'Q' để thoát
238     if cv2.waitKey(1) & 0xFF == ord('Q'):
239         break
240
241     # Nhấn 'F' để tăng độ nhạy cảm
242     if cv2.waitKey(1) & 0xFF == ord('F'):
243         sensitivity += 0.05
244
245     # Nhấn 'B' để giảm độ nhạy cảm
246     if cv2.waitKey(1) & 0xFF == ord('B'):
247         sensitivity -= 0.05
248
249     # Nhấn 'P' để tăng độ chính xác
250     if cv2.waitKey(1) & 0xFF == ord('P'):
251         accuracy += 0.05
252
253     # Nhấn 'N' để giảm độ chính xác
254     if cv2.waitKey(1) & 0xFF == ord('N'):
255         accuracy -= 0.05
256
257     # Nhấn 'R' để reset
258     if cv2.waitKey(1) & 0xFF == ord('R'):
259         reset()
260
261     # Nhấn 'Q' để thoát
262     if cv2.waitKey(1) & 0xFF == ord('Q'):
263         break
264
265     # Nhấn 'M' để tắt/mở camera
266     if cv2.waitKey(1) & 0xFF == ord('M'):
267         cap.isOpened() = not cap.isOpened()
268
269     # Nhấn 'C' để chụp ảnh
270     if cv2.waitKey(1) & 0xFF == ord('C'):
271         # Lấy khung hình
272         frame = display_frame[y:y+h, x:x+w]
273
274         # Chuyển đổi sang grayscale
275         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
276
277         # Lưu ảnh
278         cv2.imwrite('image.jpg', frame)
279
280         # Gửi yêu cầu đến API
281         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
282
283         # In kết quả
284         print(response.json())
285
286     # Nhấn 'E' để tăng độ chính xác
287     if cv2.waitKey(1) & 0xFF == ord('E'):
288         accuracy += 0.05
289
290     # Nhấn 'D' để giảm độ chính xác
291     if cv2.waitKey(1) & 0xFF == ord('D'):
292         accuracy -= 0.05
293
294     # Nhấn 'A' để tăng độ nhạy cảm
295     if cv2.waitKey(1) & 0xFF == ord('A'):
296         sensitivity += 0.05
297
298     # Nhấn 'Z' để giảm độ nhạy cảm
299     if cv2.waitKey(1) & 0xFF == ord('Z'):
300         sensitivity -= 0.05
301
302     # Nhấn 'S' để chụp ảnh và gửi
303     if cv2.waitKey(1) & 0xFF == ord('S'):
304         # Lấy khung hình
305         frame = display_frame[y:y+h, x:x+w]
306
307         # Chuyển đổi sang grayscale
308         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
309
310         # Lưu ảnh
311         cv2.imwrite('image.jpg', frame)
312
313         # Gửi yêu cầu đến API
314         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
315
316         # In kết quả
317         print(response.json())
318
319     # Nhấn 'Q' để thoát
320     if cv2.waitKey(1) & 0xFF == ord('Q'):
321         break
322
323     # Nhấn 'F' để tăng độ nhạy cảm
324     if cv2.waitKey(1) & 0xFF == ord('F'):
325         sensitivity += 0.05
326
327     # Nhấn 'B' để giảm độ nhạy cảm
328     if cv2.waitKey(1) & 0xFF == ord('B'):
329         sensitivity -= 0.05
330
331     # Nhấn 'P' để tăng độ chính xác
332     if cv2.waitKey(1) & 0xFF == ord('P'):
333         accuracy += 0.05
334
335     # Nhấn 'N' để giảm độ chính xác
336     if cv2.waitKey(1) & 0xFF == ord('N'):
337         accuracy -= 0.05
338
339     # Nhấn 'R' để reset
340     if cv2.waitKey(1) & 0xFF == ord('R'):
341         reset()
342
343     # Nhấn 'Q' để thoát
344     if cv2.waitKey(1) & 0xFF == ord('Q'):
345         break
346
347     # Nhấn 'M' để tắt/mở camera
348     if cv2.waitKey(1) & 0xFF == ord('M'):
349         cap.isOpened() = not cap.isOpened()
350
351     # Nhấn 'C' để chụp ảnh
352     if cv2.waitKey(1) & 0xFF == ord('C'):
353         # Lấy khung hình
354         frame = display_frame[y:y+h, x:x+w]
355
356         # Chuyển đổi sang grayscale
357         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
358
359         # Lưu ảnh
360         cv2.imwrite('image.jpg', frame)
361
362         # Gửi yêu cầu đến API
363         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
364
365         # In kết quả
366         print(response.json())
367
368     # Nhấn 'E' để tăng độ chính xác
369     if cv2.waitKey(1) & 0xFF == ord('E'):
370         accuracy += 0.05
371
372     # Nhấn 'D' để giảm độ chính xác
373     if cv2.waitKey(1) & 0xFF == ord('D'):
374         accuracy -= 0.05
375
376     # Nhấn 'A' để tăng độ nhạy cảm
377     if cv2.waitKey(1) & 0xFF == ord('A'):
378         sensitivity += 0.05
379
380     # Nhấn 'Z' để giảm độ nhạy cảm
381     if cv2.waitKey(1) & 0xFF == ord('Z'):
382         sensitivity -= 0.05
383
384     # Nhấn 'S' để chụp ảnh và gửi
385     if cv2.waitKey(1) & 0xFF == ord('S'):
386         # Lấy khung hình
387         frame = display_frame[y:y+h, x:x+w]
388
389         # Chuyển đổi sang grayscale
390         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
391
392         # Lưu ảnh
393         cv2.imwrite('image.jpg', frame)
394
395         # Gửi yêu cầu đến API
396         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
397
398         # In kết quả
399         print(response.json())
400
401     # Nhấn 'Q' để thoát
402     if cv2.waitKey(1) & 0xFF == ord('Q'):
403         break
404
405     # Nhấn 'F' để tăng độ nhạy cảm
406     if cv2.waitKey(1) & 0xFF == ord('F'):
407         sensitivity += 0.05
408
409     # Nhấn 'B' để giảm độ nhạy cảm
410     if cv2.waitKey(1) & 0xFF == ord('B'):
411         sensitivity -= 0.05
412
413     # Nhấn 'P' để tăng độ chính xác
414     if cv2.waitKey(1) & 0xFF == ord('P'):
415         accuracy += 0.05
416
417     # Nhấn 'N' để giảm độ chính xác
418     if cv2.waitKey(1) & 0xFF == ord('N'):
419         accuracy -= 0.05
420
421     # Nhấn 'R' để reset
422     if cv2.waitKey(1) & 0xFF == ord('R'):
423         reset()
424
425     # Nhấn 'Q' để thoát
426     if cv2.waitKey(1) & 0xFF == ord('Q'):
427         break
428
429     # Nhấn 'M' để tắt/mở camera
430     if cv2.waitKey(1) & 0xFF == ord('M'):
431         cap.isOpened() = not cap.isOpened()
432
433     # Nhấn 'C' để chụp ảnh
434     if cv2.waitKey(1) & 0xFF == ord('C'):
435         # Lấy khung hình
436         frame = display_frame[y:y+h, x:x+w]
437
438         # Chuyển đổi sang grayscale
439         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
440
441         # Lưu ảnh
442         cv2.imwrite('image.jpg', frame)
443
444         # Gửi yêu cầu đến API
445         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
446
447         # In kết quả
448         print(response.json())
449
450     # Nhấn 'E' để tăng độ chính xác
451     if cv2.waitKey(1) & 0xFF == ord('E'):
452         accuracy += 0.05
453
454     # Nhấn 'D' để giảm độ chính xác
455     if cv2.waitKey(1) & 0xFF == ord('D'):
456         accuracy -= 0.05
457
458     # Nhấn 'A' để tăng độ nhạy cảm
459     if cv2.waitKey(1) & 0xFF == ord('A'):
460         sensitivity += 0.05
461
462     # Nhấn 'Z' để giảm độ nhạy cảm
463     if cv2.waitKey(1) & 0xFF == ord('Z'):
464         sensitivity -= 0.05
465
466     # Nhấn 'S' để chụp ảnh và gửi
467     if cv2.waitKey(1) & 0xFF == ord('S'):
468         # Lấy khung hình
469         frame = display_frame[y:y+h, x:x+w]
470
471         # Chuyển đổi sang grayscale
472         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
473
474         # Lưu ảnh
475         cv2.imwrite('image.jpg', frame)
476
477         # Gửi yêu cầu đến API
478         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
479
480         # In kết quả
481         print(response.json())
482
483     # Nhấn 'Q' để thoát
484     if cv2.waitKey(1) & 0xFF == ord('Q'):
485         break
486
487     # Nhấn 'F' để tăng độ nhạy cảm
488     if cv2.waitKey(1) & 0xFF == ord('F'):
489         sensitivity += 0.05
490
491     # Nhấn 'B' để giảm độ nhạy cảm
492     if cv2.waitKey(1) & 0xFF == ord('B'):
493         sensitivity -= 0.05
494
495     # Nhấn 'P' để tăng độ chính xác
496     if cv2.waitKey(1) & 0xFF == ord('P'):
497         accuracy += 0.05
498
499     # Nhấn 'N' để giảm độ chính xác
500     if cv2.waitKey(1) & 0xFF == ord('N'):
501         accuracy -= 0.05
502
503     # Nhấn 'R' để reset
504     if cv2.waitKey(1) & 0xFF == ord('R'):
505         reset()
506
507     # Nhấn 'Q' để thoát
508     if cv2.waitKey(1) & 0xFF == ord('Q'):
509         break
510
511     # Nhấn 'M' để tắt/mở camera
512     if cv2.waitKey(1) & 0xFF == ord('M'):
513         cap.isOpened() = not cap.isOpened()
514
515     # Nhấn 'C' để chụp ảnh
516     if cv2.waitKey(1) & 0xFF == ord('C'):
517         # Lấy khung hình
518         frame = display_frame[y:y+h, x:x+w]
519
520         # Chuyển đổi sang grayscale
521         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
522
523         # Lưu ảnh
524         cv2.imwrite('image.jpg', frame)
525
526         # Gửi yêu cầu đến API
527         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
528
529         # In kết quả
530         print(response.json())
531
532     # Nhấn 'E' để tăng độ chính xác
533     if cv2.waitKey(1) & 0xFF == ord('E'):
534         accuracy += 0.05
535
536     # Nhấn 'D' để giảm độ chính xác
537     if cv2.waitKey(1) & 0xFF == ord('D'):
538         accuracy -= 0.05
539
540     # Nhấn 'A' để tăng độ nhạy cảm
541     if cv2.waitKey(1) & 0xFF == ord('A'):
542         sensitivity += 0.05
543
544     # Nhấn 'Z' để giảm độ nhạy cảm
545     if cv2.waitKey(1) & 0xFF == ord('Z'):
546         sensitivity -= 0.05
547
548     # Nhấn 'S' để chụp ảnh và gửi
549     if cv2.waitKey(1) & 0xFF == ord('S'):
550         # Lấy khung hình
551         frame = display_frame[y:y+h, x:x+w]
552
553         # Chuyển đổi sang grayscale
554         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
555
556         # Lưu ảnh
557         cv2.imwrite('image.jpg', frame)
558
559         # Gửi yêu cầu đến API
560         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
561
562         # In kết quả
563         print(response.json())
564
565     # Nhấn 'Q' để thoát
566     if cv2.waitKey(1) & 0xFF == ord('Q'):
567         break
568
569     # Nhấn 'F' để tăng độ nhạy cảm
570     if cv2.waitKey(1) & 0xFF == ord('F'):
571         sensitivity += 0.05
572
573     # Nhấn 'B' để giảm độ nhạy cảm
574     if cv2.waitKey(1) & 0xFF == ord('B'):
575         sensitivity -= 0.05
576
577     # Nhấn 'P' để tăng độ chính xác
578     if cv2.waitKey(1) & 0xFF == ord('P'):
579         accuracy += 0.05
580
581     # Nhấn 'N' để giảm độ chính xác
582     if cv2.waitKey(1) & 0xFF == ord('N'):
583         accuracy -= 0.05
584
585     # Nhấn 'R' để reset
586     if cv2.waitKey(1) & 0xFF == ord('R'):
587         reset()
588
589     # Nhấn 'Q' để thoát
590     if cv2.waitKey(1) & 0xFF == ord('Q'):
591         break
592
593     # Nhấn 'M' để tắt/mở camera
594     if cv2.waitKey(1) & 0xFF == ord('M'):
595         cap.isOpened() = not cap.isOpened()
596
597     # Nhấn 'C' để chụp ảnh
598     if cv2.waitKey(1) & 0xFF == ord('C'):
599         # Lấy khung hình
600         frame = display_frame[y:y+h, x:x+w]
601
602         # Chuyển đổi sang grayscale
603         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
604
605         # Lưu ảnh
606         cv2.imwrite('image.jpg', frame)
607
608         # Gửi yêu cầu đến API
609         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
610
611         # In kết quả
612         print(response.json())
613
614     # Nhấn 'E' để tăng độ chính xác
615     if cv2.waitKey(1) & 0xFF == ord('E'):
616         accuracy += 0.05
617
618     # Nhấn 'D' để giảm độ chính xác
619     if cv2.waitKey(1) & 0xFF == ord('D'):
620         accuracy -= 0.05
621
622     # Nhấn 'A' để tăng độ nhạy cảm
623     if cv2.waitKey(1) & 0xFF == ord('A'):
624         sensitivity += 0.05
625
626     # Nhấn 'Z' để giảm độ nhạy cảm
627     if cv2.waitKey(1) & 0xFF == ord('Z'):
628         sensitivity -= 0.05
629
630     # Nhấn 'S' để chụp ảnh và gửi
631     if cv2.waitKey(1) & 0xFF == ord('S'):
632         # Lấy khung hình
633         frame = display_frame[y:y+h, x:x+w]
634
635         # Chuyển đổi sang grayscale
636         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
637
638         # Lưu ảnh
639         cv2.imwrite('image.jpg', frame)
640
641         # Gửi yêu cầu đến API
642         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
643
644         # In kết quả
645         print(response.json())
646
647     # Nhấn 'Q' để thoát
648     if cv2.waitKey(1) & 0xFF == ord('Q'):
649         break
650
651     # Nhấn 'F' để tăng độ nhạy cảm
652     if cv2.waitKey(1) & 0xFF == ord('F'):
653         sensitivity += 0.05
654
655     # Nhấn 'B' để giảm độ nhạy cảm
656     if cv2.waitKey(1) & 0xFF == ord('B'):
657         sensitivity -= 0.05
658
659     # Nhấn 'P' để tăng độ chính xác
660     if cv2.waitKey(1) & 0xFF == ord('P'):
661         accuracy += 0.05
662
663     # Nhấn 'N' để giảm độ chính xác
664     if cv2.waitKey(1) & 0xFF == ord('N'):
665         accuracy -= 0.05
666
667     # Nhấn 'R' để reset
668     if cv2.waitKey(1) & 0xFF == ord('R'):
669         reset()
670
671     # Nhấn 'Q' để thoát
672     if cv2.waitKey(1) & 0xFF == ord('Q'):
673         break
674
675     # Nhấn 'M' để tắt/mở camera
676     if cv2.waitKey(1) & 0xFF == ord('M'):
677         cap.isOpened() = not cap.isOpened()
678
679     # Nhấn 'C' để chụp ảnh
680     if cv2.waitKey(1) & 0xFF == ord('C'):
681         # Lấy khung hình
682         frame = display_frame[y:y+h, x:x+w]
683
684         # Chuyển đổi sang grayscale
685         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
686
687         # Lưu ảnh
688         cv2.imwrite('image.jpg', frame)
689
690         # Gửi yêu cầu đến API
691         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
692
693         # In kết quả
694         print(response.json())
695
696     # Nhấn 'E' để tăng độ chính xác
697     if cv2.waitKey(1) & 0xFF == ord('E'):
698         accuracy += 0.05
699
700     # Nhấn 'D' để giảm độ chính xác
701     if cv2.waitKey(1) & 0xFF == ord('D'):
702         accuracy -= 0.05
703
704     # Nhấn 'A' để tăng độ nhạy cảm
705     if cv2.waitKey(1) & 0xFF == ord('A'):
706         sensitivity += 0.05
707
708     # Nhấn 'Z' để giảm độ nhạy cảm
709     if cv2.waitKey(1) & 0xFF == ord('Z'):
710         sensitivity -= 0.05
711
712     # Nhấn 'S' để chụp ảnh và gửi
713     if cv2.waitKey(1) & 0xFF == ord('S'):
714         # Lấy khung hình
715         frame = display_frame[y:y+h, x:x+w]
716
717         # Chuyển đổi sang grayscale
718         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
719
720         # Lưu ảnh
721         cv2.imwrite('image.jpg', frame)
722
723         # Gửi yêu cầu đến API
724         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
725
726         # In kết quả
727         print(response.json())
728
729     # Nhấn 'Q' để thoát
730     if cv2.waitKey(1) & 0xFF == ord('Q'):
731         break
732
733     # Nhấn 'F' để tăng độ nhạy cảm
734     if cv2.waitKey(1) & 0xFF == ord('F'):
735         sensitivity += 0.05
736
737     # Nhấn 'B' để giảm độ nhạy cảm
738     if cv2.waitKey(1) & 0xFF == ord('B'):
739         sensitivity -= 0.05
740
741     # Nhấn 'P' để tăng độ chính xác
742     if cv2.waitKey(1) & 0xFF == ord('P'):
743         accuracy += 0.05
744
745     # Nhấn 'N' để giảm độ chính xác
746     if cv2.waitKey(1) & 0xFF == ord('N'):
747         accuracy -= 0.05
748
749     # Nhấn 'R' để reset
750     if cv2.waitKey(1) & 0xFF == ord('R'):
751         reset()
752
753     # Nhấn 'Q' để thoát
754     if cv2.waitKey(1) & 0xFF == ord('Q'):
755         break
756
757     # Nhấn 'M' để tắt/mở camera
758     if cv2.waitKey(1) & 0xFF == ord('M'):
759         cap.isOpened() = not cap.isOpened()
760
761     # Nhấn 'C' để chụp ảnh
762     if cv2.waitKey(1) & 0xFF == ord('C'):
763         # Lấy khung hình
764         frame = display_frame[y:y+h, x:x+w]
765
766         # Chuyển đổi sang grayscale
767         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
768
769         # Lưu ảnh
770         cv2.imwrite('image.jpg', frame)
771
772         # Gửi yêu cầu đến API
773         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
774
775         # In kết quả
776         print(response.json())
777
778     # Nhấn 'E' để tăng độ chính xác
779     if cv2.waitKey(1) & 0xFF == ord('E'):
780         accuracy += 0.05
781
782     # Nhấn 'D' để giảm độ chính xác
783     if cv2.waitKey(1) & 0xFF == ord('D'):
784         accuracy -= 0.05
785
786     # Nhấn 'A' để tăng độ nhạy cảm
787     if cv2.waitKey(1) & 0xFF == ord('A'):
788         sensitivity += 0.05
789
790     # Nhấn 'Z' để giảm độ nhạy cảm
791     if cv2.waitKey(1) & 0xFF == ord('Z'):
792         sensitivity -= 0.05
793
794     # Nhấn 'S' để chụp ảnh và gửi
795     if cv2.waitKey(1) & 0xFF == ord('S'):
796         # Lấy khung hình
797         frame = display_frame[y:y+h, x:x+w]
798
799         # Chuyển đổi sang grayscale
800         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
801
802         # Lưu ảnh
803         cv2.imwrite('image.jpg', frame)
804
805         # Gửi yêu cầu đến API
806         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
807
808         # In kết quả
809         print(response.json())
810
811     # Nhấn 'Q' để thoát
812     if cv2.waitKey(1) & 0xFF == ord('Q'):
813         break
814
815     # Nhấn 'F' để tăng độ nhạy cảm
816     if cv2.waitKey(1) & 0xFF == ord('F'):
817         sensitivity += 0.05
818
819     # Nhấn 'B' để giảm độ nhạy cảm
820     if cv2.waitKey(1) & 0xFF == ord('B'):
821         sensitivity -= 0.05
822
823     # Nhấn 'P' để tăng độ chính xác
824     if cv2.waitKey(1) & 0xFF == ord('P'):
825         accuracy += 0.05
826
827     # Nhấn 'N' để giảm độ chính xác
828     if cv2.waitKey(1) & 0xFF == ord('N'):
829         accuracy -= 0.05
830
831     # Nhấn 'R' để reset
832     if cv2.waitKey(1) & 0xFF == ord('R'):
833         reset()
834
835     # Nhấn 'Q' để thoát
836     if cv2.waitKey(1) & 0xFF == ord('Q'):
837         break
838
839     # Nhấn 'M' để tắt/mở camera
840     if cv2.waitKey(1) & 0xFF == ord('M'):
841         cap.isOpened() = not cap.isOpened()
842
843     # Nhấn 'C' để chụp ảnh
844     if cv2.waitKey(1) & 0xFF == ord('C'):
845         # Lấy khung hình
846         frame = display_frame[y:y+h, x:x+w]
847
848         # Chuyển đổi sang grayscale
849         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
850
851         # Lưu ảnh
852         cv2.imwrite('image.jpg', frame)
853
854         # Gửi yêu cầu đến API
855         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
856
857         # In kết quả
858         print(response.json())
859
860     # Nhấn 'E' để tăng độ chính xác
861     if cv2.waitKey(1) & 0xFF == ord('E'):
862         accuracy += 0.05
863
864     # Nhấn 'D' để giảm độ chính xác
865     if cv2.waitKey(1) & 0xFF == ord('D'):
866         accuracy -= 0.05
867
868     # Nhấn 'A' để tăng độ nhạy cảm
869     if cv2.waitKey(1) & 0xFF == ord('A'):
870         sensitivity += 0.05
871
872     # Nhấn 'Z' để giảm độ nhạy cảm
873     if cv2.waitKey(1) & 0xFF == ord('Z'):
874         sensitivity -= 0.05
875
876     # Nhấn 'S' để chụp ảnh và gửi
877     if cv2.waitKey(1) & 0xFF == ord('S'):
878         # Lấy khung hình
879         frame = display_frame[y:y+h, x:x+w]
880
881         # Chuyển đổi sang grayscale
882         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
883
884         # Lưu ảnh
885         cv2.imwrite('image.jpg', frame)
886
887         # Gửi yêu cầu đến API
888         response = requests.post(LOCAL_API_URL, files={'image': open('image.jpg', 'rb')})
889
890         # In kết quả
891         print(response.json())
892
893     # Nhấn 'Q' để thoát
894     if cv2.waitKey(1) & 0xFF == ord('Q'):
895         break
896
897     # Nhấn 'F' để tăng độ nhạy cảm
898     if cv2.waitKey(1) & 0xFF == ord('F'):
899         sensitivity += 0.05
900
901     # Nhấn 'B' để giảm độ nhạy cảm
902     if cv2.waitKey(1) & 0xFF == ord('B'):
903         sensitivity -= 0.05
904
905     # Nhấn 'P' để tăng độ chính xác
906     if cv2.waitKey(1) & 0xFF == ord('P'):
907         accuracy += 0.05
908
909     # Nhấn 'N' để giảm độ chính xác
910     if cv2.waitKey(1) & 0xFF == ord('N'):
911         accuracy -= 0.05
912
913     # Nhấn 'R' để reset
914     if cv2.waitKey(1) & 0xFF == ord('R'):
915         reset()
916
917     # Nhấn 'Q' để thoát
918     if cv2.waitKey(1) & 0xFF == ord('Q'):
919         break
920
921     # Nhấn 'M' để tắt/mở camera
922     if cv2.waitKey(1) & 0xFF == ord('M'):
923         cap.isOpened() = not cap.isOpened()
924
925     # Nhấn 'C' để chụp ảnh
926     if cv2.waitKey(1) & 0xFF == ord('C'):
927         # Lấy khung hình
928         frame = display_frame[y:y+h, x:x+w]
929
930         # Chuyển đổi sang grayscale
931         frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
932
933         #
```

The screenshot shows a developer's environment with multiple windows open:

- File Explorer:** Shows a project structure with files like Dockerfile, client.py, server.py, and util.py.
- Terminal:** Running a command that outputs a list of detected faces with bounding boxes and labels (e.g., 'without mask').
- Browser:** A local server interface titled "Face Mask Detector - Client (Local Server)" showing a video feed of a person's face. A red bounding box highlights the person's head, and the text "without mask" is displayed in red at the top of the frame.
- Code Editor:** An integrated code editor showing the Python script for the Face Mask Detector, which includes imports for cv2, requests, numpy, json, time, and os, along with logic for capturing video from the camera and detecting faces.
- Right Sidebar:** Includes a "Build with agent mode" section with a "Let's get started" button, a "Add context" input field, and buttons for "Build Workspace" and "Show Config".