import base64

import cv2

import numpy as np

from fastapi import FastAPI, WebSocket

from fastapi.middleware.cors import CORSMiddleware

from ultralytics import YOLO

app = FastAPI()

# CORS for frontend access

app.add\_middleware(

    CORSMiddleware,

    allow\_origins=["\*"],

    allow\_credentials=True,

    allow\_methods=["\*"],

    allow\_headers=["\*"],

)

# Load your YOLOv8 model (replace with your trained weights path)

model = YOLO("best.pt")  # change this to your trained model file

def decode\_base64\_image(data: str):

    try:

        img\_bytes = base64.b64decode(data)

        np\_arr = np.frombuffer(img\_bytes, np.uint8)

        frame = cv2.imdecode(np\_arr, cv2.IMREAD\_COLOR)

        if frame is None:

            print("⚠️ Empty frame received")

        return frame

    except Exception as e:

        print("❌ Error decoding frame:", e)

        return None

@app.websocket("/ws")

async def websocket\_endpoint(ws: WebSocket):

    await ws.accept()

    print("✅ WebSocket connection open")

    try:

        while True:

            data = await ws.receive\_text()

            frame = decode\_base64\_image(data)

            if frame is None:

                continue

            # Run YOLOv8 keypoint detection

            results = model(frame, verbose=False)

            bboxes, keypoints = [], []

            for r in results:

                if r.boxes is not None:

                    for box in r.boxes.xyxy.tolist():

                        bboxes.append(box)

                if r.keypoints is not None:

                    for kp in r.keypoints.xy.tolist():

                        keypoints.append(kp)

            # Send back detections

            await ws.send\_json({"bbox": bboxes, "keypoints": keypoints})

    except Exception as e:

        print("⚠️ Connection closed:", e)

    finally:

        await ws.close()

        print("❌ WebSocket disconnected")