```
import os, json, base64, uuid, re, sqlite3
from datetime import datetime
from flask import Flask, request, jsonify, send_from_directory
from flask_cors import CORS
import anthropic

app = Flask(__name__, static_folder="static")
CORS(app)

DB_PATH = os.environ.get("DB_PATH", "trips.db")
ANTHROPIC_KEY = os.environ.get("ANTHROPIC_API_KEY", "")

def get_db():
    con = sqlite3.connect(DB_PATH)
    con.row_factory = sqlite3.Row
    return con

def init_db():
    con = get_db()
    con.execute("""CREATE TABLE IF NOT EXISTS trips (
        id TEXT PRIMARY KEY, date TEXT, time TEXT, type TEXT,
        price REAL, pickup TEXT, dropoff TEXT,
        zone_pickup TEXT, zone_dropoff TEXT,
        km REAL, duration INTEGER, rating REAL, boost REAL,
        notes TEXT, added_by TEXT, created_at TEXT)""")
    con.commit(); con.close()

init_db()

def extract_from_image(b64, mt):
    client = anthropic.Anthropic(api_key=ANTHROPIC_KEY)
    today = datetime.now().strftime("%Y-%m-%d")
    msg = client.messages.create(
        model="claude-opus-4-5", max_tokens=600,
        messages=[{"role":"user","content":[
            {"type":"image","source":{"type":"base64","media_type":mt,"data":b64}
            {"type":"text","text":f'إضافي JSON استخرج بيانات إيصال أوبر هذا وأرجع
        ]}])
    text = msg.content[0].text.strip()
    m = re.search(r'\{[\s\S]*\}', text)
    if not m: raise ValueError("لم يُعثر على JSON: " + text[:80])
    return json.loads(m.group())

@app.route("/")
```

```
def index():
    return send_from_directory("static", "index.html")

@app.route("/api/trips")
def api_trips():
    con = get_db()
    rows = con.execute("SELECT * FROM trips ORDER BY date DESC, time DESC").fetch
    con.close()
    return jsonify([dict(r) for r in rows])

@app.route("/api/upload", methods=["POST"])
def api_upload():
    if not ANTHROPIC_KEY:
        return jsonify([{"file":"?","status":"error","error":"ANTHROPIC_API_KEY ر
    files = request.files.getlist("images")
    by = request.form.get("added_by", "مجهول")
    results = []
    for f in files:
        try:
            b64 = base64.b64encode(f.read()).decode()
            mt = f.content_type or "image/jpeg"
            if any(x in mt for x in ["heic","heif"]): mt = "image/jpeg"
            if mt not in ["image/jpeg","image/png","image/gif","image/webp"]: mt
            data = extract_from_image(b64, mt)
            trip = {
                "id": str(uuid.uuid4()),
                "date": datetime.now().strftime("%Y-%m-%d"),
                "time": data.get("time",""),
                "type": data.get("type",""),
                "price": float(data.get("price",0)),
                "pickup": data.get("pickup",""),
                "dropoff": data.get("dropoff",""),
                "zone_pickup": str(data.get("zone_pickup","4")),
                "zone_dropoff": str(data.get("zone_dropoff","5")),
                "km": float(data.get("km",0)),
                "duration": int(data.get("duration_minutes",0)),
                "rating": float(data.get("rating",0)),
                "boost": float(data.get("boost",0)),
                "notes": data.get("notes",""),
                "added_by": by,
                "created_at": datetime.now().isoformat()
            }
            con = get_db()
            con.execute("INSERT INTO trips VALUES (:id,:date,:time,:type,:price,:
            con.commit(); con.close()
            results.append({"file": f.filename, "status": "ok", "trip": trip})
        except Exception as e:
```

```
            results.append({"file": f.filename, "status": "error", "error": str(e
    return jsonify(results)

@app.route("/api/trip", methods=["POST"])
def api_manual():
    t = request.json
    t["id"] = str(uuid.uuid4())
    t["created_at"] = datetime.now().isoformat()
    con = get_db()
    con.execute("INSERT INTO trips VALUES (:id,:date,:time,:type,:price,:pickup,:
    con.commit(); con.close()
    return jsonify({"ok": True})

@app.route("/api/trips/<tid>", methods=["DELETE"])
def api_del(tid):
    con = get_db()
    con.execute("DELETE FROM trips WHERE id=?", (tid,))
    con.commit(); con.close()
    return jsonify({"ok": True})

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=int(os.environ.get("PORT", 5000)))
```