```
# app.py (Backend)
from flask import Flask, request, jsonify
import cv2
import numpy as np
from deepface import DeepFace

app = Flask(__name__)

# Temporary storage for demo (replace with a database)
face_db = []

@app.route('/detect', methods=['POST'])
def detect_face():
    # Read image from request
    file = request.files['image']
    img = cv2.imdecode(np.frombuffer(file.read(), np.uint8), cv2.IMREAD_COLOR)

    # Detect faces and generate embeddings
    try:
        detected_face = DeepFace.detectFace(img, detector_backend='opencv')
        embedding = DeepFace.represent(img, model_name='Facenet')[0]["embedding"]

        # Store embedding (hash this in production)
        face_db.append(embedding)
        return jsonify({"status": "success", "embedding": embedding})

    except:
        return jsonify({"status": "no face detected"})

@app.route('/search', methods=['POST'])
def search_face():
    query_embedding = request.json['embedding']
    matches = []

    # Compare with stored embeddings (demo: Euclidean distance)
    for idx, emb in enumerate(face_db):
        distance = np.linalg.norm(np.array(emb) - np.array(query_embedding))
        if distance < 0.6:  # Threshold for similarity
            matches.append({"id": idx, "confidence": 1 - distance})

    return jsonify({"matches": matches})

if __name__ == '__main__':
    app.run(debug=True)
```