API v1.0.0

Face Scan REST API

Official Technical Reference & Endpoint Specification

Platform: AWS Elastic Beanstalk (Python 3.11)
Model Engine: DeepFace / Facenet
Output: JSON Application

> **Production Endpoint:** `http://face-scan-311.eba-yxqrprdz.ap-south-1.elasticbeanstalk.com`
> **Format:** All POST request bodies and responses use standard `application/json` encoding.

1. Overview & Architecture

The Face Scan API provides high-speed, deep-learning based facial recognition, feature extraction, and 1:1 identity verification. It detects faces, computes mathematical facial embedding vectors, and compares face signatures using Cosine similarity metrics.

Feature	Specification	Details
Default Engine	Facenet	Compact, high-accuracy neural network (128-d vector output)
Face Detector	OpenCV Haar Cascade	Fast, real-time face localization and bounding box extraction
Distance Metric	Cosine Distance	0.00 (Identical) to 1.00 (Completely Different)
Verification Threshold	0.40	Pair distance ≤ 0.40 indicates the same person
Image Inputs	Base64 or Server Path	Supports standard JPEG, PNG, WEBP encoded images

2. Supported AI Models

You may optionally specify any of the supported recognition backends using the `model` parameter:

Model Name	Vector Dimension	Accuracy	Best Use Case
Facenet *(Default)*	128	99.6%	Recommended general-purpose face recognition and fast matching
Facenet512	512	99.6%	High-precision biometric matching
VGG-Face	2622	98.8%	Large embedding size, robust feature representation
ArcFace	512	99.5%	Enterprise-grade high-security identification

Model Name	Vector Dimension	Accuracy	Best Use Case
OpenFace	128	92.9%	Lightweight mobile/edge computing

3. API Endpoints Specification

`GET` `/` — Service Status Check

Returns the operational status of the service and the active face model.

Response Schema (200 OK)

Field	Type	Description
`status`	string	Always returns `"running"` when service is active
`service`	string	Service name identifier (`"face-scan-api"`)
`model`	string	Default loaded model (`"Facenet"`)

```
{ "status": "running", "service": "face-scan-api", "model": "Facenet" }
```

`GET` `/health` — Health Check

Used by load balancers, container orchestrators, and uptime monitors.

```
{ "status": "healthy", "model": "Facenet" }
```

POST /represent

Facial Embedding & Coordinates Extraction

Detects a face in the input image and extracts its numerical embedding vector and bounding box coordinates.

Request Body

Field	Type	Requirement	Description
`img_base64`	string	Required*	Base64-encoded image string (with or without data URI scheme)
`img_path`	string	Required*	Absolute file path on server (*Provide either `img_base64` or `img_path`)
`model`	string	Optional	Model name. Defaults to `"Facenet"`

Response Body (200 OK)

Field	Type	Description
`success`	boolean	`true` when extraction succeeds
`model`	string	Name of the model used
`facial_area`	object	Coordinates: `x`, `y`, `w`, `h`, `left_eye`, `right_eye`
`embedding`	array[float]	Normalized float vector representing facial biometric features


```
{ "success": true, "model": "Facenet", "facial_area": { "x": 128, "y": 85, "w":
142, "h": 142, "left_eye": [168, 134], "right_eye": [226, 132] }, "embedding":
[-0.03451, 0.08912, 0.01243, ...] }
```

POST `/verify` — 1:1 Face Verification & Matching

Compares two facial images to verify whether they represent the same individual.

Request Body

Field	Type	Requirement	Description
`img1_base64`	string	Required*	Base64 string of reference face image
`img1_path`	string	Required*	Local path to reference image (*Use either base64 or path)
`img2_base64`	string	Required*	Base64 string of target face image to compare
`img2_path`	string	Required*	Local path to target image (*Use either base64 or path)
`model`	string	Optional	Recognition model. Defaults to `"Facenet"`

Response Body (200 OK)

Field	Type	Description
`success`	boolean	`true` when verification executes successfully
`verified`	boolean	`true` if faces match (`distance ≤ threshold`), else `false`
`distance`	float	Cosine distance score (Lower score = higher similarity)
`threshold`	float	Decision threshold boundary (`0.40` for Facenet)

```
{ "success": true, "verified": true, "distance": 0.2184, "threshold": 0.40 }
```

4. Error Codes & Handling

Status Code	Reason	Sample Error Response
400 Bad Request	Missing required image parameters	`{"success": false, "message": "No image provided"}`
422 Unprocessable Entity	No face detected in submitted image	`{"success": false, "message": "No face detected in the image"}`
500 Internal Server Error	Unexpected server or processing error	`{"success": false, "message": "<error details>"}`

5. Best Practices for High Accuracy

Image Lighting: Ensure even lighting across the subject's face without harsh shadows or backlighting.
Face Orientation: Frontal faces within ±15 degrees produce optimal similarity scores.
Image Resolution: Recommended face resolution is at least 160×160 pixels.
Base64 Formatting: Standard Base64 strings with or without `` prefixes are automatically parsed.