

ITIS6177 – Final Project

Face API

Name: Satwik Rao

ID: 801257585

Email: srao26@uncc.edu

Contents

Introduction:	2
Implementation:	2
Face Detect:	2
Face Verify:	3
Instructions to Execute the API:	5
Executing with Postman-	5
Executing with Swagger UI-	8
References:	11

Introduction:

The Azure Face service provides AI algorithms that detect, recognize, and analyze human faces in images. Facial recognition software is important in many different scenarios, such as identity verification, touchless access control, and face blurring for privacy.

This particular project will act as an interface to execute the Azure API without explicitly registering for the Azure API. The services provided are Face Detection and Face Verification.

Face Detection:

The Detect API detects human faces in an image and returns the rectangle coordinates of their locations. It also returns a unique ID that represents the stored face data. This is used in later operations to identify or verify faces. Optionally, face detection can extract a set of face-related attributes, such as head pose, age, emotion, facial hair, and glasses. These attributes are general predictions, not actual classifications. Some attributes are useful to ensure that your application is getting high-quality face data when users add themselves to a Face service

Face Verify:

The Face Verify API will be used to verify whether two faces belong to a same person or whether one face belongs to a person. Verification is also a "one-to-one" matching of a face in an image to a single face from a secure repository or photo to verify that they're the same individual.

Implementation:

This API is implemented using REST. The Request and Response for the endpoints are as follows

Face Detect:

Method - Post

Request Body: The URL of the image to be detected should be passed in the request body.

JSON fields in the request body:

Fields	Type	Description
url	String	URL of input image

Example:

```
{  
  "url": "https://c.ndtvimg.com/2022-02/2jionh7_dhoni_625x300_18_February_22.jpg"  
}
```

Validations or points to be considered for giving the image URL:

1. The supported formats of the image are JPEG, PNG, GIF and BMP.
2. The allowed image file size is from 1KB to 6MB.
3. The minimum detectable face size is 36x36 pixels in an image no larger than 1920x1080 pixels. Images with dimensions higher than 1920x1080 pixels will need a proportionally larger minimum face size.
4. Up to 100 faces can be returned for an image.

Response:

Code 200 – Success:

A successful call returns an array of face entries ranked by face rectangle size in descending order. An empty response indicates no faces detected.

JSON fields in response body:

Fields	Type	Description
faceId	String	Unique faceId of the detected face, created by detection API and it will expire 24 hours after the detection call.
faceRectangle	Object	A rectangle area for the face location on image.
faceLandmarks	Object	An array of 27-point face landmarks pointing to the important positions of face components. To return this, it requires "returnFaceLandmarks" parameter to be true
faceAttributes	Object	Face attributes like age, gender, smile intensity etc..

Code 400 – Bad Request: Error in the JSON request or invalid URL or invalid image size

Code 401 – Invalid key or user account issue: Access denied due to incorrect subscription key or user/plan is blocked.

Code 408 – Request time out: Operation exceeds maximum execution time

Code 404 – Resource not found

Code 500 – Internal server error.

Face Verify:

Method - Post

Request body:

JSON fields in face-to-face verification request body are as follows

Fields	Type	Description
faceId1	String	faceId of one face, comes from Face - Detect.
faceId2	String	faceId of another face, comes from Face - Detect.

Example:

```
{  
  "faceId1": "26d29182-5c26-473c-b8ed-dc5f657bbb8d",  
  "faceId2": "4b12fae3-d712-4fb9-9a69-f69224b6518e"  
}
```

Please note that Face detect API to be executed first to get face Id's.

Validations or points to be considered for giving the image URL:

1. The faceId's should come from Face detection API.

Response:

Code 200 – Success: A successful call returns the verification result.

JSON fields in response body:

Fields	Type	Description
isIdentical	Boolean	True if the two faces belong to the same person or the face belongs to the person, otherwise false.
confidence	Number	A number indicates the similarity confidence of whether two faces belong to the same person, or whether the face belongs to the person. By default, isIdentical is set to True if similarity confidence is greater than or equal to 0.5.

Example:

```
{  
  "isIdentical": true,  
  "confidence": 0.9  
}
```

Code 400 – Bad Request: Error in the JSON request or invalid Face Id

Code 401 – Invalid key or user account issue: Access denied due to incorrect subscription key or user/plan is blocked.

Code 408 – Request time out: Operation exceeds maximum execution time

Code 404 – Resource not found or Face Id not found

Code 500 – Internal server error.

Instructions to Execute the API:

Executing with Postman-

A.Face Detection end point

- 1.Open Postman and create a new Post request.
- 2.Enter the Request as URL as <http://167.172.151.164:3000/detect>
- 3.Goto Body and check x-www-form-urlencoded radio button.
- 4.Enter the key as “url” and give the URL of the image to be detected in the value field.

The screenshot shows the Postman interface for a POST request. The URL is set to `http://167.172.151.164:3000/detect`. The 'Body' tab is selected, and the 'x-www-form-urlencoded' radio button is chosen. A key-value pair is added with the key 'url' and the value `https://c.ndtvimg.com/2022-02/2jionh7_dhoni_625x300_18_February_22.jpg`. The 'Response' section is visible at the bottom.

- 5.Click on Send to get the response of face detection attributes

The screenshot shows the Postman interface displaying the JSON response of the face detection API. The response is a JSON object containing face attributes:

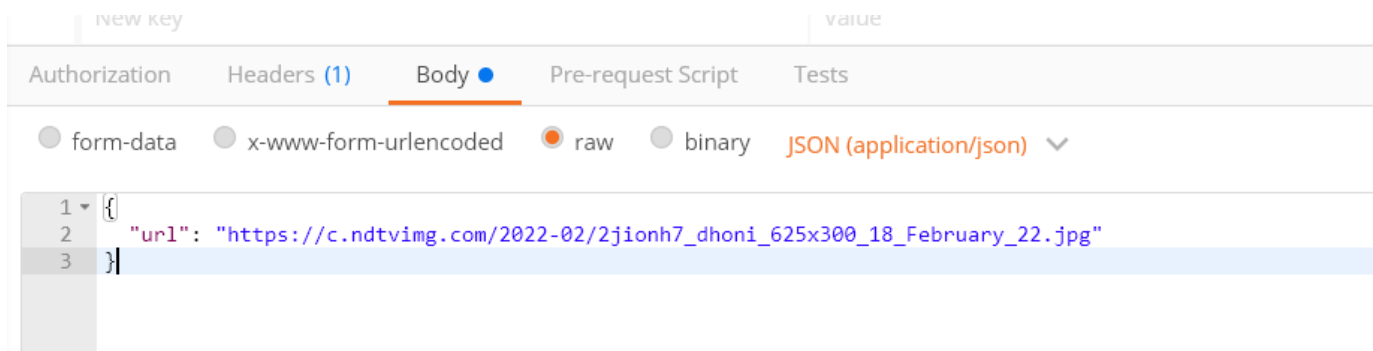
```
1  [
2  {
3    "faceId": "dbc05c49-20d7-4f0f-9c77-ea8c2b77fdb1",
4    "faceRectangle": {
5      "top": 67,
6      "left": 223,
7      "width": 75,
8      "height": 75
9    },
10   "faceLandmarks": {
11     "pupilLeft": {
12       "x": 250.5,
13       "y": 83.8
14     },
15     "pupilRight": {
16       "x": 277.6,
17       "y": 87.4
18     },
19     "noseTip": {
20       "x": 255,
21       "y": 105.9
22     },
23     "mouthLeft": {
24       "x": 248.1,
25       "y": 124.3
26     },
27     "mouthRight": {
28       "x": 272.2,
29       "y": 126.4
30     },
31     "eyebrowLeftOuter": {
32       "x": 242,
33       "y": 73.2
34     }
35   }
36 }
```

The input can also be given as the JSON request. Goto Body and check the Raw radio button and type as JSON(application/json).

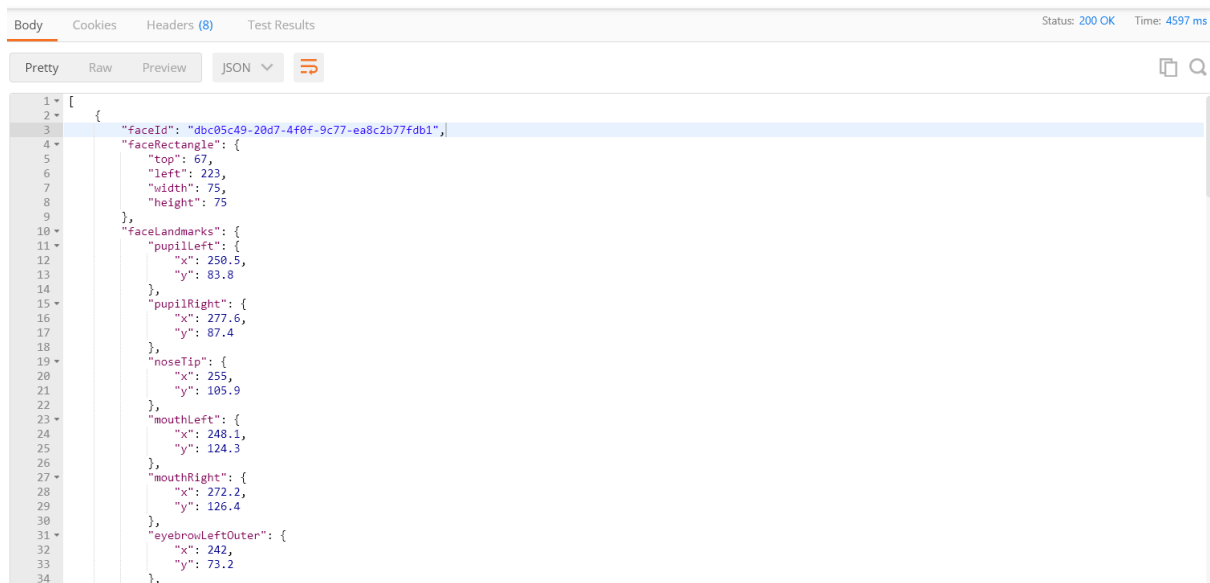
Enter the JSON request body in the below format

```
{
  "url": "https://c.ndtvimg.com/2022-02/2jionh7_dhoni_625x300_18_February_22.jpg"
}
```

Please note that this API will support only one URL in a single request. Multiple URL's with single request is not allowed.



Response:



B.Face Verification Endpoint

- 1.Open Postman and create a new Post request.
- 2.Enter the Request as URL as <http://167.172.151.164:3000/verify>
- 3.Goto Body and check x-www-formurlencoded radio button.
- 4.Enter the key as "facelid1" and give the face Id of the face that you got from the Face detection.

5. Enter the key as “faceId2” and give the face Id of the another face to be compared.

6. Click on the Send to verify if both the faces are identical.

POST ⌵ http://167.172.151.164:3000/verify Params Send ⌵

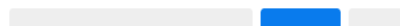
Authorization Headers (1) **Body** ● Pre-request Script Tests

☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary

	Key	Value	Description
<input checked="" type="checkbox"/>	faceId1	191ff01a-40f7-47e0-965b-ff465eff358d	
<input checked="" type="checkbox"/>	faceId2	b38f17f1-17f3-4606-818c-9f7baf54d568	
	New key	Value	Description

Response

Hit the Send button to get a response.



Response:

Body Cookies Headers (8) Test Results Status: 200 OK

Pretty Raw Preview JSON ⌵

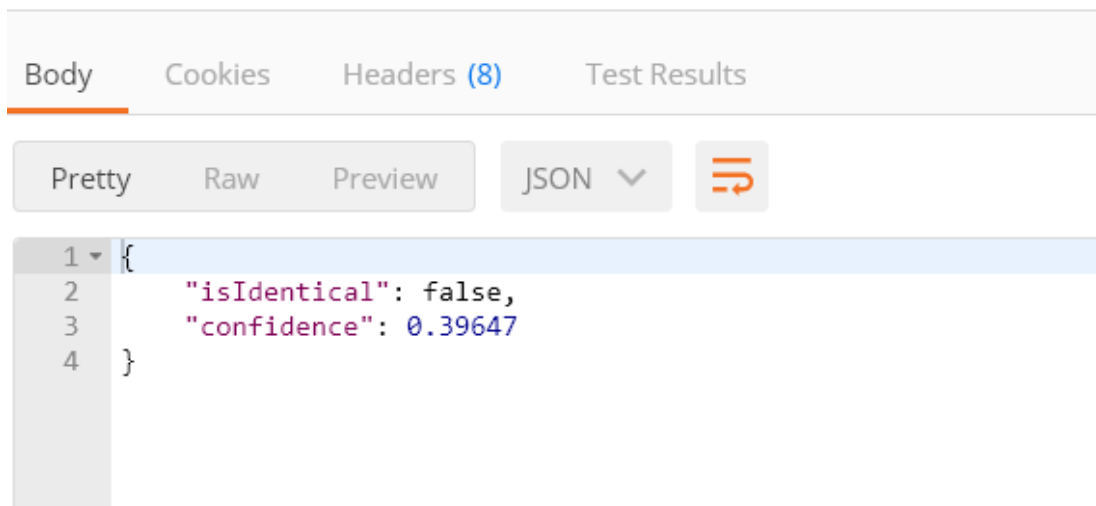
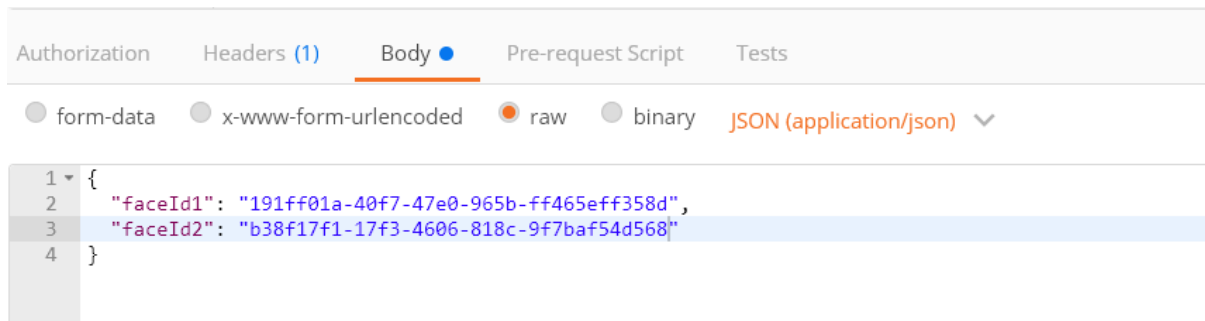
```
1 {  
2   "isIdentical": false,  
3   "confidence": 0.39647  
4 }
```

The input can also be given as the JSON request. Goto Body and check the Raw radio button and type as JSON(application/json).

Enter the JSON request body in the below format

```
{  
  "faceId1": "26d29182-5c26-473c-b8ed-dc5f657bbb8d",  
  "faceId2": "4b12fae3-d712-4fb9-9a69-f69224b6518e"  
}
```

Please note that this API will support to verify only two faces in a single request. Verifying multiple faces is not allowed.



Executing with Swagger UI-

Face Detect:

1. Open Browser and enter the URL as <http://167.172.151.164:3000/docs>
2. Click on /detect – Face Detect API method.
3. Click on Try it out to execute the API. The sample request body will be auto populated.
4. Update the URL for which the faces to be detected and click on Execute.
5. Swagger will only execute the API if the request body is in correct format.

Request:

POST /detect Face Detect API

API to detect faces in an image

ParametersCancel

Name	Description
body <small>* required</small>	
object (body)	body of the face detect API
	Edit Value Model

```
{
  "url": "https://c.ndtvimg.com/2022-02/2jionh7_dhoni_625x300_18_February_22.jpg"
}
```


Cancel

Parameter content type
application/json

Execute

Response:

Server response

Code	Details
200	<div>Response body</div> <pre>{ "faceId": "704e0192-bb88-45d4-9442-b2c516909bb0", "faceRectangle": { "top": 67, "left": 223, "width": 75, "height": 75 }, "faceLandmarks": { "pupilLeft": { "x": 250.5, "y": 83.8 }, "pupilRight": { "x": 277.6, "y": 87.4 }, "noseTip": { "x": 255, "y": 105.9 }, "mouthLeft": { "x": 248.1, "y": 124.3 }, "mouthRight": { "x": 272.2, </pre> <div> Download</div>

Response headers

Responses

Face Verify:

1. Open Browser and enter the URL as <http://167.172.151.164:3000/docs>
2. Click on /verify – Face Detect API method.
3. Click on Try it out to execute the API. The sample request body will be auto populated.
4. Update the faceId1 and faceId2 for which you want to verify and click on Execute.
5. The Face Id's should be taken from the Face detection API
6. Swagger will only execute the API if the request body is in correct format

Request:

POST `/verify` Face Verify API

API to verify whether two faces belong to a same person or whether one face belongs to a person.

Parameters

Name	Description
body * required object (body)	Body of the Face verify API Edit Value Model

```
{
  "faceId1": "191ff01a-40f7-47e0-965b-ff465eff358d",
  "faceId2": "b38f17f1-17f3-4606-818c-9f7baf54d568"
}
```

Response:

Server response

Code	Details
200	<div>Response body<pre>{ "isIdentical": false, "confidence": 0.39647 }</pre>Download</div> <div>Response headers<pre>access-control-allow-origin: * connection: keep-alive content-length: 42 content-type: application/json; charset=utf-8 date: Sun, 01 May 2022 22:59:26 GMT etag: W/"2a-ng/wf77X19Jy0tXbelUomL+60" keep-alive: timeout=5 x-powered-by: Express</pre></div>

Responses

Code	Description
------	-------------

References:

1. <https://westus.dev.cognitive.microsoft.com/docs/services/563879b61984550e40cbb8d/operations/563879b61984550f30395236>
2. <https://docs.microsoft.com/en-us/azure/cognitive-services/face/>