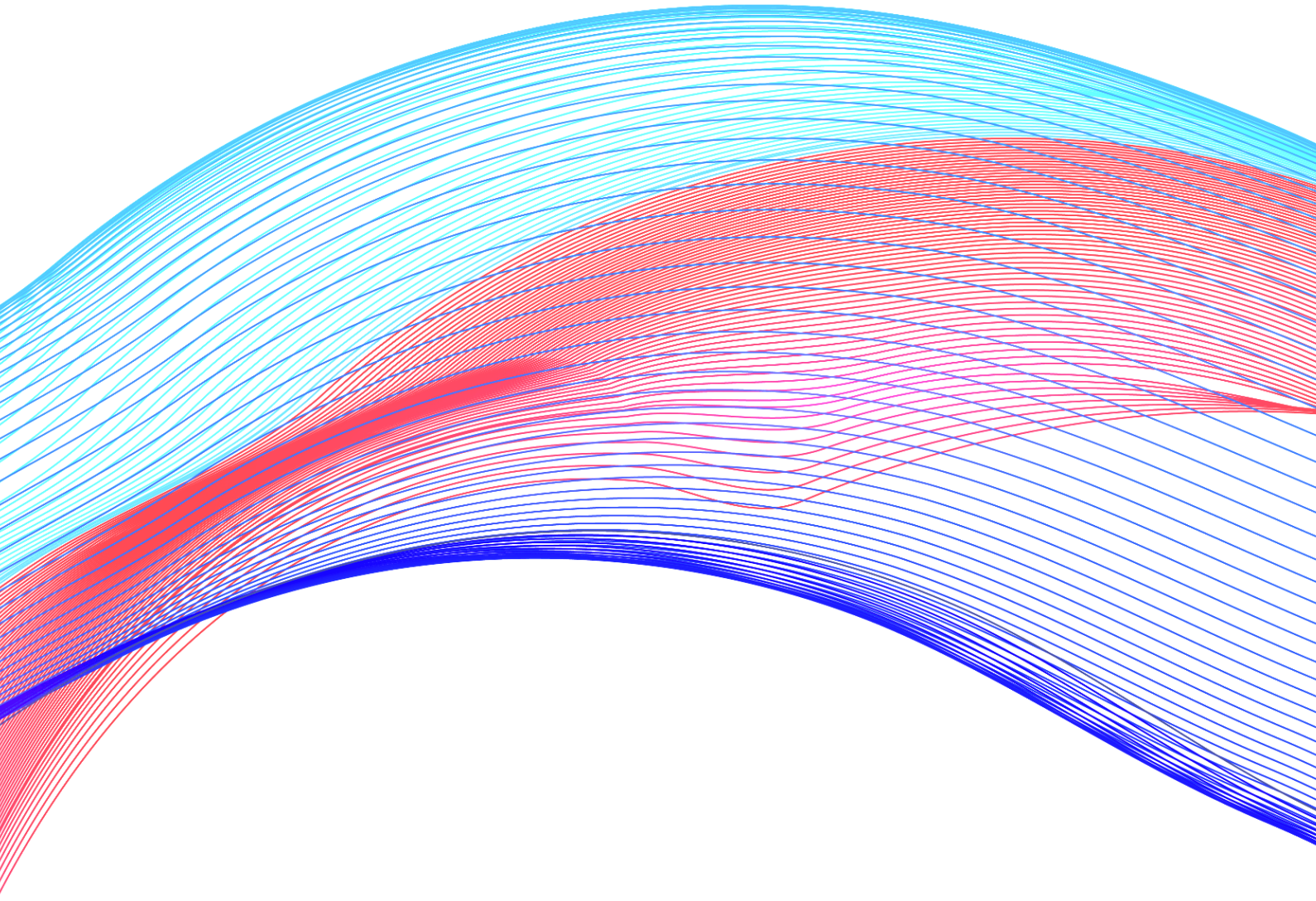
Image API 1.0.1

User Guide



Contents

Glossary

Term	Description
Sample	A set of processing input/output data. Image API processing services receive samples with input data and return the processing result also generated as a sample. In Image API, the sample to be processed should contain an image in base64.
Object attributes (objects: faces and bodies	Gender, age, emotions, anthropometric points, liveness, medical face masks.
Face verification	Comparison of two face images to verify whether they belong to the same person (1:1).
Biometric template	A unique set of biometric features extracted from a face image. Templates are used to compare two face images and determine a degree of their similarity.

Introduction

Image API is a suite of software tools designed to perform face and body recognition, analysis, identification, and verification based on input images and data samples. System services process images in jpg, png or bmp, or samples that contain images in base64. The data output is a sample with a set of attributes.

The system is ready to solve the following functional tasks:

- Face and body detection;
- Determination of anthropometric points and head rotation angles (yaw, pitch, roll);
- Estimation of age, gender and emotions;
- Face mask detection;
- Liveness check;
- Image quality assessment;
- Face verification;
- Extraction of a biometric template.

Architecture

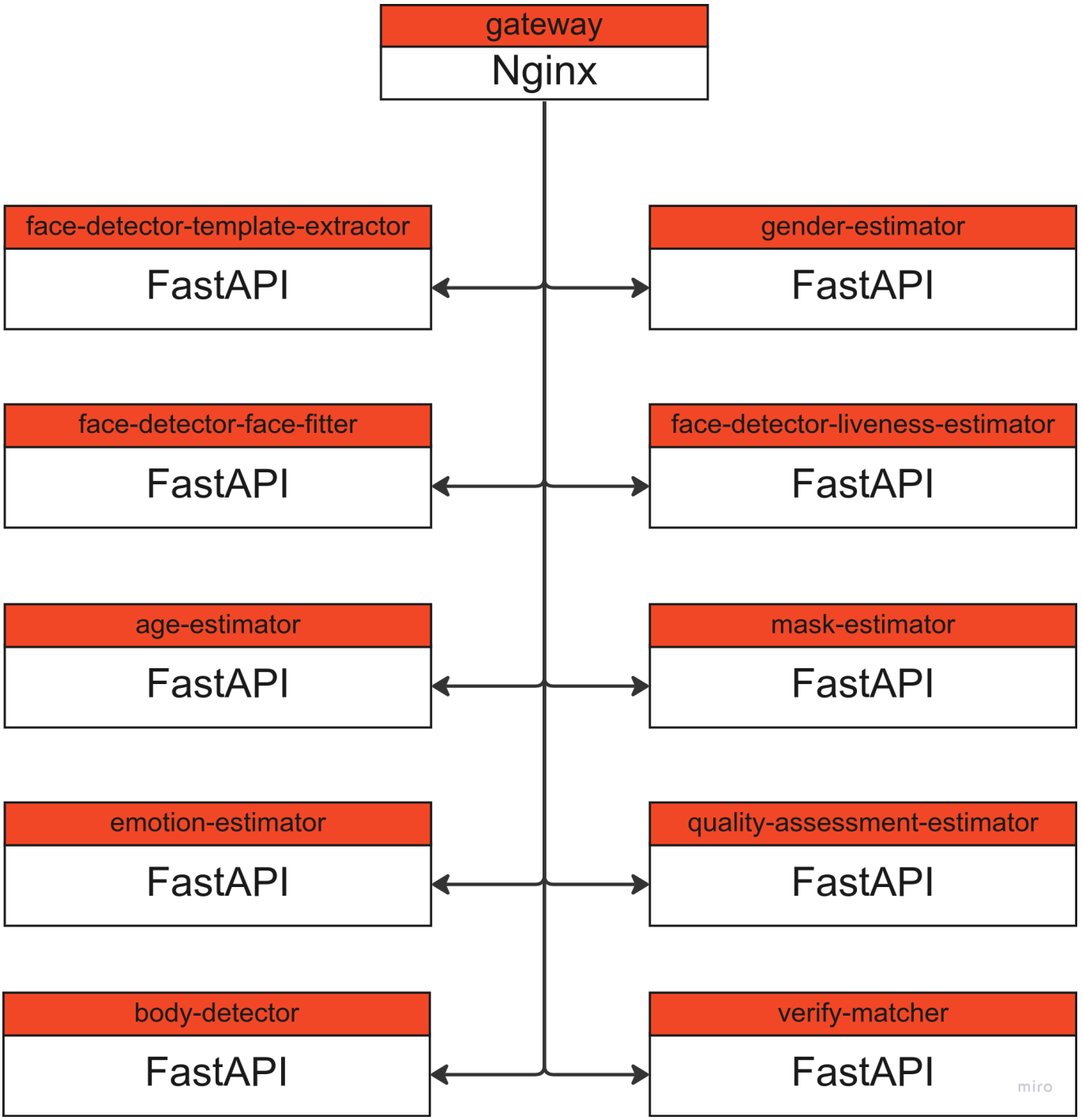


Image API consists of the following services:

- **gateway:** the service is responsible for getting access to Image API and interacting with the processing services listed below;
- **body-detector:** the service is designed to detect bodies in an image. The detection result is the coordinates of the bounding box (bbox) around the detected body;

- **face-detector-face-fitter:** the service is used to detect faces and determine the anthropometric points and the head rotation angles. The returned result is a set of 21 anthropometric points and the values of head rotation angles (yaw, pitch, roll);
- **emotion-estimator:** the service estimates a person's emotions from a face image. The result is the degree of manifestation of each estimated emotion;
- **age-estimator:** the service estimates a person's age from a face image. The estimation result is the numerical value of the person's age;
- **gender-estimator:** the service is used to determine a person's gender from a face image;
- **face-detector-liveness-estimator:** the service is used to detect faces and determine if a person in the image is real or fake. The result is a Real/Fake verdict with a numerical value of confidence;
- **mask-estimator:** the service detects if a person in the image is wearing a medical mask. The result is a True/False verdict with a numerical value of confidence;
- **quality-assessment-estimator**: the service is designed to estimate the quality of a face image. The result is a list of detected faces with a detailed image quality analysis;
- **verify-matcher:** the service compares two face images to determine if they belong to the same person;
- **face-detector-template-extractor:** the service is designed to detect faces and extract the biometric template from a face image.

Interaction with the services is performed via REST API. The following sections contain the detailed description of API requests and responses that provide the main functionality of Image API.

Access to Image API

To review how Image API services work, we provide a special web interface available at ***<image_api_url>***. From this entry point you can get access to the Swagger documentation.

To directly access the Swagger documentation for a specific service, go to ***<image_api_url>/<service_name>/docs***, where ***image_api_url*** is the address of the deployed product, and ***service_name*** is the service name.

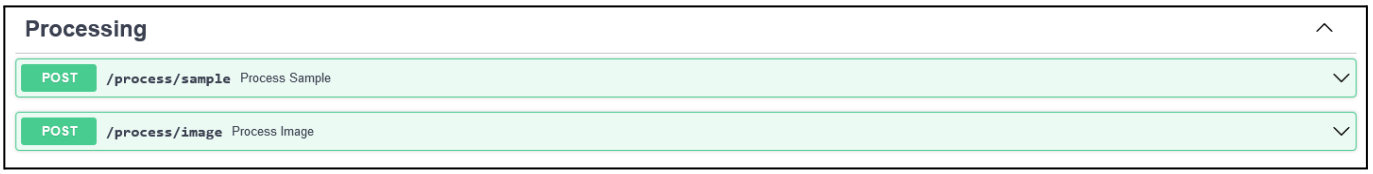
Learn more about Swagger at https://swagger.io/.

List of service names:

- face-detector-face-fitter
- face-detector-liveness-estimator
- face-detector-template-extractor
- age-estimator
- emotion-estimator
- gender-estimator
- mask-estimator
- quality-assessment-estimator
- body-detector
- verify-matcher

Structure of API-request

Swagger web interface contains the following types of requests: **Process Sample** and **Process Image**.



Process Sample

Content-Type: application/json

Path: *<image_api_url>/<service_name>/process/sample*

The request body is a sample that contains:

- ***$image*** - input image in base64
- ***objects*** - processing objects (faces and bodies)

Each object has a certain set of attributes, which differs depending on the service used. In response, API returns these attributes with calculated values.

Note: *fields marked with a "$", for example $image, indicate that the value is encoded in base64 format.*

Process Image

Content-Type: *multipart/form-data*

Path: *<image_api_url>/<service_name>/process/image*

The request body is an image in jpg, png or bmp.

Note: *this interface is not available for quality-assessment-estimator and verify-matcher services, as they require additional object attributes as input data. You can check the availability of a certain service interface in the Swagger web interface.*

Attributes

objects:

- **angles:** head rotation angles:
  - **yaw**: rotation around vertical Y-axis;
  - **pitch**: rotation around horizontal Z-axis;
  - **roll**: rotation around horizontal X-axis.
- **age:** age;
- **bbox:** the rectangle that represents face bounds in the image. The bounds are calculated relative to the coordinates of the original image;
- **class:** object class name, e.g., face or body;
- **confidence:** numerical value of detection confidence;
- **emotions:**
  - **emotion:** 7 basic emotions: angry, disgusted, scared, happy, neutral, sad, surprised;
  - **confidence:** the numerical value of manifestation of each estimated emotion. Value range: 0 to 1.
- **fitter:**
  - **fitter_type:** type of set of anthropometric points. The *fda* provides high accuracy in a wide range of facial angles (up to the full profile), in contrast to the previous sets, however, recognition algorithms still require face samples to be close to frontal. The *fda* set contains 21 points;
  - **keypoints:** anthropometric points;
  - **left_eye:** left eye coordinates;
  - **right_eye:** right eye coordinates;
- **gender:** gender;
- **id:** the ordinal number of a face in the image;
- **liveness:**
  - **confidence:** numerical value of confidence that the image belongs to a real person. Value range: 0 to 1;
  - **value:** verdict: REAL - the face image belongs to a real person, FAKE - the face image does not belong to a real person.
- **mask**:
  - **value:** verdict: true - masked person, false - unmasked person;

- **confidence:** the numerical value of confidence that a person in the image is/isn't wearing a mask. Value range: 0 to 1.

- **quality:**
  - **qaa** (Quality Assessment Algorithm) contains the following data:
    - **totalScore:** numerical value for overall image quality score in points from 0 to 100;
    - **isSharp:** boolean value for image sharpness;
    - **sharpnessScore:** numerical value for sharpness assessment in points from 0 to 100;
    - **isEvenlyIlluminated:** boolean value for image illumination uniformity;
    - **illuminationScore:** numerical value for score of illumination uniformity in points from 0 to 100;
    - **noFlare:** boolean value for presence/absence of image flares;
    - **isLeftEyeOpened:** boolean value for position of left eye (open/closed);
    - **leftEyeOpennessScore:** numerical value for eye openness score in points from 0 to 100;
    - **isRightEyeOpened:** boolean value for position of right eye (open/closed);
    - **rightEyeOpennessScore:** numerical value for eye openness score in points from 0 to 100;
    - **isBackgroundUniform:** boolean value for background uniformity;
    - **backgroundUniformityScore:** numerical value for background uniformity score in points from 0 to 100;
    - **isDynamicRangeAcceptable:** boolean value shows that dynamic range of image intensity in face area exceeds/does not exceed the value of 128;
    - **dynamicRangeScore:** numerical value for score of dynamic range of intensity in points from 0 to 100;
    - **isEyesDistanceAcceptable:** boolean value for acceptable/unacceptable distance between the eyes;
    - **eyesDistance:** numerical value of distance between eyes in px;
    - **isNotNoisy:** boolean value for presence/absence of image noise;
    - **noiseScore:** numerical value of image noise in points from 0 to 100;
    - **isMarginsAcceptable:** boolean value for acceptable/unacceptable margins;
    - **marginInnerDeviation:** numerical value of inner deviation in px;
    - **marginOuterDeviation:** numerical value of outer deviation in px;

    - **isNeutralEmotion:** boolean value for presence/absence of neutral emotions;
    - **neutralEmotionScore:** numerical value for score of neutral emotions in points from 0 to 100;
    - **notMasked:** boolean value for presence/absence of medical mask;
    - **notMaskedScore:** numerical value of confidence that a person in the image isn't wearing a medical mask in points from 0 to 100;
    - **hasWatermark:** boolean value for presence/absence of watermark in the image;
    - **watermarkScore:** numerical value of confidence that the image contains a watermark in points from 0 to 100;
    - **isRotationAcceptable:** boolean value for acceptable/ unacceptable yaw, pitch and roll angles;
    - **maxRotationDeviation:** numerical value for max deviation of yaw, pitch and roll angles.
- **$template:** biometric template coded in base64;
- **template_size:** size of biometric template (vector length);

verification:

- **distance:** distance between vectors of biometric templates;
- **fa_r:** false acceptance rate;
- **fr_r:** false rejection rate;
- **score:** face verification rate from 0 (0%) to 1 (100%).

Application

Face Detection

The following Image API services perform face detection:

- **face-detector-face-fitter**
- **face-detector-template-extractor**
- **face-detector-liveness-estimator**

Body Detection

The request is sent to **body-detector** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **confidence**
- **bbox**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:

```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "body",
      "confidence": 0.8266383409500122,
      "bbox": [
        0.648772656917572,
        0.13773296773433685,
        0.9848934412002563,
        0.8240703344345093
      ]
    },
    {
```

```
      "id": 1,
      "class": "body",
      "confidence": 0.7087612748146057,
      "bbox": [
        0.35164034366607666,
        0.15803256630897522,
        0.6833359003067017,
        0.8854727745056152
      ]
    }
  ]
}
```


Gender Estimation

The request is sent to **gender-estimator** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **gender**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:


```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "gender": "MALE"
    }
  ]
}
```


Age Estimation

The request is sent to **age-estimator** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **age**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:

```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "age": "25"
    }
  ]
}
```

Emotion Estimation

The request is sent to **emotion-estimator** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **emotions**
  - **emotion**
  - **confidence**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:


```
{
  "$image":"image in base64",
 "objects": [
    {
      "id": 0,
      "class": "face",
      "emotions": [
        {
          "confidence": 0.10151619818478974,
          "emotion": "ANGRY"
        },
        {
          "confidence": 0.07763473911731263,
          "emotion": "DISGUSTED"
        },
        {
          "confidence": 0.20321173801223097,
          "emotion": "SCARED"
        },
        {
          "confidence": 0.08768639197580883,
          "emotion": "HAPPY"
        },
        {
          "confidence": 0.19000983487515088,
          "emotion": "NEUTRAL"
        },
        {
          "confidence": 0.08262699313446588,
          "emotion": "SAD"
        },
        {
          "confidence": 0.257314104700241,
          "emotion": "SURPRISED"
        }
      ]
    }
  ]
}
```


Liveness Estimation

The request is sent to **face-detector-liveness-estimator** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **bbox**
- **confidence**
- **liveness:**
  - **confidence**

- **value**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:

```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "confidence": 0.8233476281166077,
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ],
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      }
    }
  ]
}
```

Face mask check

The request is sent to **mask-estimator** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **mask**:
  - **value**
  - **confidence**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
```

```
  ]
}
```

Response example:

```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "mask": {
        "confidence": 0.07230597734451294,
        "value": false
      }
    }
  ]
}
```

Anthropometric Points

This request is sent to **face-detector-face-fitter** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **bbox**
- **confidence**
- **fitter:**
  - **fitter_type**
  - **keypoints**
  - **left_eye**
  - **right_eye**
- **angles:**
  - **yaw**
  - **pitch**
  - **roll**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
```

```
}
```

Response example:


```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "confidence": 0.895225465297699,
      "bbox": [
        0.10445103857566766,
        0.05966162065894924,
        0.7008902077151336,
        0.9243098842386465
      ],
      "fitter": {
        "fitter_type": "fda",
        "keypoints": [
          344.24078369140625,
          379.23858642578125,
          0,
          443.0493469238281,
          364.8091125488281,
          0,
          547.5462646484375,
          384.35833740234375,
          0,
          724.5175170898438,
          385.01220703125,
          0,
          816.4994506835938,
          366.4952697753906,
          0,
          899.7161865234375,
          380.7967224121094,
          0,
          391.20654296875,
          461.12066650390625,
          0,
          461.524169921875,
          459.44287109375,
          0,
          531.2512817382812,
          467.7398681640625,
          0,
          721.8792724609375,
          468.227294921875,
          0,
          784.9144897460938,
          461.4508056640625,
          0,
          854.609130859375,
          465.002685546875,
          0,
          250.21035766601562,
          657.1244506835938,
          0,
          559.1598510742188,
```

```
          666.7738647460938,
          0,
          641.8836059570312,
          678.353515625,
          0,
          710.0083618164062,
          670.3438110351562,
          0,
          939.4479370117188,
          656.3207397460938,
          0,
          509.8494873046875,
          816.0798950195312,
          0,
          634.861083984375,
          823.5408325195312,
          0,
          748.5276489257812,
          813.4531860351562,
          0,
          633.5501098632812,
          1033.822509765625,
          0
        ],
        "left_eye": [
          461.524169921875,
          459.44287109375
        ],
        "right_eye": [
          784.9144897460938,
          461.4508056640625
        ]
      },
      "angles": {
        "yaw": 6.648662090301514,
        "roll": 0.3107689917087555,
        "pitch": -23.410654067993164
      }
    }
  ]
}
```


Quality Assessment

This request is sent to **quality-assessment-estimator** service. In the request body pass the values of face attributes obtained after processing the image by **face-detector-face-fitter**. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **confidence**
- **bbox**

- **quality:**
  - **qaa:**
    - **totalScore**
    - **isSharp**
    - **sharpnessScore**
    - **isEvenlyIlluminated**
    - **illuminationScore**
    - **noFlare**
    - **isLeftEyeOpened**
    - **leftEyeOpennessScore**
    - **isRightEyeOpened**
    - **rightEyeOpennessScore**
    - **isBackgroundUniform**
    - **backgroundUniformityScore**
    - **isDynamicRangeAcceptable**
    - **dynamicRangeScore**
    - **isEyesDistanceAcceptable**
    - **eyesDistance**
    - **isNotNoisy**
    - **noiseScore**
    - **isMarginsAcceptable**
    - **marginInnerDeviation**
    - **marginOuterDeviation**
    - **isNeutralEmotion**
    - **neutralEmotionScore**
    - **notMasked**
    - **notMaskedScore**
    - **hasWatermark**
    - **watermarkScore**
    - **isRotationAcceptable**
    - **maxRotationDeviation**

Request example:

```
{
```

```
"$image": "string",
"objects": [
    {
    "id": 0,
    "class": "face",
    "confidence": 0.895225465297699,
    "bbox": [
      0.10445103857566766,
      0.05966162065894924,
      0.7008902077151336,
      0.9243098842386465
    ],
    "fitter": {
      "fitter_type": "fda",
      "keypoints": [
        344.24078369140625,
        379.23858642578125,
        0,
        443.0493469238281,
        364.8091125488281,
        0,
        547.5462646484375,
        384.35833740234375,
        0,
        724.5175170898438,
        385.01220703125,
        0,
        816.4994506835938,
        366.4952697753906,
        0,
        899.7161865234375,
        380.7967224121094,
        0,
        391.20654296875,
        461.12066650390625,
        0,
        461.524169921875,
        459.44287109375,
        0,
        531.2512817382812,
        467.7398681640625,
        0,
        721.8792724609375,
        468.227294921875,
        0,
        784.9144897460938,
        461.4508056640625,
        0,
        854.609130859375,
        465.002685546875,
        0,
        250.21035766601562,
        657.1244506835938,
        0,
        559.1598510742188,
        666.7738647460938,
        0,
        641.8836059570312,
        678.353515625,
        0,
        710.0083618164062,
```

```
          670.3438110351562,
          0,
          939.4479370117188,
          656.3207397460938,
          0,
          509.8494873046875,
          816.0798950195312,
          0,
          634.861083984375,
          823.5408325195312,
          0,
          748.5276489257812,
          813.4531860351562,
          0,
          633.5501098632812,
          1033.822509765625,
          0
        ],
        "left_eye": [
          461.524169921875,
          459.44287109375
        ],
        "right_eye": [
          784.9144897460938,
          461.4508056640625
        ]
      },
      "angles": {
        "yaw": 6.648662090301514,
        "roll": 0.3107689917087555,
        "pitch": -23.410654067993164
      }
    }
  ]
}
```


Response example:


```
{
  "$image": "string",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "confidence": 0.69044026635,
      "bbox": [
        0.42242398858070374,
        0.05838850140571594,
        0.5360375642776489,
        0.17216356098651886
      ],
      "quality": {
        "qaa": {
          "totalScore": 0,
          "isSharp": true,
          "sharpnessScore": 0,
          "isEvenlyIlluminated": true,
          "illuminationScore": 0,
          "noFlare": true,
          "isLeftEyeOpened": true,
```

```
          "leftEyeOpennessScore": 0,
          "isRightEyeOpened": true,
          "rightEyeOpennessScore": 0,
          "isRotationAcceptable": true,
          "maxRotationDeviation": 0,
          "notMasked": true,
          "notMaskedScore": 0,
          "isNeutralEmotion": true,
          "neutralEmotionScore": 0,
          "isEyesDistanceAcceptable": true,
          "eyesDistance": 0,
          "isMarginsAcceptable": true,
          "marginOuterDeviation": 0,
          "marginInnerDeviation": 0,
          "isNotNoisy": true,
          "noiseScore": 0,
          "watermarkScore": 0,
          "hasWatermark": true,
          "dynamicRangeScore": 0,
          "isDynamicRangeAcceptable": true,
          "backgroundUniformityScore": 0,
          "isBackgroundUniform": true
        }
      }
    }
  ]
}
```


Extraction of Biometric Template

This request is sent to **face-detector-template-extractor** service. API returns the following attributes with calculated values:

objects:

- **id**
- **class**
- **confidence**
- **bbox**
- **$template**
- **template_size**

Request example:

```
{
  "$image": "image in base64",
  "objects": [
      {}
  ]
}
```

Response example:

```
{
  "$image":"image in base64",
 "objects": [
    {
      "id": 0,
      "class": "face",
      "confidence": 0.895225465297699,
      "bbox": [
        0.10445103857566766,
        0.05966162065894924,
        0.7008902077151336,
        0.9243098842386465
      ],
      "$template": "template in base64",
      "template_size": 74
    }
  ]
}
```


Face Verification

This request is sent to **verify-matcher** service. In the request body pass the values of face attributes obtained after processing the image by **face-detector-template-extractor**. API returns the following attributes with calculated values:

verification:

- **distance**
- **fa_r**
- **fr_r**
- **score**

Request example:


```
{
  "objects": [
      {"id": 0,

      "class": "face",

      "confidence": 0.9171707630157471,

      "bbox": [

      0.14427860696517414,

      0.21912350597609562,

      0.8656716417910447,

      0.796812749003984

      ],
```

```
      "$template": "template in base64",

      "template_size": 74},

      {"id": 0,

      "class": "face",

      "confidence": 0.8453116416931152,

      "bbox": [

      0.16477272727272727,

      0.22272727272727272,

      0.875,

      0.7954545454545454

      ],

      "$template": "template in base64",

      "template_size": 74}
  ]
}
```


Response example:


```
{
 "objects": [
   {
     "bbox": [
       0.14427860696517414,
       0.21912350597609562,
       0.8656716417910447,
       0.796812749003984
     ],
     "$template": "template in base64",
     "id": 0,
     "class": "face",
     "confidence": 0.9171707630157471,
     "template_size": 74
   },
   {
     "bbox": [
       0.16477272727272727,
       0.22272727272727272,
       0.875,
       0.7954545454545454
     ],
     "$template": "template in base64",
     "id": 0,
     "class": "face",
     "confidence": 0.8453116416931152,
     "template_size": 74
   }
 ],
 "verification": {
```

```
    "distance": 4796,
    "fa_r": 0,
    "fr_r": 0.522820770740509,
    "score": 0.9515298008918762
  }
}
```

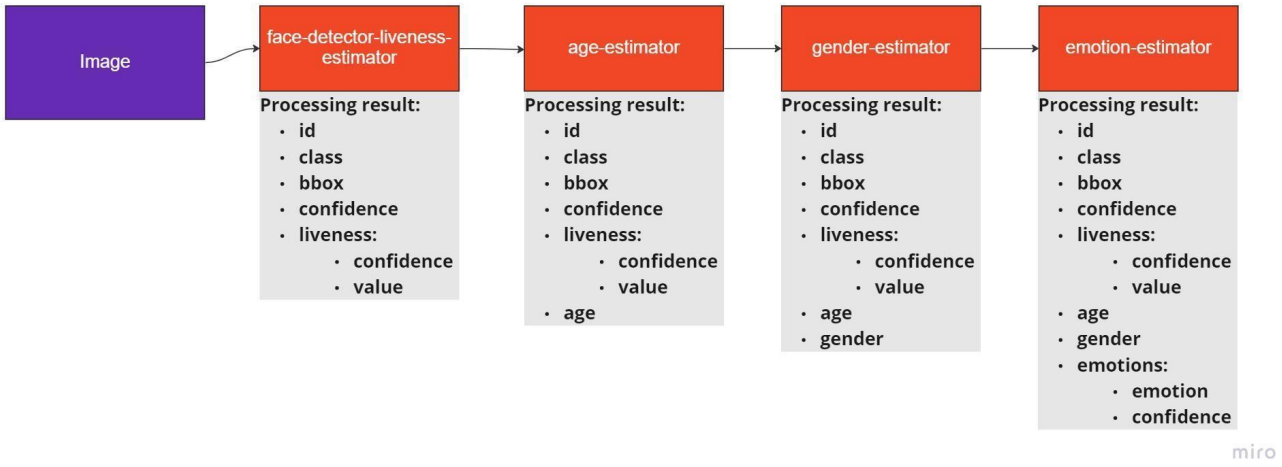

Note: *the input sample for the verify-matcher service may not contain an image, because this service does not process images, but biometric templates extracted from them.*

Service Pipelines

Image API can submit the processing result from one service as input to another service. Thus, you can get enriched data with the composition of results and build pipelines of services depending on your tasks.

The first service in the pipeline should be a service with face detection function. It processes the image and submits the detection result to the next service in the pipeline as input. The face detection function is performed in **face-detector-face-fitter, face-detector-template-extractor, face-detector-liveness-estimator**.

For example, to estimate liveness, age, gender and emotions from a face image, you can build a pipeline of the following services:



An image is sent to **face-detector-liveness-estimator** service. The processing result (face detection and liveness check result) is passed as input data to **age-estimator** service, where the age value is added to the result. Then the obtained enriched data is sent to the **gender-estimator** service to determine a person's gender, and from there to the **emotion-estimator**, where an emotion rating is added.

Examples:

Request to face-detector-liveness-estimator:

```
{
  "$image": "image in base64",
  "objects": [
```

```
      {}
  ]
}
```

Response:

```
{
  "$image":"image in base64",
  "objects": [
    {
      "id": 0,
      "class": "face",
      "confidence": 0.8233476281166077,
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ],
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      }
    }
  ]
}
```

Request to age-estimator:

```
{
  "$image": "image in base64",
  "objects": [
      {"id": 0,
      "class": "face",
      "confidence": 0.8233476281166077,
      "bbox": [
      0.375,
      0.12333333333333334,
      0.7645833333333333,
      0.42
      ],
      "liveness": {
      "confidence": 0.9989556074142456,
      "value": "REAL"
      }
    }
  ]
}
```

Response:

```
{
  "$image":"image in base64",
  "objects": [
    {
     "id": 0,
      "class": "face",
      "age": 21,
      "liveness": {
```

```
        "confidence": 0.9989556074142456,
        "value": "REAL"
      },
      "confidence": 0.8233476281166077,
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ]
    }
  ]
}
```


Request to gender-estimator:


```
{
  "$image": "image in base64",
  "objects": [
      {"id": 0,
      "class": "face",
      "age": 21,
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      },
      "confidence": 0.8233476281166077,
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ]
    }
  ]
}
```


Response:


```
{
  "$image":"image in base64",
  "objects": [
    {
     "id": 0,
      "class": "face",
      "gender": "FEMALE",
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      },
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ],
      "age": 21,
      "confidence": 0.8233476281166077
```

```
    }
  ]
}
```

Request to emotion-estimator:

```
{
  "$image": "image in base64",
  "objects": [
      {"id": 0,
      "class": "face",
      "gender": "FEMALE",
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      },
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ],
      "age": 21,
      "confidence": 0.8233476281166077
    }
  ]
}
```

Response:

```
{
  "$image":"image in base64",
  "objects": [
    {
     "id": 0,
      "class": "face",
      "emotions": [
        {
          "confidence": 0.00046337815752155756,
          "emotion": "ANGRY"
        },
        {
          "confidence": 0.000010690175584454014,
          "emotion": "DISGUSTED"
        },
        {
          "confidence": 0.0000651997511831193,
          "emotion": "SCARED"
        },
        {
          "confidence": 0.050197473080296776,
          "emotion": "HAPPY"
        },
        {
          "confidence": 0.6730428226960512,
          "emotion": "NEUTRAL"
        },
        {
          "confidence": 0.274063680489088,
```

```
          "emotion": "SAD"
        },
        {
          "confidence": 0.0021567556502748936,
          "emotion": "SURPRISED"
        }
      ],
      "bbox": [
        0.375,
        0.12333333333333334,
        0.7645833333333333,
        0.42
      ],
      "confidence": 0.8233476281166077,
      "age": 21,
      "liveness": {
        "confidence": 0.9989556074142456,
        "value": "REAL"
      },
      "gender": "FEMALE"
    }
  ]
}
```