**Como cadastrar uma pessoa no BioPass ID ABIS?**

 Para realizar a operação de cadastro de um indivíduo, é necessário utilizar o método **Enroll**. Esta operação é uma das principais funcionalidades do BioPass ID ABIS e tem como objetivo registrar as informações biométricas de uma pessoa no sistema, permitindo seu reconhecimento em futuras consultas.

Para utilizar essa operação o cliente deve realizar a chamada através da API. A documentação completa está disponível no site<https://docs.biopassid.com/>.

**Campos de entrada:**

* **CustomId:** Identificador único do indivíduo na base.
* **EnrollWithDeduplication:** Possui a finalidade de indicar a necessidade de realizar a deduplicação do registro.
* **BiographicData:** Informações biográficas do usuário, como nome, CPF, data de nascimento e assinatura.
* **Face:** Objeto contendo as informações sobre a biometria facial do cadastro.
* **Fingers:** Objeto contendo as informações da biometria digital de cada dedo que será cadastrado.

Exemplo de como ficaria o corpo da requisição:

{

 "Candidate": {

 "GalleryNames": [

 "string"

 ],

 "CustomId": "string",

 "EnrollWithDeduplication": true,

 "BiographicData": {

 "Nome": "string",

 "Cpf": "string",

 "Renach": [

 {

 "Id": "string"

 }

 ],

 "DataDeNascimento": "2024-07-01",

 "NomeDaMae": "string",

 "NomeDoPai": "string",

 "Gender": "string",

 "Signature": {

 "ImageFileName": "string",

 "ImageBase64": "string"

 },

 },

 "Face": {

 "Face": [

 {

 "ImageFileName": "string",

 "ImageBase64": "string",

 "HorzResolution": 0,

 "VertResolution": 0

 }

 ]

 },

 "Fingers": {

 "Finger": [

 {

 "FingerName": "string",

 "Position": "string",

 "CollectedType": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "HorzResolution": 0,

 "VertResolution": 0,

 "AnomalyDescription": {

 "AnomalyType": "string",

 "AnomalyTimeFrame": "string"

 }

 }

 ]

 }

 },

 "PriorityOrder": 0,

 "DelayOrder": 0

}

O header da requisição, por sua vez, é onde deve ser inserido o tipo do conteúdo. Abaixo, é apresentado um exemplo de como o header deve ser representado:

{

 'Content-Type': 'application/json'

}

Por fim, na requisição, devem ser passados como parâmetros a especificação de que é um método POST, a URL ‘https://abis.api.biopassid.com/enroll/create’, além do header e do corpo descrito anteriormente. Abaixo, apresentamos um exemplo de código em Python que realiza essa requisição. Exemplos em outras linguagens de programação podem ser encontrados em nossa documentação.

import http.client

import json

conn = http.client.HTTPSConnection("abis.api.biopassid.com")

payload = json.dumps({

 "Candidate": {

 "GalleryNames": [

 "string"

 ],

 "CustomId": "string",

 "EnrollWithDeduplication": True,

 "BiographicData": {

 "Nome": "string",

 "Cpf": "string",

 "Renach": [

 {

 "Id": "string"

 }

 ],

 "DataDeNascimento": "2024-07-01",

 "NomeDaMae": "string",

 "NomeDoPai": "string",

 "Gender": "string",

 "Signature": {

 "ImageFileName": "string",

 "ImageBase64": "string"

 },

 },

 "Face": {

 "Face": [

 {

 "ImageFileName": "string",

 "ImageBase64": "string",

 "HorzResolution": 0,

 "VertResolution": 0

 }

 ]

 },

 "Fingers": {

 "Finger": [

 {

 "FingerName": "string",

 "Position": "string",

 "CollectedType": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "HorzResolution": 0,

 "VertResolution": 0,

 "AnomalyDescription": {

 "AnomalyType": "string",

 "AnomalyTimeFrame": "string"

 }

 }

 ]

 }

 },

 "PriorityOrder": 0,

 "DelayOrder": 0

})

headers = {

 'Content-Type': 'application/json'

}

conn.request("POST", "/enroll/create", payload, headers)

res = conn.getresponse()

data = res.read()

print(data.decode("utf-8"))

 A resposta recebida em caso de uma operação bem-sucedida incluirá diversos campos de resposta, como descritos abaixo.

**Campos de resposta:**

* **Success:** Valor booleano informando se houve sucesso na operação.
* **Message:** Mensagem adicional referente ao sucesso da operação.
* **TransactionId:** ID interno da operação realizada.
* **Candidate:** Informações sobre o candidato recém-cadastrado. São elas:
	+ **Person:** Objeto criado na base com as informações do candidato, incluindo seu identificador, biometria facial e digital (caso tenham sido passadas), entre outros.
	+ **BiographicData:** Dados biográficos do indivíduo, passados na chamada da operação.
	+ **Faces:** Informações sobre a biometria facial do indivíduo.
	+ **Fingers:** Informações sobre as biometrias digitais do indivíduo.
	+ **CreateDateUtc:** Data da criação do registro do indivíduo na base.
	+ **MatchingCandidates:** Usuários registrados na base que possuem um match biométrico com o candidato recém-cadastrado.
	+ **CustomId:** Identificador único do usuário na base.
	+ **CandidateStatus:** Status do candidato.

Caso seja retornado um erro, verifique se o texto em formato JSON está corretamente escrito. Caso receba uma mensagem com o código 200 (OK), significa que sua requisição foi solicitada e realizada com sucesso e você pode prosseguir com as próximas requisições. A mensagem de resposta para o exemplo fornecido pode ser verificada abaixo:

{

 "Success": true,

 "Message": "string",

 "TransactionId": "example-candidate-id",

 "Candidate": {

 "Id": "string",

 "Galleries": [

 {

 "Id": "example-candidate-id",

 "Name": "string",

 "Description": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z"

 }

 ],

 "Person": {

 "Id": "string",

 "BiographicData": {

 "Id": "string",

 "Name": "string",

 "Signature": {

 "Id": "string",

 "ImageFileName": "string",

 "PathImageBlob": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z"

 },

 "Cpf": "string",

 "Renachs": [

 {

 "Id": "example-candidate-id",

 }

 ],

 "BirthDate": "2024-07-01T18:09:17.293Z",

 "MotherName": "string",

 "FatherName": "string",

 "Gender": "string",

 "AbisImportedDateUtc": "2024-07-01T18:09:17.293Z"

 },

 "Faces": {

 "Faces": [

 {

 "Id": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "PathImageBlob": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string"

 }

 ],

 "FacesMatch": [

 {

 "InicialImageGuid": "example-candidate-id",

 "ReceivedImageFileName": "string",

 "Score": 0,

 "Threshold": 0

 }

 ]

 },

 "Fingers": {

 "Fingers": [

 {

 "Id": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "PathImageBlob": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string",

 "Nfiq1": 0,

 "Nfiq2": 0,

 "Nfiq21": 0,

 "Position": 0,

 "PositionName": "string",

 "CollectedType": "string",

 "AnomalyTimeFrame": "string",

 "AnomalyType": 0,

 "AnomalyTypeName": "string"

 }

 ],

 "FingersMatch": [

 {

 "InicialImageGuid": "example-candidate-id",

 "ReceivedImageFileName": "string",

 "Score": 0,

 "Threshold": 0,

 "StatusName": "string",

 "ErrorMessage": "string"

 }

 ]

 },

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "MatchingPersons": [

 {

 "CandidateId": "example-candidate-id",

 "Score": 0,

 "Threshold": 0,

 "FacesMatched": [

 {

 "MatchedIndex": 0,

 "Score": 0,

 "SondaTemplateFaceId": "example-candidate-id",

 "ReferenceTemplateFaceId": "example-candidate-id"

 }

 ],

 "FingerMatched": [

 {

 "FingerMatchedId": 0,

 "Score": 0,

 "FingerSourceMatchedId": 0,

 "FingerSourcePosition": 0,

 "SondaTemplateFingerId": "example-candidate-id",

 "ReferenceTemplateFingerId": "example-candidate-id"

 }

 ]

 }

 ]

 },

 "BiographicData": {

 "Id": "string",

 "Name": "string",

 "Signature": {

 "Id": "string",

 "ImageFileName": "string",

 "PathImageBlob": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z"

 },

 "Cpf": "string",

 "Renachs": [

 {

 "Id": "example-candidate-id",

 }

 ],

 "BirthDate": "2024-07-01T18:09:17.293Z",

 "MotherName": "string",

 "FatherName": "string",

 "Gender": "string",

 "AbisImportedDateUtc": "2024-07-01T18:09:17.293Z"

 },

 "Faces": {

 "Faces": [

 {

 "Id": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "PathImageBlob": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string"

 }

 ],

 "FacesMatch": [

 {

 "InicialImageGuid": "example-candidate-id",

 "ReceivedImageFileName": "string",

 "Score": 0,

 "Threshold": 0

 }

 ]

 },

 "Fingers": {

 "Fingers": [

 {

 "Id": "string",

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "PathImageBlob": "string",

 "ImageFileName": "string",

 "ImageBase64": "string",

 "ImagePngBase64": "string",

 "ImageThumbnail120PngBase64": "string",

 "Nfiq1": 0,

 "Nfiq2": 0,

 "Nfiq21": 0,

 "Position": 0,

 "PositionName": "string",

 "CollectedType": "string",

 "AnomalyTimeFrame": "string",

 "AnomalyType": 0,

 "AnomalyTypeName": "string"

 }

 ],

 "FingersMatch": [

 {

 "InicialImageGuid": "example-candidate-id",

 "ReceivedImageFileName": "string",

 "Score": 0,

 "Threshold": 0,

 "StatusName": "string",

 "ErrorMessage": "string"

 }

 ]

 },

 "CreateDateUtc": "2024-07-01T18:09:17.293Z",

 "MatchingCandidates": [

 {

 "CandidateId": "example-candidate-id",

 "Score": 0,

 "Threshold": 0,

 "FacesMatched": [

 {

 "MatchedIndex": 0,

 "Score": 0,

 "SondaTemplateFaceId": "example-candidate-id",

 "ReferenceTemplateFaceId": "example-candidate-id"

 }

 ],

 "FingerMatched": [

 {

 "FingerMatchedId": 0,

 "Score": 0,

 "FingerSourceMatchedId": 0,

 "FingerSourcePosition": 0,

 "SondaTemplateFingerId": "example-candidate-id",

 "ReferenceTemplateFingerId": "example-candidate-id"

 }

 ]

 }

 ],

 "CustomId": "string",

 "CandidateStatus": "string",

 "HitStatus": "string"

 }

}