CAREERPRO - FEDERAL LEARNING ACCOUNT

REST API manual for training providers

03/09/2025

A service from

asbl Sigedis vzw Place Victor Horta / Victor Hortaplein 11 (P3) Bruxelles 1060 Brussel T +32 (0)2 791 50 00 M info@sigedis.fgov.be www.sigedis.be

Inhoudstabel

1. Introduction

For the FLA platform various channels were developed that are to make it as easy as possible for the training provider to communicate training data to Sigedis.

Training providers who work with IT systems to manage their trainings can use a web service (online channel, REST API) or file transfer (sFTP, BATCH channel) to transmit data. The transfer of data via BATCH or API is an automated process between two IT systems. This means that there is no need for a manual user intervention to transfer the data.

The documentation below describes how the API works: the provided /paths and methods.

It is intended for IT professionals who want to know more about the technical aspects of the FLA API for training providers. Specialized terminology and technical examples are used to clearly explain key concepts.

This document is part of the documents available to training providers:

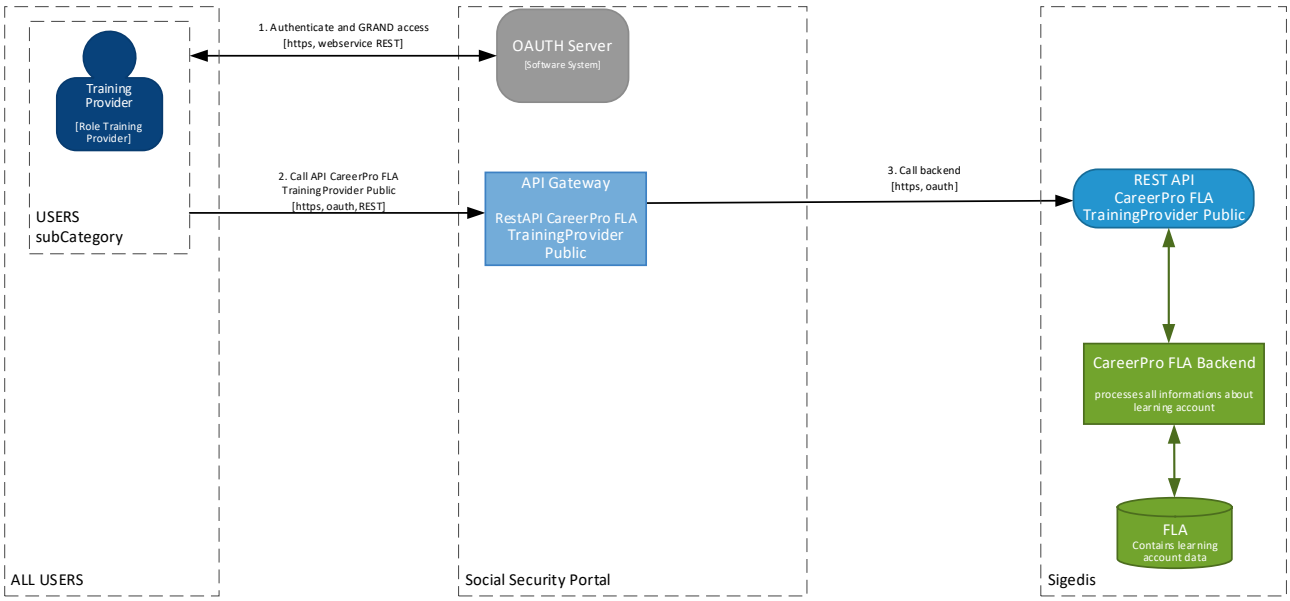
Document	Description
BATCH Channel Manual	Document describing the necessary steps to transmit FLA data by using the BATCH channel.
API Channel Manual	Document describing the necessary steps to transmit FLA data by using the API channel.
Description of the anomalies	List of all anomalies and warnings related to the declaration of FLA data.
Glossary	Technical documentation describing the data blocks and zones of the batch and API.
XSD	Technical scheme defining the BATCH structure.
SWAGGER	Technical scheme defining the API structure.
Setting up the batch channel	Document describing the necessary steps to configure the BATCH channel on the social security portal.
Setting up the API channel	Document describing the necessary steps to configure the web service channel (API) on the social security portal.

2. Communication CareerProFLA TrainingProvider API

By means of the 'CareerProFLA TrainingProvider' API the information systems of the training provider can communicate with Sigedis. More in particular through the secured channels of the Social Security portal.

If this communication has not been configured yet, you first have to go through the 'setting up API communication' manual.

The scheme below shows how the exchange of data between the training provider and Sigedis shall be realized (high level).



1. The training provider using the configured certificate, requests an authentication token with the oAuth server of the Social Security portal.
2. The authentication token, together with the 'CareerProFLA API TrainingProvider'-call , is transmitted to the Social Security API Gateway.
3. The API call is transmitted to Sigedis, who will process it in the CareerProFLA Backend. This processing will be done online and immediately.
   Once the processing is completed, the response to the API call is returned to the user (through the Social Security API Gateway)

3. CareerPro FLA TrainingProvider API

Use of the 'CareerPro FLA TrainingProvider'-API allows a synchronous communication with Sigedis. Within this API it is provided to register, correct, delete and consult trainings. This consultation can be of a specific training, the training history of a specific citizen with the requesting training provider, as well as the training history of a citizen for a specific employer (also only for the trainings of the requesting training provider).

3.1. Concepts

3.1.1. Unique identification number of the training (UUID)

The transfer of a training must always be accompanied by a unique reference number "UUID". This number must be passed on in the <TrainingId> zone and allows a training to be uniquely identified. The UUID must be calculated by the training provider who must respect the standards of this reference type.

A UUID:

- is composed of 32 hexadecimal digits, separated by 4 hyphens.
  Example: ffa072c4-6ece-43de-beef-1d1927252d58
- is created by one of the 5 main algorithms for generating a UUID. There are *libraries* that can help you get a UUID.

In case of a change/cancellation of a training that has already been sent, the previously sent UUID must be resumed.

3.1.2. The concept of a 'complete picture' of a training

Each declaration (or correction) of a training must be complete (= a complete 'picture' of the training, with all participants and detailed periods).

When a training consists of multiple periods (sessions, modules, etc.), all training periods must be included in the declaration. **Trainings that take place over multiple FLA windows (= 5-year period in which training rights are built up) must be split: at least one training period per FLA window.**

If a training with the same unique number (UUID) is transferred multiple times, it is always the last registration that remains 'active'. Previously registered data with the same unique identification number will be completely deleted.

This unique UUID number must be used for original registration, as well as for any correction to the training in question.

Example:

A training was originally registered/transferred for two participants. This later turned out to be incorrect: there were three participants. When correcting this registration/declaration, all three participants must be included in the declaration. If only the added, third participant is transferred, the training will be removed for the first two participants.

3.1.3. Structure of the data

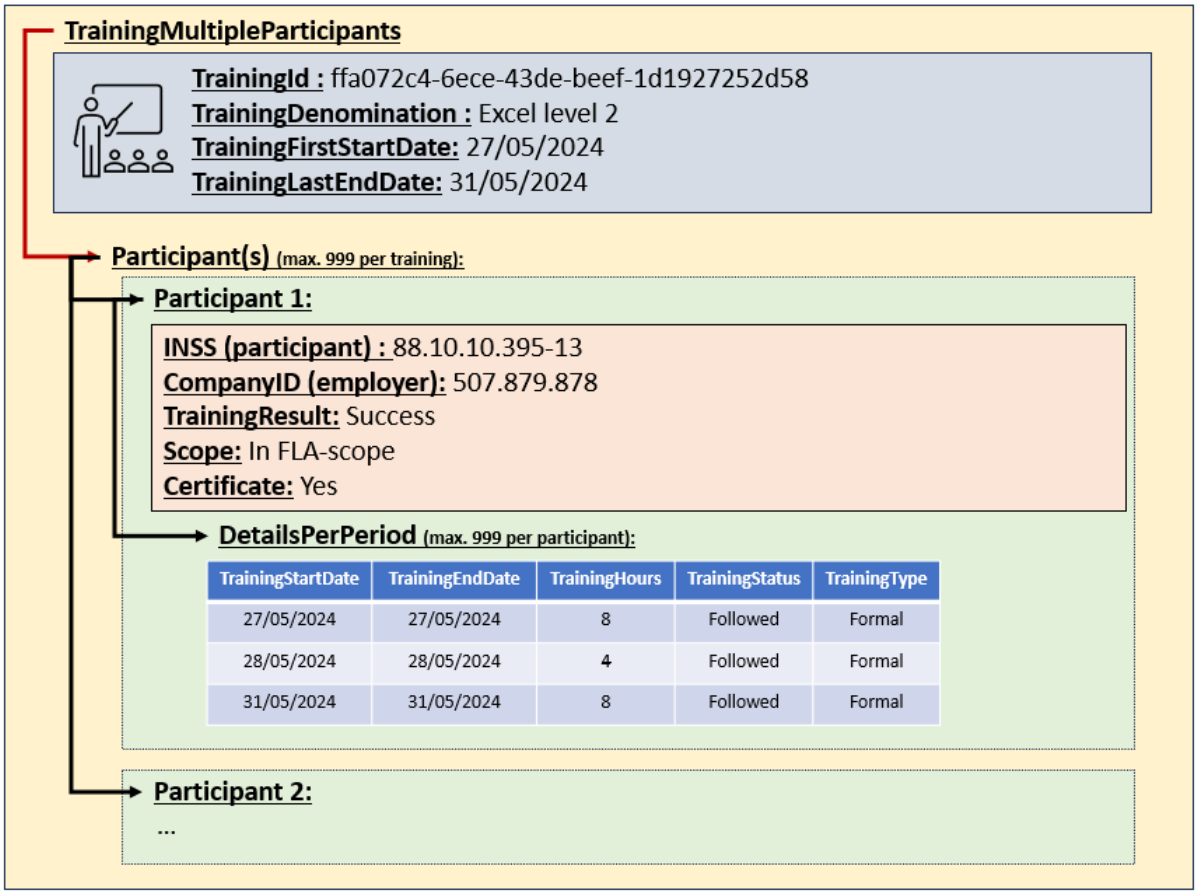
Through the API channel, the data of a training can be transferred using two different models/structures of data. It is recommended to make a choice regarding the data model to be used and then not to change it anymore. Using the API channel with different models is strongly discouraged.

The two possible data models are:

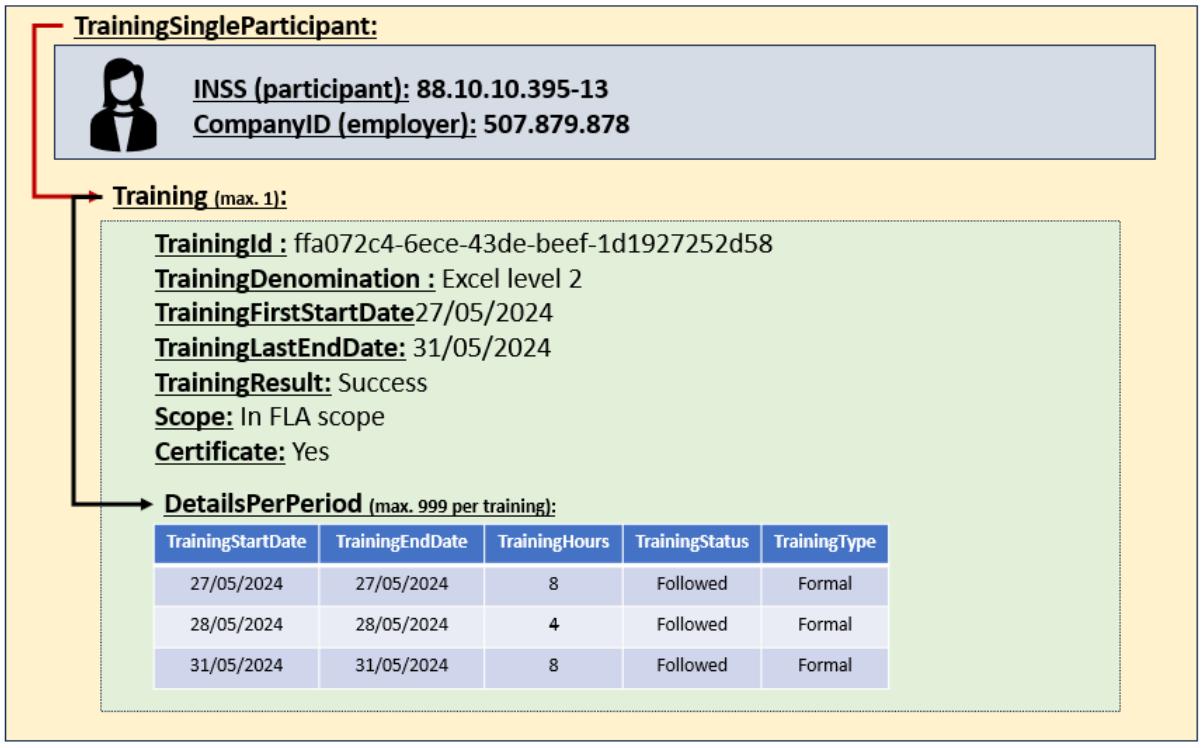
1) Data model 1: per training
   The declaration first specifies the training and then, within that training, you can specify a list of participants.
   Ratio: 1 training with 1 to 999 participants.

2) Data model 2: per participant and per training
   The participant's details are first defined and then you can specify a training for that participant.
   Ratio: 1 training with 1 participant.

The following diagrams show the difference between the two data models:



Figuur 1: Data model 1



Figuur 2: Data model 2

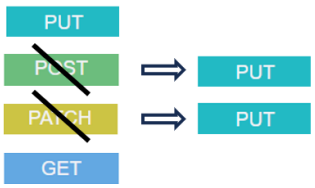
3.2. API calls

3.2.1. HTTPS Methods Used

The choice was made to include the logic of the **"POST"** and **"PATCH"** within a **"PUT"**. The call always receives a complete picture that is uploaded into the database as the current situation. From the user's point of view there is no difference and therefore there is no distinction.

3.2.2. Available paths

The tables below provide an overview of the available 'paths' and the corresponding provided methods. A short functional description is given for each 'path'. The following sections will go into more detail about each 'path' separately.

/multipleParticipants	
By means of the API call 'multipleParticipants' an organised training can be transferred for multiple participants (and multiple employers) (max. 999 participants for one training). The training can also be modified or consulted. The unique training number (a UUID) forms the unique key to make modifications/consultations. When registering, it is important that the complete training (with all participants) is always forwarded. (This is the *path* for data model 1 – per training). More information can be found at 3.2.3.	

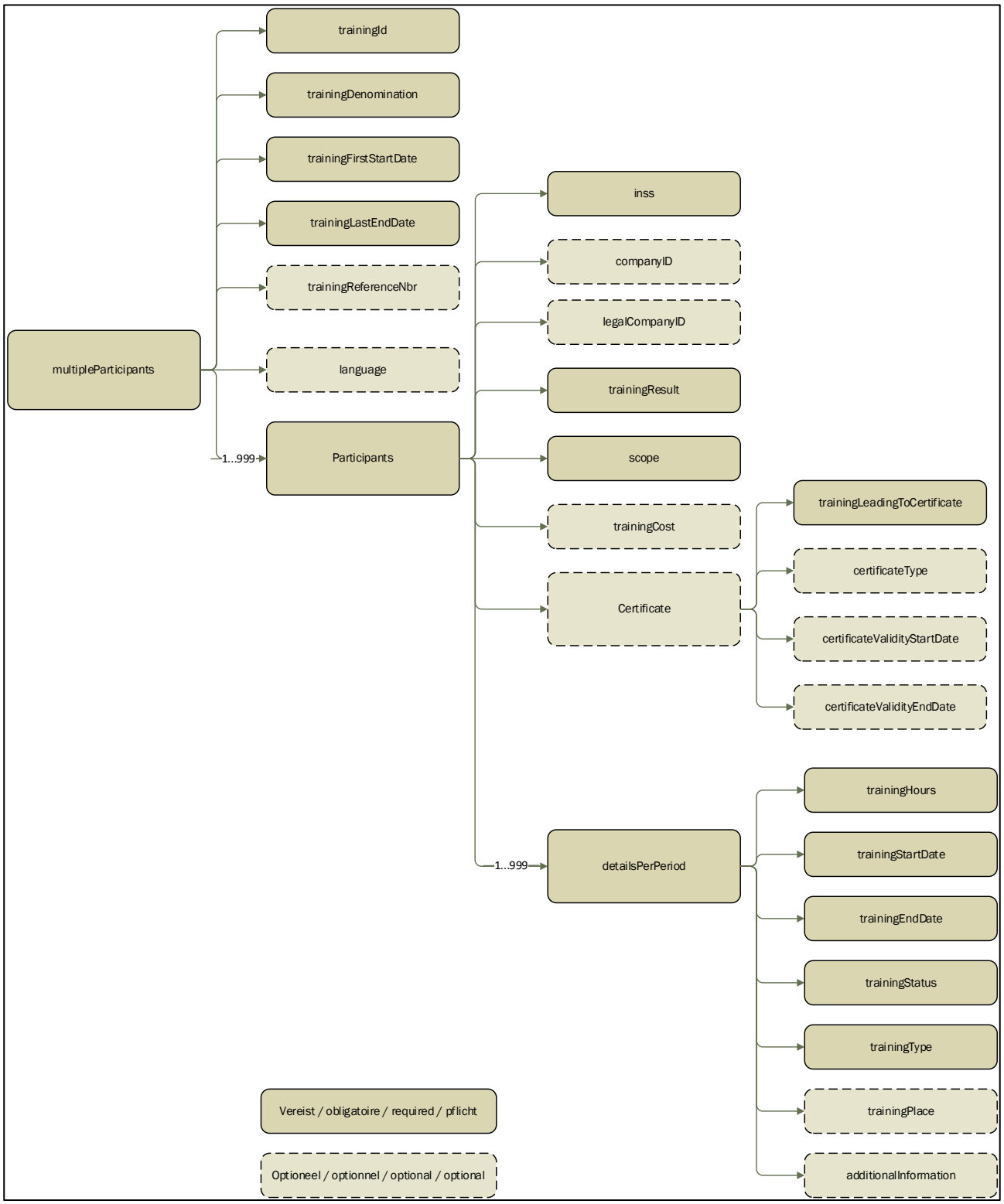
/singleParticipant	
The API call 'singleParticipant' starts from the participant's point of view. For one participant, one training can be registered (or consulted). The unique training number (a UUID) and the INSS form the unique key for making modifications/consultations. (This is the *path* for data model 2 – per participant and per training). More information can be found at 3.2.4	

/trainingHistory	
The training history of a specific citizen can be requested using the API call 'trainingHistory'. This applies to the trainings that the training provider has organised for this citizen (max. 999 trainings per consultation). Optionally, the request can be limited to one specific employer and/or to a specific period (for which the training provider has organised trainings). More information can be found at 3.2.5	GET

/training	
Using the API call 'training' a registered training can be deleted. This is always done based on the unique training number (UUID). More information can be found at 3.2.6	DELETE

3.2.3. Definition of the objects in '/multipleParticipants'

3.2.3.1. Structure of transferring, consulting or modifying a course



The data type, length and functional description of the fields can be found in the training provider's glossary.

3.2.3.2. PUT /multipleParticipants

Each training that is organised must be registered separately on the FLA platform. The input for the original registration (and any corrections) always consists of the complete photo of the training. This includes the unique identification number (type UUID), some metadata and all participants with at least one training period per participant (e.g. the main period of the training). Up to 999 participants can be transferred per training.

Optionally, multiple, specific training periods can be transferred per participant (max. 999 individual periods per participant). This can be used when a participant joined later, a participant was absent for a day, etc.

(This is the first data model, see 3.1.3).

The training time is always declared in hours so that the training provider does not have to take the participant's work regime into account.

A. Path

/providers/{companyId}/trainings/{trainingId}/multipleParticipants

B. Example scenarios

- ✓ A new training course is being organised that can be registered on the FLA platform.
- ✓ A correction is necessary because some additional participants have taken part in a training.

C. Input example

Methode: PUT

Path:

https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/trainings/ffa072c4-6ece-43de-beef-1d1927252d58/multipleParticipants

Body:


```
{
  "training": {
    "trainingId": "ffa072c4-6ece-43de-beef-1d1927252d58",
    "trainingDenomination": "Excel level 1",
    "trainingFirstStartDate": "2025-01-06",
    "trainingLastEndDate": "2025-01-10",
    "language": 1
  },
  "participants": [
    {
      "inss": 70081500504,
      "companyId": 880820673,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 2000,
          "trainingStartDate": "2025-01-06",
          "trainingEndDate": "2025-01-10",
          "trainingStatus": 1,
          "trainingType": 1,
          "trainingPlace": 3
        }
      ]
    },
    {
      "inss": 81511716525,
      "companyId": 206731645,
      "trainingResult": 1,
```

```
        "scope": 1,
        "detailsPerPeriod": [
          {
            "trainingHours": 1600,
            "trainingStartDate": "2025-01-06",
            "trainingEndDate": "2025-01-10",
            "trainingStatus": 1,
            "trainingType": 1,
            "trainingPlace": 3
          }
        ]
      }
  ]
}
```


The body content is also available in the sample file «PutTrainingMultipleParticipants_input.json».

D. Output example


```
{
  "training": {
    "trainingId": "ffa072c4-6ece-43de-beef-1d1927252d58",
    "trainingDenomination": "Excel level 1",
    "trainingFirstStartDate": "2025-01-06",
    "trainingLastEndDate": "2025-01-10",
    "language": 1
  },
  "participants": [
    {
      "inss": 70081500504,
      "companyId": 880820673,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 2000,
          "trainingStartDate": "2025-01-06",
          "trainingEndDate": "2025-01-10",
          "trainingStatus": 1,
          "trainingType": 1,
          "trainingPlace": 3
        }
      ]
    },
    {
      "inss": 81511716525,
      "companyId": 206731645,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 1600,
          "trainingStartDate": "2025-01-06",
```

```
                "trainingEndDate": "2025-01-10",
                "trainingStatus": 1,
                "trainingType": 1,
                "trainingPlace": 3
            }
        ]
    }
  ],
  "anomalies": []
}
```


The body content is also available in the sample file «PutTrainingMultipleParticipants_output.json».

E. Returncodes (Status Code)

If the processing is realized without error messages/anomalies, the response will be a “Status Code 200”.
If the processing can be realized, but with non-blocking anomalies, the response will be a "Status Code 200” + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400” + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

3.2.3.3. GET /multipleParticipants

This is based on the unique training number.

The input parameters are incorporated into the 'path' that makes the call.

A. Path

/providers/{companyId}/trainings/{trainingId}/multipleParticipants

B. Example scenarios

✓ After the registration, you want to check that you have not made a mistake.

C. Input example

Methode: GET

Path:

https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/trainings/ffa072c4-6ece-43de-beef-1d1927252d58/multipleParticipants

Body: No Body

D. Output example


```
{
  "training": {
    "trainingId": "ffa072c4-6ece-43de-beef-1d1927252d58",
    "trainingDenomination": "Excel level 1",
    "trainingFirstStartDate": "2025-01-06",
    "trainingLastEndDate": "2025-01-10",
    "language": 1
  },
  "participants": [
    {
      "inss": 70081500504,
      "companyId": 880820673,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 2000,
          "trainingStartDate": "2025-01-06",
          "trainingEndDate": "2025-01-10",
          "trainingStatus": 1,
          "trainingType": 1,
          "trainingPlace": 3
        }
      ]
    },
    {
      "inss": 81511716525,
      "companyId": 206731645,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 1600,
          "trainingStartDate": "2025-01-06",
          "trainingEndDate": "2025-01-10",
```

```
                "trainingStatus": 1,
                "trainingType": 1,
                "trainingPlace": 3
            }
        ]
    }
  ],
  "anomalies": []
}
```


The body content is also available in the sample file «GetTrainingMultipleParticipants_output.json».

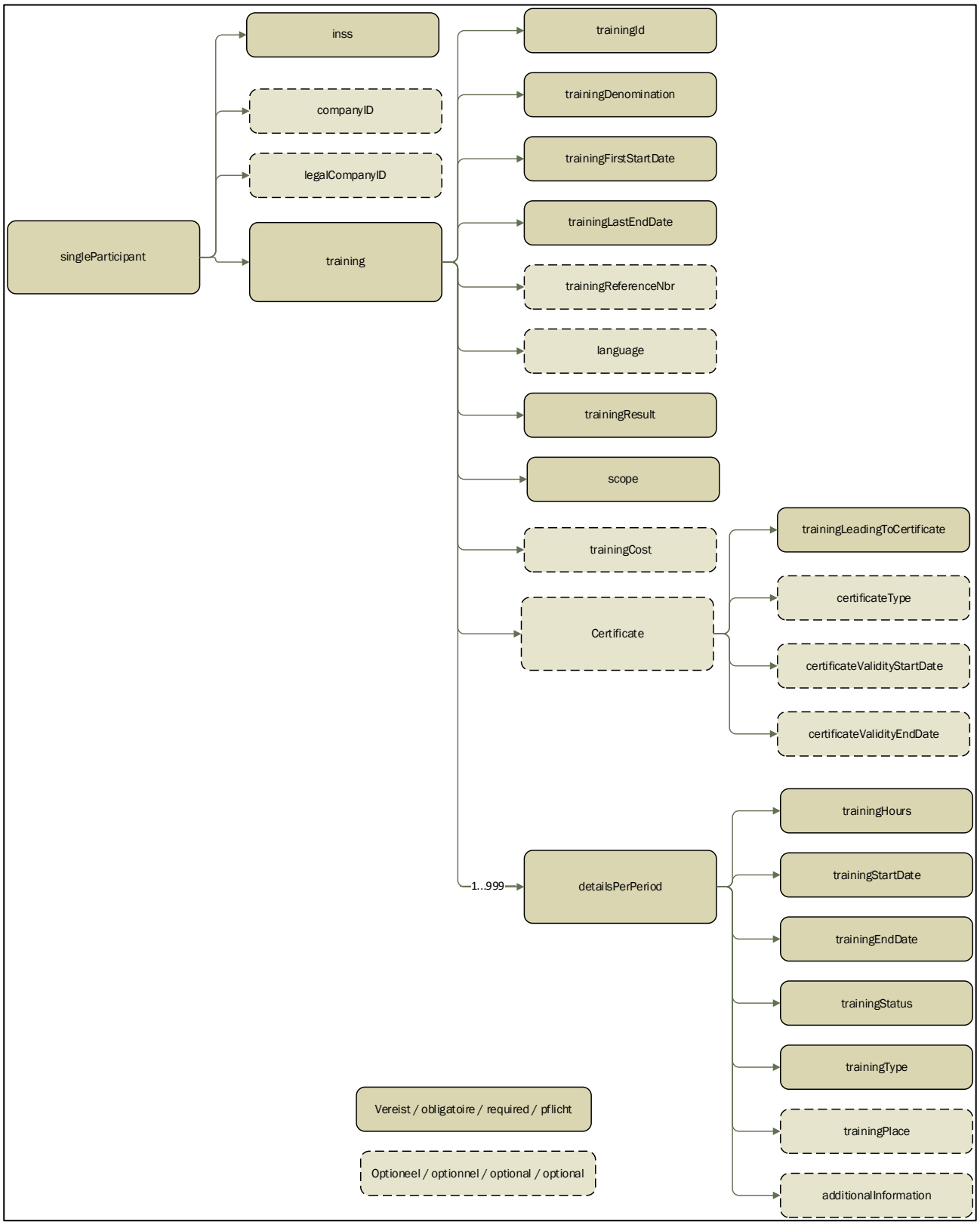
E. Retourcodes (Status Code)

If the processing is realized without error messages/anomalies, the response will be a “Status Code 200”.
If the processing can be realized, but with non-blocking anomalies, the response will be a "Status Code 200” + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400” + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

3.2.4. Definition of the objects in '/singleParticipant'

3.2.4.1. Structure of transferring, consulting or correcting a training for one participant.



The data type, length and functional description of the fields can be found in the training provider's glossary.

3.2.4.2. PUT /singleParticipant

The **PUT** /singleParticipant allows to register one training for one specific participant on the FLA platform. Changes can also be registered via the PUT (always the full picture, see 3.1.2).

This way of registering trainings follows the second data model (see 3.1.3).

Up to 999 individual training periods are possible per participant and training. The training time must always be indicated in hours so that the training provider does not have to take into account the participant's work regime.

A. Path

/providers/{companyId}/participants/{inss}/trainings/{trainingId}/singleParticipant

B. Example scenarios

- ✓ A citizen has followed a training course to support his job (expansion of knowledge).
- ✓ The citizen in question was absent during a certain training day. The registration needs to be corrected.

C. Input example

Methode: PUT

Path:
https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/participants/70081500504/trainings/f973143f-f557-4e7c-8410-911c6aeb0878/singleParticipant

Body:


```
{
  "inss": 70081500504,
  "companyId": 880820673,
  "training": {
    "trainingId": "f973143f-f557-4e7c-8410-911c6aeb0878",
    "trainingDenomination": "Word level 1",
    "trainingFirstStartDate": "2025-03-17",
    "trainingLastEndDate": "2025-03-19",
    "language": 1,
    "trainingResult": 9,
    "scope": 1,
    "detailsPerPeriod": [
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-17",
        "trainingEndDate": "2025-03-17",
        "trainingStatus": 1,
        "trainingType": 1
      },
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-19",
        "trainingEndDate": "2025-03-19",
        "trainingStatus": 1,
        "trainingType": 1
      }
    ]
  }
}
```


The body content is also available in the sample file «PutTrainingSingleParticipant_input.json».

D. Output example


```
{
  "inss": 70081500504,
  "companyId": 880820673,
  "training": {
    "trainingId": "f973143f-f557-4e7c-8410-911c6aeb0878",
    "trainingDenomination": "Word level 1",
    "trainingFirstStartDate": "2025-03-17",
    "trainingLastEndDate": "2025-03-19",
    "trainingReferenceNbr": "string",
    "language": 1,
    "trainingResult": 1,
    "scope": 1,
    "detailsPerPeriod": [
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-17",
        "trainingEndDate": "2025-03-17",
        "trainingStatus": 1,
        "trainingType": 1
      },
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-19",
        "trainingEndDate": "2025-03-19",
        "trainingStatus": 1,
        "trainingType": 1
      }
    ]
  },
  "anomalies": []
}
```


The body content is also available in the sample file «PutTrainingSingleParticipant_output.json».

E. Returncodes (Status Code)

If the processing is realized without error messages/anomalies, the response will be a “Status Code 200”.
If the processing can be realized, but with non-blocking anomalies, the response will be a "Status Code 200” + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400” + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

3.2.4.3. GET /singleParticipant

The API is called for the training of one participant.

A. Path

/provider/{companyId}/participant/{inss}/training/{trainingId}/singleParticipant

B. Example scenarios

✓ A previously registered training must be retrieved to check the input.

C. Input example

Methode: GET

Path:
https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/participants/70081500504/trainings/f973143f-f557-4e7c-8410-911c6aeb0878/singleParticipant

Body: No body

D. Output example


```
{
  "inss": 70081500504,
  "companyId": 880820673,
  "training": {
    "trainingId": "f973143f-f557-4e7c-8410-911c6aeb0878",
    "trainingDenomination": "Word level 1",
    "trainingFirstStartDate": "2025-03-17",
    "trainingLastEndDate": "2025-03-19",
    "trainingReferenceNbr": "string",
    "language": 1,
    "trainingResult": 1,
    "scope": 1,
    "detailsPerPeriod": [
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-17",
        "trainingEndDate": "2025-03-17",
        "trainingStatus": 1,
        "trainingType": 1
      },
      {
        "trainingHours": 800,
        "trainingStartDate": "2025-03-19",
        "trainingEndDate": "2025-03-19",
        "trainingStatus": 1,
        "trainingType": 1
      }
    ]
  },
  "anomalies": []
}
```


The body content is also available in the sample file «GetTrainingSingleParticipant_input.json».

E. Returncodes (Status Code)

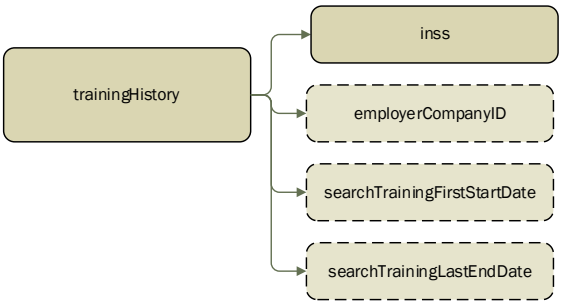
If the processing is realized without error messages/anomalies, the response will be a “Status Code 200”.
If the processing can be realized, but with non-blocking anomalies, the response will be a "Status Code 200” + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400” + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

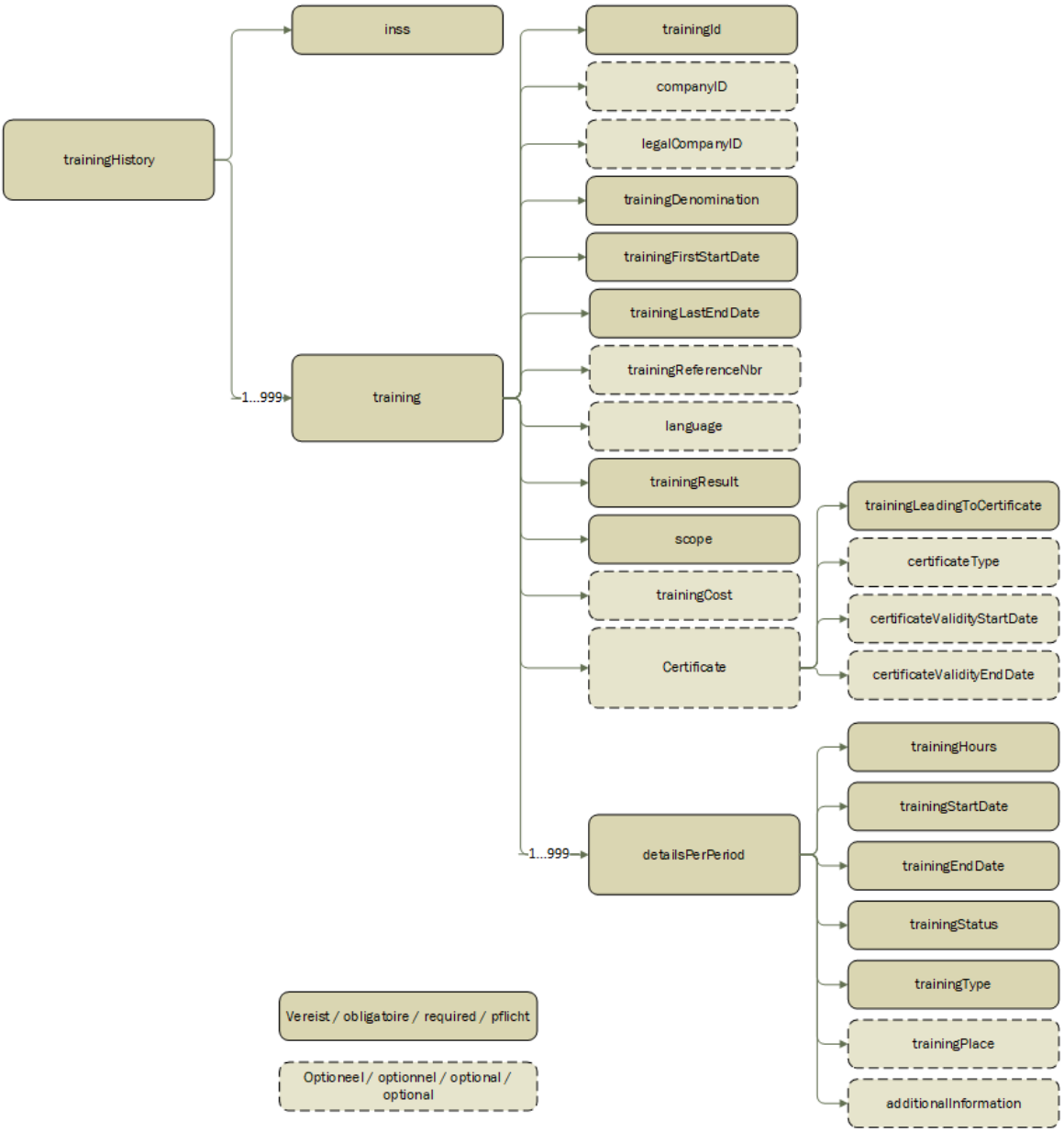
3.2.5. Definition of the objects in "/trainingHistory"

3.2.5.1. Structure of consulting the training history

Request GET '/trainingHistory':



Response GET '/trainingHistory':



The data type, length and functional description of the fields can be found in the training provider's glossary.

3.2.5.2. GET /trainingHistory

The training history of a specific citizen can be called up via the API call 'trainingHistory'. This for all trainings that the training provider has organised for this citizen.

Optionally, additional parameters can be added to the path (= query parameters) to restrict the search.

The following optional fields are possible:

- companyID = the employer for whom the trainings of the citizen in question are requested.
  - If not specified, all trainings from the training provider will be returned.
- searchTrainingFirstStartDate
- searchTrainingLastEndDate

A maximum of 999 trainings are returned per search.

A. Path

- Basis-path (all trainings of a citizen):
  - /providers/{employerCompanyId}/trainingHistory/{inss}

- More extended path, with optional query parameters (limited to a specific employer and/or period):
  - /providers/{companyId}/trainingHistory/{inss}?employerCompanyId={employerCompanyId}
  - /providers/{companyId}/trainingHistory/{inss}?employerCompanyId={employerCompanyId}&searchTrainingFirstStartDate={searchTrainingFirstStartDate}&searchTrainingLastEndDate={searchTrainingLastEndDate}
  - /providers/{companyId}/trainingHistory/{inss}?searchTrainingFirstStartDate={searchTrainingFirstStartDate}& searchTrainingLastEndDate={searchTrainingLastEndDate}
  - /providers/{companyId}/trainingHistory/{inss}?employerCompanyId={employerCompanyId}&searchTrainingFirstStartDate={searchTrainingFirstStartDate}&searchTrainingLastEndDate={searchTrainingLastEndDate}

B. Example scenario's

✓ The citizen would have liked to have an overview of all trainings followed by the training provider.
✓ Company X would like to receive the training history for the period 2022-2024 for employee Y.

C. Input example

Methode: GET

Path:
https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/trainingHistory/70081500504

Body: No body

D. Output example


```
{
  "inss": 70081500504,
  "trainings": [
    {
      "trainingId": "ffa072c4-6ece-43de-beef-1d1927252d58",
      "companyId": 880820673,
      "trainingDenomination": "Excel level 1",
      "trainingFirstStartDate": "2025-01-06",
      "trainingLastEndDate": "2025-01-10",
      "language": 1,
      "trainingResult": 1,
      "scope": 1,
      "detailsPerPeriod": [
        {
          "trainingHours": 2000,
          "trainingStartDate": "2025-01-06",
```

```
            "trainingEndDate": "2025-01-10",
            "trainingStatus": 1,
            "trainingType": 1,
            "trainingPlace": 3
          }
        ]
      },
      {
        "trainingId": "f973143f-f557-4e7c-8410-911c6aeb0878",
        "companyId": 880820673,
        "trainingDenomination": "Word level 1",
        "trainingFirstStartDate": "2025-03-17",
        "trainingLastEndDate": "2025-03-19",
        "trainingResult": 1,
        "scope": 1,
        "detailsPerPeriod": [
          {
            "trainingHours": 800,
            "trainingStartDate": "2025-03-17",
            "trainingEndDate": "2025-03-17",
            "trainingStatus": 1,
            "trainingType": 1
          },
          {
            "trainingHours": 800,
            "trainingStartDate": "2025-03-19",
            "trainingEndDate": "2025-03-19",
            "trainingStatus": 1,
            "trainingType": 1
          }
        ]
      }
    ],
    "anomalies": []
}
```


The body content is also available in the sample file « GetTrainingHistoryculation_output.json ».

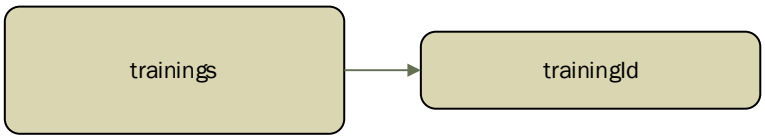
E. Returncodes (Status Code)

If the processing is realized without error messages/anomalies, the response will be a “Status Code 200”.
If the processing can be realized, but with non-blocking anomalies, the response will be a "Status Code 200” + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400” + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

3.2.6. Definition of the objects in "/training"

3.2.6.1. Structure for deleting a registered training



The data type, length and functional description of the fields can be found in the training provider's glossary.

3.2.6.2. DELETE /training

A training can be deleted from the FLA platform via the API call 'training'.

A. Path

/providers/{companyId}/trainings/{trainingId}

B. Example scenario's

- ✓ The training did not take place and was incorrectly registered. It should be removed.

C. Input example

Methode: DELETE

Path:

https://services.socialsecurity.be/REST/federalLearningAccount/trainingProvider/v1/providers/406798006/trainings/3fa85f64-5717-4562-b3fc-2c963f66afa6

Body: No body

D. Output example

```
HTTPS statuscode '204 No Content'
```

E. Returncodes (Status Code)

If the processing is realized without error messages/anomalies, the response will be a "Status Code 200".
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400"
If a technical problem occurs, the response will be a "Status Code 500".

See chapter 3.3 for more detailed information about the status codes and anomalies.

3.3. Status code and anomaly block

Upon successful oAuth authorization, for every 'GET', 'PUT' or 'DELETE' call, an https response is returned from the FLA platform. This always contains a status code.

If the processing is realized without anomalies, the response will be a "Status Code 200".
If the processing can be realized, but with non-blocking anomalies (warning anomalies), the response will be a "Status Code 200" + the list of warning anomalies.
If the processing cannot be realized because of a blocking anomaly, the response will be a "Status Code 400" + list of blocking anomalies.
If a technical problem occurs, the response will be a "Status Code 500".

The anomalies block consists of the following fields:

Field name	Type	Description
Anomalyclass	String	Level of the anomaly (in max 2 char) W => Warning (non-blocking anomaly(ies)) B => Blocking – Error message (blocking anomaly(ies))
Tagname	String (Max:100)	Name of the field that generated the anomaly, e.g.: "trainingStartDate"
Path	String (max:500)	Exact location of the anomaly in the Json
Errorid	String (max:250)	The error code
Label	String	A description specified in multiple languages

Below are some examples of these anomaly blocks:

1) Status code 200:"Created" - with **non**-blocking anomalies (=warning anomalies).


```
{
  "anomalies": [
    {
      "anomalyClass": "W",
      "tagName": "trainingStatus",
      "path": "trainingId:3fa85f64-5717-4562-b3fc-2c963f66afa6",
      "errorId": "FLA39-187",
      "label": {
        "nl": "Status van de opleiding - Reservering onmogelijk",
        "fr": "Statut de la formation - Réservation impossible",
        "de": "Fortbildungsstatus - Reservierung unmöglich",
        "en": "Training status - Reservation impossible"
      }
    }
  ]
}
```


2) Status code 400: "Bad Request" - with blocking anomalies.


```
{
  "anomalies": [
    {
      "anomalyClass": "B",
      "tagName": "trainingStatus",
      "path": "trainingId:3fa85f64-5717-4562-b3fc-2c963f66afa6",
      "errorId": "FLA39-511",
      "label": {
        "nl": "Status van de opleiding - Onverenigbaar met het resultaat van de opleiding",
        "fr": "Statut de la formation - Incompatibilité avec le résultat de la formation",
        "de": "Fortbildungsstatus - Nicht mit dem Fortbildungsergebnis kompatibel",
        "en": "Training status - Incompatible with the result of the training"
```

```
        }
      }
    ],
    "type": "about:blank",
    "title": "Bad Request",
    "status": 12,
    "detail": "The input message is incorrect"
}
```


3) Status code 500: "Internal Server Error"


```
{
    "type": "about:blank",
    "title": "Unexpected Error",
    "status": 500,
    "detail": "putMultipleParticipants.arg3.participant.inss: must not be null"
}
```


4. Contact

If you have any questions, please consult the "**Contact**" section found under "**Frequently Asked Questions**" on the site https://federallearningaccount.be/