

**Forsikring
& Pension**

**Guide for Claim History
business, agriculture
and vehicle**

**Appendix 1
REST API**

Version 3.0 Final

Document information

Title:	Guide for Claim History, Appendix 1 REST API
Project:	EDI Office industry coordinated data exchange
By:	Morten Lassen, F&P IT Department
Document contributors:	
Approved by:	Mette Ellermann Jespersen, F&P EDI Office
Person responsible for document:	Mette Ellermann Jespersen, F&P EDI Office
Distribution:	EDI Office, F&P Distributed to stakeholders in the data exchange
Comment:	The document can be requested from F&P

Change log

Version	Date	By	Changed pages or sections
2.0 Draft B	24/09/2018	MLA	First edition
2.0 Draft C	22/10/2018	MLA / ADI	Corrections after working group meeting on 12 October 2018
2.0 Draft C	06/12/2018	MLA / ADI	Corrections after engineers' meeting on 19 November 2018 and that the Web solution will be discontinued. Description of JWT tokens
2.0 Draft D	June 2019	MLA / MEJ	Corrections after test period
2.0 Draft E	September 2019	MLA / MEJ	Corrections up to implementation
2.0 Final	31 October 2019	MLA / MEJ	Corrections up to implementation
3.0 Draft F	January 2020	MLA / MEJ	Addition of industry group 002 Private, 003 Pleasure Craft and 005 Accident and minor adjustments
3.0 Draft G	March 2020	MLA / MEJ	API changed from version 1.0 to version 3.0 and minor adjustments
3.0 Draft H	June 2020	MLA / MEJ	DEMOEDI changed to TESTEDI
3.0 Final	September 2020	MLA / MEJ	No changes

Abbreviations and definitions:

https	An encrypted version of http used for data communication via the Internet
WS, Webservice	Technology for the exchange of data via the Internet
Online company	The company exchanges online via API (integrated)
Web company	The company exchanges via WebEDI (web based)
API	Application Program Interface
REST	Representational State Transfer
REST API	REST based Program interface using http requests for data exchange via GET, PUT, POST and DELETE
Token	A dynamic key/access ticket identifying the sender
OAuth2	An open standard for providing access to data

JSON	JavaScript Object Notation. A text-based data exchange format
JWT	JSON Web Token
JWS	JSON Web Signature

References:

DateTime data types follow the ISO 8601 standard

CCYY-MM-DDThh:mm:ss[Z|(+|-)hh:mm]. For example 2016-12-31T20:00:01.

See https://en.wikipedia.org/wiki/ISO_8601

Dates without values are indicated with a null value

Description of REST API

https://en.wikipedia.org/wiki/Overview_of_RESTful_API_Description_Languages

Description and recommendation for ClientId and ClientSecret

<https://www.oauth.com/oauth2-servers/client-registration/client-id-secret>

Table of Contents:

1. Introduction	5
2. Flow diagram	5
3. Security	5
Authorization	6
4. ClaimService API	11
TEST	11
Production	11
Online documentation	11
Operations	13
Status	13
Companies	14
CloseMessage	16
OpenMessage	16
HistoryRequest	17
HistoryRequest – Coverage levels	22
Server checks	24
Return codes and text	25
5. Company API	26
Preconditions	26
JWT tokens	26
Operations	28
Status	28
HistoryRequest	28
Return codes and text	29
REST API with Oauth2 template	29
6. Test	29
Technical test report	29

1. Introduction

This document describes a REST API for online claim history.

REST API security is provided by a standardised OAuth2 architecture which issues an Access Token. This access token grants the company access to the Claim History API.

The REST API returns a HTTP status 200/OK if the operation was successful. Return data is sent in the HTTP body as a JSON string. In case of an error, a standard HTTP status code is returned – see the section on return codes – page 25.

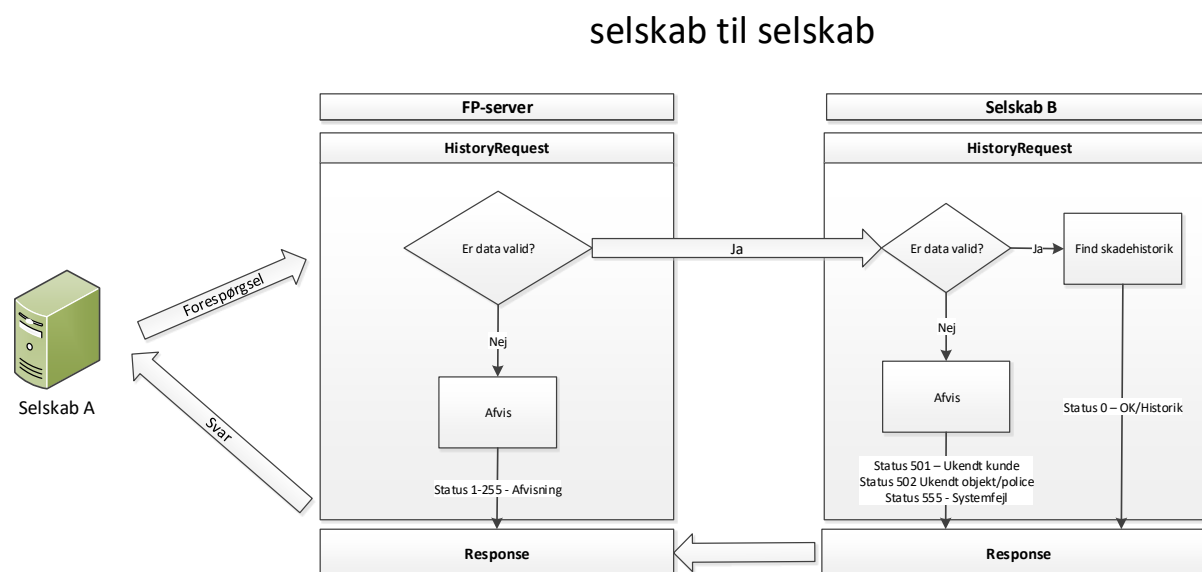
The description of an access token and security in general can be found in chapter 3.

2. Flow diagram

The following describes the flow between and F&P's FP server.

Before a company can call the Claim History API, it has to retrieve an access token, cf. section 3. This token is issued by F&P authorization server based on a client ID and a client secret unique to the company.

Company A submits a request by calling the **HistoryRequest** operation on FP server. The server validates the requests and rejects it in case of error.



FP server **forwards** the request to a company's REST API via the **HistoryRequest** operation. The responding company B must respond immediately and the response from the company is returned to company A. See section 5.

3. Security

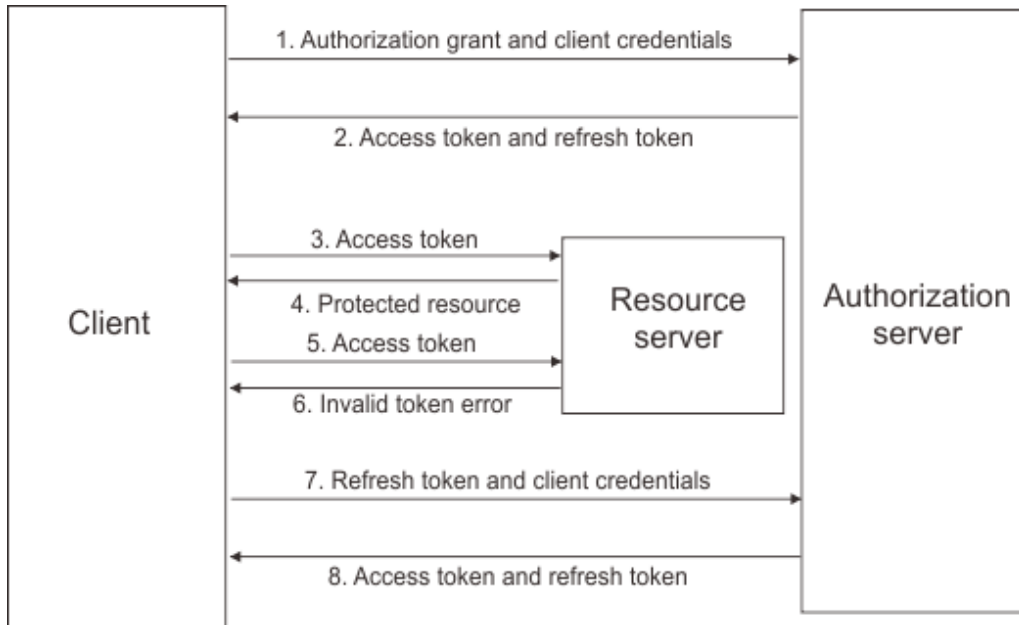
F&P's Claim History REST API is protected by the following technologies:

- HTTPS / SSL / TLS 1.2 - ensuring an encrypted data connection
- IP whitelist - preventing unauthorized access
- Access token - identifying the company

A company is allocated a client ID and a client secret, which is used for attaining an access token - see next section. A client secret is confidential.

Authorization

Granting Access Tokens utilize the OAuth2 standard.as illustrated below.



Access token

Before a company can call an operation from F&P’s REST API, it must request an access token. This token is issued by the EDI authorization server based on a client ID (ClientId) and a client secret (ClientSecret). In addition to the access token, a refresh token is returned, which can be used to retrieve a new access token when the original expires.

An access token expires after 20 minutes and must be used for all subsequent calls to the server’s REST API.

Access tokens have independent life-cycles, meaning that multiple active tokens may have been granted to the same company if the company has made multiple simultaneous calls.

ClientId and ClientSecret

A ClientId is unique to the company and is generated by the FP server. A ClientId is a 32 character hex string and can be viewed in a companys API administration web page.

A ClientSecret is a secret, cryptographically generated string known only to the company and the authorization server. As the authorization server stores the hash value of this string, it is not possible to receive information about a ClientSecret after it has been generated. A company is able to generate a new ClientSecret via API administration.. A ClientSecret is valid for 365 days and must be changed by the company before expiry.

A reminder is automatically sent to API Administrators 30 days before expiry.

Use the following addresses for token requests:

Test: <https://testedi.forsikringopension.dk/authserver/oauth/token>

Production: <https://edi.forsikringopension.dk/authserver/oauth/token>

The following information is required as POST parameters:

"Grant_type" must have the value: `client_credentials`

"Scope" must have the value: `claimhistory`

"Client_id" and "Client_secret"

Example of an an access token request:

```
POST /authserver/oauth/token HTTP/1.1
Host: demoedi.forsikringopension.dk
grant_type=client_credentials
&client_id=a18344f95c694272b6203b35a1ffe059
&client_secret=NPklLbBdD08yymkCABapH3Af1ue17I2gZEKpjyYXYhs2
&scope=claimhistory
```

When a valid client ID and client secret are provided, the server returns an access token and a refresh token.

Response example:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
  "access_token": "Vv7gPtgWE4YQ6JqgweOv5k_OFouRsT6Y7J3BKgxClTotPCFSbSUrdT7ieqBN
0vucljtgZDk1m1FI7qwn5I4Kh_AcVtOnvJYPZFDViMHjUf07R16cLxPe2VPrc_peGsXQbFncjtd6S1HEGNw
7Zo6eeNkQY1t2ga97PakAWmwAiXOdnjnUsxcbDoG_W8bnQBNEkPEGetXHtvQ9V21hLoxeoL4UQ6pe3Xyrfr
uAx-ZmpEvIZ0wKAtdWTSaebWXHnudYJwtXoAA9IHPfSGJWHdD92vIgljxrNEj7Ln6ycW7ICrRIUk2Fxpik6
woPit4szcCdic1_nhVYnySzdjBM7wHeRv-2cV4-IIvz_6u4i7od_zxjh6o1m1pfGoy9-e-5vtXOQUA",
  "token_type": "bearer",
  "expires_in": 1200,
  "refresh_token": "x-GAzugdkRSQUh9akGI1ROYEe9Is0uAE7URgd2P6ezigovhbccpGNggYHA4r
IO_H9aA3Ue7JqHpakl4GtaQbK2RfWEM8kdeUCIMQ6am4p5ASJl8ktxL81-qq7iIGQ1b4pj9sDu5Ddov7Vv
6sZbVbIkJmyx2JfaLuvVElujIrmLE8AGdtFF1340fr9-87r-R7sZaBrXi1qp0AVxT1RLPGLA9A0On5hwlQ
6-qXangZAP6tq_sIbhEp5VnAIbhfPrHMA6h81NKA-MwAs2fxl4oRtjP7W1N6oNPrsh-zlsG82HHq3yfbY5
B07WYLVJBteWQsa525s1CHhkVqofURm57i60tfcKdYOReW0ZOrnyStmcyHzWC_D_q33argKWfwxE33Q"
}
```

An access token (`access_token`) must be provided as a HTTP header element for all subsequent calls as follows: `Authorization: Bearer <token>`

An access token expires after a short period determined by the issuer. In the example above, it expires (`expires_in`) after 1200 seconds, i.e. 20 minutes.

Example of a status operation call:

```
GET /api/v3/claimservice/claimhistory/status
Host: demoedi.forsikringopension.dk
Authorization: Bearer Vv7gPtgWE4YQ6JqgweOv5k_OFouRsT6Y7J3BKgxClTotPCFSbSUrdT7ieqBN
0vucljtgZDk1m1FI7qwn5I4Kh_AcVtOnvJYPZFDViMHjUf07R16cLxPe2VPrc_peGsXQbFncjtd6S1HEGNw
7Zo6eeNkQY1t2ga97PakAWmwAiXOdnjnUsxcbDoG_W8bnQBNEkPEGetXHtvQ9V21hLoxeoL4UQ6pe3Xyrfr
uAx-ZmpEvIZ0wKAtdWTSaebWXHnudYJwtXoAA9IHPfSGJWHdD92vIgljxrNEj7Ln6ycW7ICrRIUk2Fxpik6
woPit4szcCdic1_nhVYnySzdjBM7wHeRv-2cV4-IIvz_6u4i7od_zxjh6o1m1pfGoy9-e-5vtXOQUA
```


Example of a response to a correctly submitted token:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
  "ResultDate": "2018-09-25T13:44:59.5565937+02:00",
  "ResultCode": 0,
  "ResultText": "Insurance company for test #1"
}
```

Example of a response to an expired or incorrect token:

```
HTTP/1.1 401 Unauthorized
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
  "Message": "Authorization has been denied for this request."
}
```

Refresh token

A refresh token expires after 48 hours and is renewed every time a new access token is retrieved. When an access token has expired, the company has to retrieve a new access token by using the refresh token in the request.

The following information is required as POST parameters:

“Grant_type” must be `refresh_token`

“Client_id” is the same used when requesting an access token

“Refresh_token” must contain the refresh_token string returned from the access token request.

Example of a token request:

```
POST /authserver/oauth/token HTTP/1.1
Host: demoedi.forsikringogpension.dk
grant_type=refresh_token
&client_id=a18344f95c694272b6203b35a1ffe059
&refresh_token=x-GAzugdkRSQUh9akGI1ROYEe9Is0uAE7URgd2P6ezigovhbccpGNnggyHA4r
IO_H9aA3Ue7JqHpakl4GtaQbK2RfWEM8kdeUCIMQ6am4p5ASJl8ktxL81-qq7iIGQ1b4pj9sDu5Ddov7Vv
6sZbVbIkJmyx2JfaLuvVElujIrmLE8AGdtFF1340fr9-87r-R7sZaBrXi1qp0AVxT1RLPGLA9A0On5hw1Q
6-qXangZAP6tq_sIbhEp5VnAIbhfPrHMA6h81NKa-MwAs2fx14oRtjp7W1N6oNPrsh-zlsG82HHq3yfbY5
B07WYLVJBteWQsa525s1CHhkVqofURm57i60tfcKdYOReW0ZOrnyStmcyHzWC_D_q33argKWfwxE33Q
```

When a valid client ID and refresh token are provided, the server returns a new access token and a new refresh token.

Response example:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
```

```
{
  "access_token": fo1BGSAhfudG46-gFmNHWAzA6ZfqBkMfVzWcoVn4uniGeQLJ50YjIs02ou6sJE
  SepOhaTSlAzdRY8w8S9of7s2jzF5CZvjU6vGb1J6yy1m7QLLdg2kirrO70R6oXK_ygCxfue68zhFqEu0E
  Kijvgpjz6Sp0NT-go-dPKfk113CcGD3Pgzx77R3x2HaAKc_bupbTwBYjcZtMSvKH1-Ac5bVGPPlik8n5F
  YpWLgbWLLMRK7ejifq1BiBsiAq4koQEct1GQAKpGacRNu0twlhkk1NOFE8jf3_cOHfI3zI-t-QZhL-UDGM
  Du32TftniZjiCA5ProNubQW87TkjjgRQEA9I525dmeFCfkg_-Mz5tirQfn128QHIXNduJu3LjluY9otUA",
  "token_type": "bearer",
  "expires_in": 1200,
  "refresh_token": "SIHDB5pdaBuofHRLLvyBrNdfnMxDkYq6kBXb_dV8-mm28TTZgYj-x6yLs49
  oMjbr3g2h89uSoiGHi7K1ZUjVLHu0UikrXW2o15ejfJ_ad2H_Bdboxyrd1-yzKKxpQKJK3PFb0N583UQD
  qyyHpA7CtTJ_sdC-BnO7sYCctez-rJSH1zTBKq5eCIx7yG2ZTK0sE6kBVfaE5mz8QgpbneQrxwIiqvxVR
  9KAR2-sjOaD6ZasiPXDXMe5duColwsWY0pYqOUIWgYU19v1guYQANrXGqOFbNNO4QclQfdws_joC1RKuR
  FsesL4nfJoNvFFGIQdveGzGG6sj-sr9rE3oER6s1dsLoHDQyY-vKTXRk0h0BN1U-p4k9td3E1QnnIiOzKA"
}
```

If both the access token and the refresh token have expired, a new access token must be requested using the grant type `client_credentials`.

4. ClaimService API

The following section describes F&P's REST API for Claim History.

Via this API, a company can submit a claim history request to another company. It is also possible to retrieve a company list and to open and close access to your own company – see guide for online claim history.

All participating companies must make a corresponding REST API which the FP server can call when a request is forwarded to the responding company – see page 26 for a description of a company's REST API.

TEST

API Endpoint: <https://testedi.forsikringogpension.dk/api/v3/claimservice/claimhistory>

IP address from which the FP server is calling: 86.48.32.198

Production

API Endpoint: <https://edi.forsikringogpension.dk/api/v3/claimservice/claimhistory>

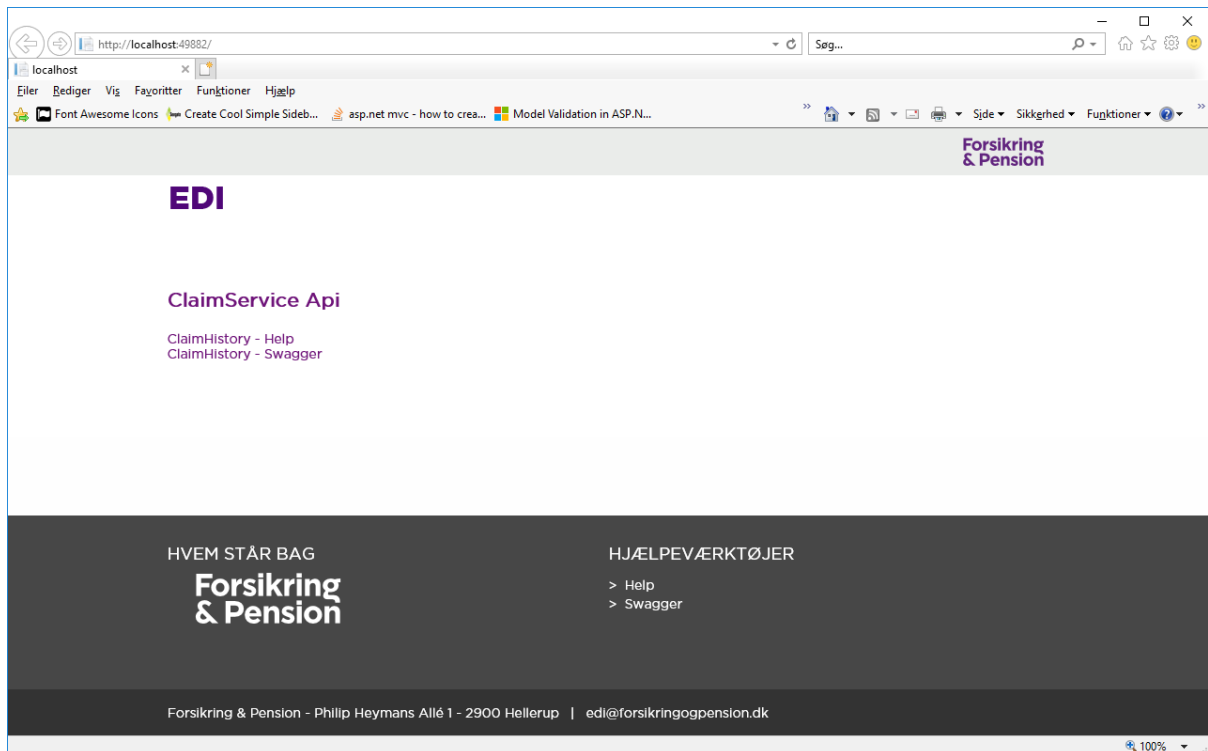
IP address from which the FP server is calling: 86.48.32.197

Online documentation

Online documentation for the ClaimService API is available at the address below:

<https://testedi.forsikringogpension.dk/api/v3/ClaimService/Swagger>

Swagger describes all calls and data structures as well as integrates token and test operations. From Swagger it is also possible to test the authorization, where an access token is retrieved.



The online documentation exists in 2 versions:


- **Help**, describing all operations and data structures
- **Swagger**, describing all operations and data structures with integrated token based authentication and testing of operations

It is also possible to test authorization, where an access token is retrieved, from Swagger.

Operations

The following describes the operations exposed by the REST API. As previously mentioned, this can also be found in an online version.

ClaimHistory		Show/Hide List Operations Expand Operations
GET	/claimhistory/status	Returns Company Information
GET	/claimhistory/companies	Returns list of Companies
GET	/claimhistory/closemessage	Send Company Close Message
GET	/claimhistory/openmessage	Send Company Open Message
POST	/claimhistory/historyrequest	Returns Claim History Response

[BASE URL: /api/v3/ClaimService , API VERSION: v3] INVALID 

Fields that have no value can either be ignored or given a null value.

Status				
This operation is used to test authorization and to obtain the status of a company. It returns a date/time, code, text and open/close status.				
Input				
Parameter	Type	Required	Length	Description
companyId	String	No	9	VIR number for the company for which status is required Blank=Displays status for own company
Output				
Parameter	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time
ResultCode	Integer	Yes		Return code
ResultText	String	Yes	255	Return text. <company name>
OpenStatus	Boolean	Yes		Returns the company's open/close status. true=open, false=closed
GET api/v3/claimservice/claimhistory/status?companyId={companyId}				
Example of response <pre>{ "ResultDate": "2018-09-25T13:44:59.5565937+02:00", "ResultCode": 0, "ResultText": "Insurance company for test #1", "OpenStatus": true }</pre>				

Companies				
<p>This operation is used to retrieve a list of companies registered for claim history. The operation returns a date/time, code, text and a list of companies and the industry/product groups for which the company in question is registered.</p>				
Input				
Parameter	Type	Required	Length	Description
Output				
Parameter	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time
ResultCode	Integer	Yes		Return code
ResultText	String	Yes	255	Return text
Companies	List<Company>	Yes		
<Company>				
Name	String	Yes	35	Company name
Id	String	Yes	9	VIR number
Address1	String	Yes	35	Address 1
Address2	String	No	70	Address 2
PostalCode	String	Yes	4	Postal code
City	String	Yes	35	City
IndustryProductGroups	List<String>	Yes		List of industry/product groups for which the company is registered. Possible values are: 001/001 001/002 001/003 001/004 001/005 001/006 001/007 002/001 002/002 002/003 002/004 002/005 003/001 004/001 004/002 004/003 004/004 004/005 004/006 004/007 004/008 004/009

				005/001 006/001 006/002 006/003 006/004 006/005 006/006 006/007
--	--	--	--	--

GET api/v3/claimservice/claimhistory/companies

Example of response

```
{
  "ResultDate": "2018-09-25T13:46:53.4430232+02:00",
  "ResultCode": 0,
  "ResultText": "OK",
  "Companies": [
    {
      "Name": "Company A ",
      "Id": "VIR000001",
      "Address1": "Address 1",
      "Address2": "",
      "PostalCode": "4000",
      "City": "Roskilde",
      "IndustryProductGroups": [
        "001/001",
        "004/001",
        "006/001"
      ]
    },
    {
      "Name": "Company B ",
      "Id": "VIR000002",
      "Address1": "Address 2",
      "Address2": "",
      "PostalCode": "2900",
      "City": "Hellerup",
      "IndustryProductGroups": [
        "001/001",
        "004/001",
        "006/001"
      ]
    }
  ]
}
```

CloseMessage				
This operation is used to notify F&P that the company is closing for ingoing and outgoing requests.				
Input	Type	Required	Length	Description
Output	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time
ResultCode	Integer	Yes		Return code
ResultText	String	Yes	255	Return text
GET api/v3/claimservice/claimhistory/closemessage				
Example of response				
<pre>{ "ResultDate": "2018-09-25T13:49:32.220004+02:00", "ResultCode": 0, "ResultText": "Company Closed successfully" }</pre>				

OpenMessage				
This operation is used to notify F&P that the company is re-opening for ingoing and outgoing requests.				
Input	Type	Required	Length	Description
Output	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time
ResultCode	Integer	Yes		Return code
ResultText	String	Yes	255	Return text
GET api/v3/claimservice/claimhistory/openmessage				
Example of response				
<pre>{ "ResultDate": "2018-09-25T13:49:32.220004+02:00", "ResultCode": 0, "ResultText": "Company Opened successfully" }</pre>				

HistoryRequest				
<p>This operation is used to submit a request to another company for claim history information. The sender must be registered for the service and the industry/product groups for which requests are made. An online company must respond to the request promptly.</p> <p>The operation returns a date/time, code, text and response to request.</p>				
Input	Type	Required	Length	Description
RequestId	String	Yes	9	VIR number for the requesting company
ResponseId	String	Yes	9	VIR number for the company responding to the request
Version	String	No	3	Indication of version. Permitted values: null or 3.0 The FP server sets the value 3.0 before call to the company.
Test	Boolean	Yes		Test marking. true = test false or null = production
RequestDate	DateTime	Yes		Date/time of request
ReferenceNumber	String	Yes	50	Request serial/reference no.
CustomerIdQualifier	String	Yes	3	Policyholder/insured-identification type. Possible values: CPR or CVR
CustomerId	String	Yes	10	Policyholder/insured CPR or CVR number. Format: CVR: nnnnnnnn CPR: ddmmyynnnn
CustomerName	String	Yes	128	Policyholder/insured name
ObjectIdQualifier	String	Yes	3	Object identification type: POL=policy number (only industry group 001 vehicle) REG=reg no. (only industry group 001 vehicle) VIN=chassis no. (only industry group 001 vehicle) ALL=combined
ObjectId	String	No	20	Object identification Omitted or null in case of identification type ALL
RequestType	Integer	Yes	1	Request type: 1=Claim history 2=Bonus only (only industry/product group 001/001) 3=Bonus and claim history (minimum 001/001)
ConsentForm	Boolean	Yes		Confirmation that consent has been given.

				True = Yes False or null = No
ConsentFormArrears	Boolean	No		Confirmation that consent has been given for arrears. True = Yes False or null = No
IndustryProductGroups	List<String>	Yes		List of industry/product groups requested for
Output	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time of response
ResultCode	Integer	Yes		Return code – see page 25
ResultText	String	Yes	255	Return text – see page 25
RequestId	String	Yes	9	VIR number for the requesting company
ResponseId	String	Yes	9	VIR number for the company responding to the request
ReferenceNumber	String	Yes	50	Request reference no.
CustomerIdQualifier	String	Yes	3	Policyholder/insured identification type. Possible values: CPR = CPR number CVR = CVR number
CustomerId	String	Yes	10	Policyholder/insured CPR or CVR number
CustomerName	String	Yes	128	Policyholder/insured name
Policies	List <Policy>	No		List of policies held or formerly held by policyholder/insured
<Policy>				
PolicyNumber	String	Yes	50	Policy number
PolicyStartDate	String	Yes	10	Policy commencement date: Format: yyyy-mm-dd
PolicyEndDate	String	No	10	Policy expiry date. Format: yyyy-mm-dd In case of active policy, the field is omitted or indicated as <i>null</i>
Arrears	Integer	Yes		Arrears. 0=No, 1=Yes, 2=Not requested
IndustryProductGroups	List<String>			List of industry/product groups covered by the policy
ProductName	String	Yes	50	Name of insurance product
ObjectIdQualifier	String	Yes	3	Object identification type (only industry group 001 vehicle): REG=registration number VIN=chassis number NA = Unknown
ObjectId	String	Yes	20	Object identification If NA, indicate "unknown"
Bonuses	List<Vehicle Bonus>	No		Bonus list (only industry group 001 vehicle)
<VehicleBonus>	Bonus			
Industry	String	Yes	7	Industry/product.

ProductGroup				Format: bbb/ppp = 001/001
RegistrationNumber	String	No	7	Registration number
VINNumber	String	No	20	Chassis number
VehicleType	String	Yes	50	Vehicle type
ClaimFreeYears	Integer	Yes		Number of years without claims
LastStepDate	String	Yes	10	Date of last step change Format: yyyy-mm-dd If the date is not known, the policy commencement date is indicated.
FixedPremium	Bool	Yes		Fixed premium. True = yes, False = no
Claims	List<Claim>	No		List of claims for policy in question
<Claim>	Claim			
IndustryProductGroup	String	Yes	7	Industry/product of claim. Format bbb/ppp.
ProductName	String	Yes	50	Name of insurance product
ObjectIdQualifier	String	Yes	3	Object identification type (only industry group 001 vehicle): REG=registration number VIN=chassis number NA = Unknown
ObjectId	String	Yes	20	Object identification In NA, indicate "unknown"
ClaimLevels	List<String>	No		List of coverage levels (must be indicated for industry group 001 vehicle, 002 private, 003 pleasure craft and 005 accident)). See section on HistoryRequest – Coverage levels for possible values
ClaimDate	String	Yes	10	Claim date Format: yyyy-mm-dd
ClaimType	String	No	70	Claim type
ClaimStatus	Integer	Yes		Status. 0=Open, 1=Closed
ClaimPaid	Integer	No		Sum paid indicated in DKK 1/100 ("øre") -10 = Not stated
ClaimReserve	Integer	No		Reserve sum indicated in DKK 1/100 ("øre") -10 = Not stated
ClaimImpact	Bool	No		Impact (only industry group 001 vehicle) True = yes False = no
<p>POST api/v3/claimservice/claimhistory/historyrequest</p> <p>BODY</p> <pre>{ "RequestId": "VIR000001", "ResponseId": "VIR000000", "Version": "3.0", "Test": false, "RequestDate": "2018-09-25T12:35:51", "ReferenceNumber": "0123456789A",</pre>				

```

"CustomerIdQualifier": "CVR",
"CustomerId": "11111114",
"CustomerName": "Anders And",
"ObjectIdQualifier": "ALL",
"ObjectId": null,
"RequestType": 3,
"ConsentForm": true,
"ConsentFormArrears": false,
"IndustryProductGroups": [
  "001/001",
  "004/001",
  "006/001"
]
}

```

Example of response

```

{
  "ResultDate": "2018-09-25T15:35:58.1588783+02:00",
  "ResultCode": 0,
  "ResultText": "OK",
  "RequestId": "VIR000001",
  "ResponseId": "VIR000000",
  "ReferenceNumber": "0123456789A",
  "CustomerIdQualifier": "CVR",
  "CustomerId": "11111114",
  "CustomerName": "Hansen A/S",
  "Policies": [
    {
      "PolicyNumber": "policy 0",
      "PolicyStartDate": "2018-09-25",
      "PolicyEndDate": null,
      "Arrears": false,
      "ProductName": "Car insurance",
      "ObjectIdQualifier": "REG",
      "ObjectId": "CW12345",
      "IndustryProductGroups": [
        "001/001",
        "004/001",
        "006/001"
      ],
      "Bonuses": [
        {
          "IndustryProductGroup": "001/001",
          "RegistrationNumber": "CW12345",
          "VINNumber": "WBS66512436",
          "VehicleType": "Car",
          "ClaimFreeYears": 6,
          "LastStepDate": "2018-09-25",
          "FixedPremium": true,
        },
        {
          "IndustryProductGroup": "001/001",
          "RegistrationNumber": "ZP12345",
          "VINNumber": "AKB66512436",
          "VehicleType": "Car",
          "ClaimFreeYears": 1,
          "LastStepDate": "2018-09-25",
          "FixedPremium": true,
        }
      ],
      "Claims": [

```

```
{
  "IndustryProductGroup": "001/001",
  "ProductName": "Car insurance",
  "ObjectIdQualifier": "REG",
  "ObjectId": "CW12345",
  "ClaimLevels": [
    "001-1",
    "001-2"
  ],
  "ClaimDate": "2018-09-25",
  "ClaimType": "Road traffic accident",
  "ClaimStatus": 1,
  "ClaimPayed": 245000,
  "ClaimReserved": 0,
  "ClaimImpact": true
},
{
  "IndustryProductGroup": "004/001",
  "ProductName": "AgricultureBuilding insurance",
  "ClaimDate": "2018-09-25",
  "ClaimType": "Comprehensive",
  "ClaimStatus": 1,
  "ClaimPayed": 1212000,
  "ClaimReserved": 0,
  "ClaimImpact": null
}
]
}
```

HistoryRequest – Coverage levels

Code	Coverage level
001 Vehicles regardless of industry/product group	
001-1	Liability
001-2	Comprehensive
001-3	Glass
001-4	Roadside assistance
001-5	Driver
001-6	Miscellaneous
002/001 Private building regardless of industry/product group	
002/001-1	Liability
002/001-2	Fire – including electrical damage
002/001-3	Water damage (rising, breakages to visible internal pipes)
002/001-4	Weather damage (cloudburst, storm, drift damage, thaw)
002/001-5	Burglary/Theft/Vandalism
002/001-6	Sudden damage (collision)
002/001-7	Other comprehensive building – glass/basin/sanitation, animals, renovation and extension
002/001-8	Hidden pipes/cables
002/001-9	Service connection
002/001-10	Fungi/Rot/Insects
002/001-11	Miscellaneous – including legal aid, psychological crisis counselling, etc. (= all damage which is not building related)
002/002 Private contents regardless of industry/product group	
002/002-1	Liability
002/002-2	Fire
002/002-3	Water damage (rising, discharge)
002/002-4	Weather damage (cloudburst, storm, drift damage, thaw damage)
002/002-5	Electrical damage/electronics damage
002/002-6	Theft/Robbery/Plunder/Vandalism
002/002-7	Burglary
002/002-8	Sudden damage
002/002-9	Other contents damage (lost baggage on return from travel, refrigerator/freezer damage, traffic damage, glass/basin/sanitation)
002/002-10	Miscellaneous – including legal aid, psychological crisis counselling, identity theft, etc. (= all damage which is not object related)
002/003 Private liability regardless of industry/product group	
002/003-1	Liability
002/003-2	Miscellaneous
002/004 - Private travel insurance regardless of industry/product group	
002/004-1	Illness/injury
002/004-2	Cancellation claims
002/004-3	Baggage delay – during travel
002/004-4	Other travel claims
002/005 - Private animal insurance regardless of industry/product group	
002/005-1	Liability
002/005-2	Life
002/005-3	Illness/treatment
002/005-4	Miscellaneous
003 - Pleasure craft regardless of industry/product group	

003-1	Liability
003-2	Comprehensive
003-3	Miscellaneous
005 - Accident regardless of industry/product group	
005-1	Permanent injury
005-2	Illness
005-3	Dental injury
005-4	Treatment costs
005-5	Critical illness
005-6	Immediate compensation
005-7	Miscellaneous

Server checks

The FP server checks the contents of a request before it is forwarded to the responding company. The server rejects the request if all checks are not complied with. The following is checked when receiving a request:

- Client ID and secret must be authorized.
- The request is checked for correct data types
- Test bit (in production) must not be set unless submitting to a robot company (VIR000000) or own company
- Requesting company (RequestId) must be associated with a client ID
- Requesting company (RequestId) must be known on the server and registered for the claim history service
- Responding company (ResponseId) must be known on the server and registered for the claim history service
- RequestId and ResponseId must not be identical unless the test bit is set to true
- If CustomerIdQualifier is CVR, CustomerId must be a valid CVR number (modulus 11 check)
- If CustomerIdQualifier is CPR, CustomerId must be a valid CPR number (modulus 11 check)
- ConsentForm must be true
- IndustryProductGroups must include one of the following values: 001/001-001/007, 002/001-002/005, 003/001, 004/001-004/009, 005/001, 006/001-006/007
- Requesting company (RequestId) must be registered for the industry/product groups for which requests are made.
- Responding company must be registered for the industries for which requests are made
- ReferenceNumber must not have been used before
- Sender and recipient must both be open
- Policyholder/insured identification type must be CPR or CVR
- Object identification type must be POL, REG, VIN or ALL
- If object identification type is ALL, object identification must not be completed
- If object identification type is POL, REG or VIN, object identification must be completed
- Object identification types POL, REG and VIN must only be used for industry 001
- Request type must be 1, 2 or 3
- Request type 2 (bonus only) must only be used for requests for industry/product group vehicle (001/001)
- Request type 3 (bonus and claim history) must only be used for requests for at least industry/product group type vehicle (001/001)
- Version must be 3.0

Return codes and text

The following describes the values of HTTP StatusCode, ResultCode and ResultText which are returned for all API calls to the FP server.

HTTP status codes		API return codes	
Code	Message	Code	Text
The following return codes come from the FP server			
200	OK	0	OK
400	Bad request	1	Requesting Company is closed
400	Bad request	2	Responding Company is closed
400	Bad request	11	Responding Company is unknown
504	Gateway timeout	12	No answer from responding Company
400	Bad request	22	Requesting and responding company cannot be the same unless test bit has been set
400	Bad request	23	ConsentForm must be true
400	Bad request	24	Customer id is not valid CPR/CVR-number
400	Bad request	25	Invalid industry/product group(s)
400	Bad request	26	Requesting company does not have selected industry/product groups
400	Bad request	27	ReferenceNumber is not unique
400	Bad request	28	Test bit not allowed
400	Bad request	29	Invalid CustomerIdQualifier
400	Bad request	30	Invalid ObjectIdQualifier
400	Bad request	31	ObjectId must be omitted when IdQua is ALL
400	Bad request	32	ObjectId is mandatory
400	Bad request	33	Invalid RequestType
400	Bad request	34	Invalid RequestType for selected IndustryProductGroups
400	Bad request	35	Responding company does not have selected industry groups
400	Bad request	36	Invalid Request datatype
400	Bad request	37	Invalid version
400	Bad request	252	ReferenceNumber is mandatory
400	Bad request	253	Invalid requestId
401	Unauthorized		
502	Bad Gateway	251	Error calling responding Company
503	Service Unavailable	254	Service currently unavailable
500	Internal System Error	255	System error from FP server
The following return codes are forwarded from the receiving company			
400	Bad request	501	Unknown customer
400	Bad request	502	Unknown object/policy
400	Bad request	504	Invalid Authorization to responding Company
400	Bad request	555	System error from Company

5. Company API

The following section describes the REST API that companies need to implement in their own system.

Preconditions

A participating company must create a REST API which complies with the structure which F&P has defined and described below.

As a starting point, the company must implement an OAuth2 service and flow as described in section 3. However, the company can decide whether or not to support Refresh tokens. This is not a requirement according to the OAuth2 standard.

The FP server offers limited support for authenticating calls to the companies' APIs via JWT tokens (<https://tools.ietf.org/html/rfc7519>), described in the next section (JWT tokens).

Companies wishing to implement OAuth2 must submit the following information to F&P:

- The address of the company's API (`EndPoint`)
- Token endpoint for token retrieval
- Client ID which identifies F&P (`Client_Id`)
- IP address(es) from which the company calls F&P

Companies wishing to implement JWT tokens must submit the following information to F&P:

- The address of the company's API (`EndPoint`)
- JWT encryption algorithm (HS256, HS384, or HS512)
- Username which identifies F&P to the company (`Username`)
- IP address(es) from which the company calls F&P

Secrets/keys are managed by the API administrator via the EDI Web user interface – see separate guidelines.

The company must also notify the EDI office of the industry/product groups available for the company. Requests can only be made for industry/product groups for which the company is registered.

JWT tokens

The FP server supports the issuing of JWT tokens with the following specifications:

Security: JWT with JSON Web Signature (JWS) (<https://tools.ietf.org/html/rfc7515>) and support of the following symmetric encryption algorithms: HS256, HS384, or HS512.

JWS format: [Base64-URL encoded header].[Base64-URL encoded payload].[Signature]

JWS (JOSE) header example: `{"alg":"HS256","typ":"JWT"}`

Operations

The company's API must implement the 2 operations described in the following section.

Status				
This operation is used to test authorization. It returns a date/time, code and text.				
Input				
Parameter	Type	Required	Length	Description
Output				
Parameter	Type	Required	Length	Description
ResultDate	DateTime	Yes		Date/time
ResultCode	Integer	Yes		Return code
ResultText	String	Yes	255	Return text
GET <company endpoint>/status				
Example of response				
<pre>{ "ResultDate": "2018-09-25T13:44:59.5565937+02:00", "ResultCode": 0, "ResultText": "OK" }</pre>				
HistoryRequest				
This operation is used to receive a claim history request from a company. It is called by the FP server on behalf of another company. Input and output are identical to the HistoryRequest operation, described in section 4.				

Return codes and text

The following describes the values of HTTP StatusCode, ResultCode and ResultText which the company must return for all API calls.

HTTP status codes		API return codes	
code	message	Result Code	Result Text
200	OK	0	OK
400	Bad request	501	Unknown customer
400	Bad request	502	Unknown object/policy
401	Unauthorized		
500	Internal System Error	555	System error from Company

REST API with Oauth2 template

F&P has created a REST API sample project with source code in C# which illustrates how the different operations are implemented.

This project can be requested from the EDI office.

6. Test

Testing a request can be accomplished by sending the request to a robot company on the FP server which generates an automated response. The robot company's VIR number is `VIR000000`.

The name of the robot company is "Test claim company (auto response)" – but it will sometimes respond with "Unknown customer" and otherwise with "History".

By using the CVR numbers in CustomerId indicated below, the robot company can respond with the following optional responses:

CustomerId	Result
<code>00000nnn</code>	The relevant return code is returned. E.g. <code>00000002 = 2 - Responding Company is Closed</code> <code>00000502 = 502 - Unknown object/policy</code>
<incorrect CPR/CVR number>	24 – CustomerId is not a valid CPR/CVR-number
<correct CPR/CVR number>	0 - Random response every time Number of policies: 0-4 Number of claims: 0-10

Testing a receipt of request can be accomplished by the company sending the request to itself.

For all tests, the test marking must be set to `true`.

Technical test report

The test report tool is an automated service which provides an overview of the requirements which must be completed and tested before the company can start production.

The various tests are divided into sections and each section has its own status. The sections are:

- Signing up

- Exchange
- Calls to EDI API
- Calls to Company API
- Number of requests
- Number of responses

The test report tool retrieves a token and calls the company's status operation. The remainder of the tests must be performed by the company and a report summarizes the test status (number and latest event date). The test report is available to API Administrators.

The guidelines for creating API administrators and a description of the test report tool can be requested from the EDI office, edi@forsikringogpension.dk