**Corona BorgerStatus (CBS) API specification (draft v2), April 8 2021**

**Introduction**

This document provides an overview of the CBS API draft. The documentation consists of three parts:

* WSDL specifying the service available to the app backend
* A mock WSDL specifying a mock service, which can be useful for sandbox experiments
* XSD specifying the contents of messages sent to NAS (“National AdviseringsService”)

Also, XML examples of API request/responses are provided in this document.

The API is draft, and may be subject to change. Along with the documentation resources, online resources are also referenced, which provide possibilities for sandbox experiments.

**Deliverable Overview**

* **CBS\_2021-04-08.pdf**: This document
* **cbs-idws.wsdl**: WSDL specifying the service available to the app backend
* **cbs-mock.wsdl**: A mock WSDL specifying a mock service, which can be useful for sandbox experiments
* **cbs2nas.xsd**: XML Schema for CBS job notifications to NAS

**WSDL Specification**

The mock WSDL does not support IDWS headers and IDWSFault message, which is the case for the real IDWS WSDL. The mock WSDL is available in **cbs-mock.wsdl**. The IDWS WSDL is available in **cbs-idws.wsdl.**

A mock/stub implementation of the draft WSDL is available as a docker image on DockerHub:

<https://hub.docker.com/repository/docker/trifork/coronapass-mock>

The main flow is to make a CPR-based lookup for the citizen Corona Pass, which will return either a JobId indicating that a job has been scheduled for collecting the Corona Pass information, or the actual Corona Pass consisting of a list of signed JWTs (inspect contents on eg. jwt.io) along with a type, representing different versions/editions of the Corona Pass. Also, the WSDL allows for key/value-pairs for each JWT in case this may prove useful for future extensions, eg. to provide additional information about a specific JWT.

The stub implementation is available in a public github repository:

<https://github.com/trifork/coronapass-mock>

Out of the box, the service returns actual answers on even minutes and FutureResult/JobId on odd minutes. If a specific behaviour is desired, an HTTP header can be provided: -Xcachehit=true or -Xcachehit=false.

Running the mock service can be done using

docker run -p 8080:8080 --name coronapass-mock trifork/coronapass-mock: 0.0.1-SNAPSHOT-bed8b5f

After startup the service WSDL will be available at http://localhost:8080/ws/coronapass.wsdl

Sample request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:cor="http://sundhedsdatastyrelsen.dk/coronapass/2021/03/coronapass-web-service">

 <soapenv:Header/>

 <soapenv:Body>

 <cor:GetCoronaPassRequest>

 <cor:Identifier>0101010101</cor:Identifier>

 </cor:GetCoronaPassRequest>

 </soapenv:Body>

</soapenv:Envelope>

Sample response (cache miss):

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">

 <SOAP-ENV:Header/>

 <SOAP-ENV:Body>

 <ns2:GetCoronaPassResponse xmlns:ns2="http://sundhedsdatastyrelsen.dk/coronapass/2021/03/coronapass-web-service">

 <ns2:FutureResult>

 <ns2:JobId>e5fa7874-2a3a-4e44-ae34-24b287b39e93</ns2:JobId>

 </ns2:FutureResult>

 </ns2:GetCoronaPassResponse>

 </SOAP-ENV:Body>

</SOAP-ENV:Envelope>

Sample response (cache hit):

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">

 <SOAP-ENV:Header/>

 <SOAP-ENV:Body>

 <ns2:GetCoronaPassResponse xmlns:ns2="http://sundhedsdatastyrelsen.dk/coronapass/2021/03/coronapass-web-service">

 <ns2:CoronaPasses>

 <ns2:Pass>

 <ns2:Type>DK</ns2:Type>

 <ns2:JWT> eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2MTc3MTg3MTYsImlhdCI6MTYxNjQwMzY1NywianRpIjoiOWQwMjhmMjktMjU0Ni00ZDU5LThjN2EtOGJjOWI1MmEzNmU4In0.c-FZaQWVoosnJA18uoOItgX10v9XHDHkAyNgfDgdipQ</ns2:JWT>

 </ns2:Pass>

 <ns2:Pass>

 <ns2:Type>DKPLUS</ns2:Type>

 <ns2:JWT> eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2MTc3MTg4MTEsImlhdCI6MTYxNjQwMzY1NywianRpIjoiOWQwMjhmMjktMjU0Ni00ZDU5LThjN2EtOGJjOWI1MmEzNmU4IiwibGVnYWxOYW1lIjoiSmFudXMgTmllbHNlbiIsImRhdGVPZkJpcnRoIjoiMTk4NS0xMi0wMyJ9.YQpCr3igO61wBjcrSIsIXGp\_5FBTV0KQUTw-ZrZDEOg</ns2:JWT>

 </ns2:Pass>

 <ns2:Pass>

 <ns2:Type>EU</ns2:Type>

 <ns2:JWT>6BFC$B5Z7+J2M433FH\*:R71FZ7F+IMVC3L6Q6WCPCGU+CLGE86U9.PK0E5QV2UF$XCL66N93/.DMNK4FO/34\*LGTP2K+Q/RO602W5P%H9H%RM5A8I1$CAROQC1252AQIS\*ZUB17+K74VMWZS8/E8V2CY2M2HO8J:0H1XI$E5KCOCC07NB9EB$KC0:2W-R$\*2RYRQ6PW.VYKVUNC4QK4\*U7+QZ3KFTOK56LSLQV4JNB0ML7RINLGBPK7U7T.4PNI%CBEY3ECF7W37XK.73JA9IQGKOKK62 9VPG0\*CIFGBLK5P\*I$DNCE4PN8LFNO769AHBILWRSPA644AYKUOOL$MRIFCUR0+.2%9J$/2VHS1ML325 80E:2Z:C1 TI8PP$P84E3-3CNDD2C/4ED D.3NL9BW7P.BQQZFE8QI 1Q9G/ECHW8NS4VN6.%7JMN:Y8JLUY9DEF90319JF4A8.TS+4NSDMHD1:-JW 02DA5\*P4$OWMAHLCJ2Q266JA0TNBC0QWOBO$TO8QM%A5P7QVPN3HUBAHJF0KA EUPH2N-JEM753B4KRS3CU.7FYNBUCVND\*9E7VJS:VZBQH2GWTDV2JQHGZ7O6:36F6+$RU/UEUR76IY17XZN706YCFS FMCSO04R38C4OA$VI0C470/J9H1

</ns2:JWT>

 <ns2:Properties>

 <ns2:Key>eu:keytype</ns2:Key>

 <ns2:Value>digital\_green\_certificate</ns2:Value>

 </ns2:Properties>

 </ns2:Pass>

 </ns2:CoronaPasses>

 </ns2:GetCoronaPassResponse>

 </SOAP-ENV:Body>

</SOAP-ENV:Envelope>

**Token types**

**DK**

The DK token consist of a JWT where the body follows the example outlined below:

{ “exp”: 1617573600, “iat”: 1616403657, “jti”: “9d028f29-2546-4d59-8c7a-8bc9b52a36e8", }

**DKPLUS**

The DKPLUS token consist of a JWT where the body follows the example outlined below:

{ “exp”: 1617573600, “iat”: 1616403657, “jti”: “9d028f29-2546-4d59-8c7a-8bc9b52a36e8”, “legalName”: “Janus Nielsen”, “dateOfBirth”: “1985-12-03” }

**EU**

The EU token consists of a CWT (https://tools.ietf.org/html/rfc8392) as specified on <https://github.com/ehn-digital-green-development/hcert-spec> (Version 0.1.4, 2021-04-06 commit SHA 7ab316fee77d910677a0542a1dceb2e7690a395d) and is encoded as depicted on the following diagram:

An example of such token is listed below in its Base45 encoded form:

6BFC$B5Z7+J2M433FH\*:R71FZ7F+IMVC3L6Q6WCPCGU+CLGE86U9.PK0E5QV2UF$XCL66N93/.DMNK4FO/34\*LGTP2K+Q/RO602W5P%H9H%RM5A8I1$CAROQC1252AQIS\*ZUB17+K74VMWZS8/E8V2CY2M2HO8J:0H1XI$E5KCOCC07NB9EB$KC0:2W-R$\*2RYRQ6PW.VYKVUNC4QK4\*U7+QZ3KFTOK56LSLQV4JNB0ML7RINLGBPK7U7T.4PNI%CBEY3ECF7W37XK.73JA9IQGKOKK62 9VPG0\*CIFGBLK5P\*I$DNCE4PN8LFNO769AHBILWRSPA644AYKUOOL$MRIFCUR0+.2%9J$/2VHS1ML325 80E:2Z:C1 TI8PP$P84E3-3CNDD2C/4ED D.3NL9BW7P.BQQZFE8QI 1Q9G/ECHW8NS4VN6.%7JMN:Y8JLUY9DEF90319JF4A8.TS+4NSDMHD1:-JW 02DA5\*P4$OWMAHLCJ2Q266JA0TNBC0QWOBO$TO8QM%A5P7QVPN3HUBAHJF0KA EUPH2N-JEM753B4KRS3CU.7FYNBUCVND\*9E7VJS:VZBQH2GWTDV2JQHGZ7O6:36F6+$RU/UEUR76IY17XZN706YCFS FMCSO04R38C4OA$VI0C470/J9H1

The Json payload of that token is listed below:

{ "hcert": { "sub": { "n": "Gaby Doe", "id": [ { "t": "nid", "i": "0102030405" } ] }, "vac": [ { "des": "J07BX03", "nam": "COMIRNATY", "aut": "Pfizer BioNTech", "seq": 1, "tot": 2, "dat": "2021-02-20", "adm": "Region Halland" } ], "tst": [ { "dis": "Covid-19", "typ": "LP6464-4", "tna": "Nucleic acid amplification with probe detection", "tma": "BIOSYNEX SWISS SA BIOSYNEX COVID-19 Ag BSS", "ori": "258500001", "dat": "2021-02-20", "res": false, "fac": "Region Midtjylland", "cou": "DNK" } ] }, "iss": "DNK", "iat": 1617876226, "exp": 1617876226 }

**Customizing return values**

An admin SOAP service is available, which allows for customized return values. The admin WSDL is available at http://localhost:8080/ws/admin.wsdl

The admin service simply allows for setting the returned JWT for a specific JWT type.

Sample request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:adm="http://sundhedsdatastyrelsen.dk/admin/admin">

 <soapenv:Header/>

 <soapenv:Body>

 <adm:AdminRequest>

 <adm:identifier>

 <adm:Type>DKPLUS</adm:Type>

 <adm:JWT>eyJraWQiOiIxYTA4MmY4ZS02ZGRhLTQ3ZGQtYWMxYy1hYTJhMTM2ZjY1N2QiLCJhbGciOiJFUzI1NiJ9.eyJuYW1lIjoiTCoqKiBLKioqKioqIiwiZG9jdW1lbnRJZCI6IjEwMDAwMTgiLCJ2YWNjaW5lRm9yIjoiU0FSUy1Db1YtMiAoQ09WSUQtMTkpIiwiZXhwIjoxNjE3NTczNjAwLCJpYXQiOjE2MTY0MDM2NTYsInByb2R1Y3ROYW1lIjoiTW9kZXJuYSBDb3ZpZC0xOSB2YWNjaW5lIiwianRpIjoiOGU3NzU3NzktOGZhNi00N2UwLWJmNWUtZmY1YWU5MDdiMDM4Iiwic3NuIjoiMTEwMjAxKioqKiJ9.Q1BNUT4TCKunqsyX\_cdTFxjL1KM5fSjbnYnahvqUKgm2ErnJPif6BzukO2xuTLyNSVrYoEB6SZluo1yC8kU2XA</adm:JWT>

 </adm:identifier>

 </adm:AdminRequest>

 </soapenv:Body>

</soapenv:Envelope>

**NAS XSD Specification**

Notifications from CBS to the app-backend regarding job statuses are provided as "NotifyContent" payloads in NAS messages. For general information regarding NAS notifications, consult the official documentation on the NSP documentation web site:

[https://www.nspop.dk/display/public/web/NAS+-+Kom+Godt+i+Gang](https://www.nspop.dk/display/public/web/NAS%2B-%2BKom%2BGodt%2Bi%2BGang)

The actual payload provided in the NotifyContent section consists of the JobId along with the status of the job. If data collection was not possible (JobStatus=FAILED), an error code will be provided (FailedReasonCode). Error codes are not provided as part of the XSD to allow for future extensions. Some error codes will indicate logical errors, eg. non-existing CPR, while others may be due to technical reasons, eg. if backends for vaccine and/or testresult data are unavailable. These reason codes will follow, and it will be possible to distinguish between logical and technical errors.

The structure of NAS notifications is specified in **cbs2nas.xsd**, and an example of the NotifyContent is displayed below:

<ns2:JobStatusNotification xmlns:ns2="http://sundhedsdatastyrelsen.dk/coronapass/2021/03/coronapass-web-service>

 <ns2:JobId>d26d5429-a9b8-41df-97bb-c960d8d61430</ns2:JobId>

 <ns2:JobStatus>FAILED</ns2:JobStatus>

 <ns2:FailedReasonCode>002</ns2:FailedReasonCode>

</ns2:JobStatusNotification>