WSAPI

System Dokumentation for Sundk´s Web-Service 2.0 løsning

rkkp

regionernes kliniske kvalitetsudviklingsprogram

Udgiver:
Regionernes Kliniske Kvalitetsudviklingsprogram
Ryesgade 53B, 3. etage
2100 København Ø

www.rkkp.dk

Version 1.0
Versionsdato: 25.09.2023

WSAPI Dokumentation

WSAPI Generelt

Introduktion

Formålet med dette dokument er at videreformidle de oplysninger, som det forventes er nødvendige for at komme i gang med WSAPI som er den nye webservice løsning i sundk.

WSAPI anvender en ny teknologi stack og der skal forventes, at der skal laves systemændringer hos nuværende dataleverandører, hvis de benytter sig af den originale webservice.

Dette dokument er under løbende udvikling, og feedback på processen modtages gerne på rkkp.inddata@rm.dk.

WSAPI-resumé

Det nyudviklede WSAPI-system er en REST-ful API, med brugerkontrol gennem JSON-web-tokens (JWT). WSAPI vil over tiden erstatte vores originale webservice-stack, som er baseret på en SOAP-protokol og som arbejder med WSDL/XML. I WSAPI skal Indberetninger sendes i JSON format, og bliver valideret mod et prædefineret skema. Skema strukturen forsøges at standartiseres med en overordnet datamodel.

Der blev især fokuseret på, at øge performance og brugeranvendeligheden. Desuden er der udviklet nye features, bl.a.:

- Flere versioner af database endpoints aktive på samme tid.
- Delvis opdatering af indberetninger (PATCH method).
- Ekstra sikkerhed med JWT authentication.
- Nye dokumentationsværktøjer: Swagger UI / Redocs / Web dokumentation (ex. DAH)
- Brugere kan finde og søge efter egne indberetninger.
- Brugere kan ved kritiske fejl slette deres egne indberetninger.
- Brugere kan se alle succesfulde transaktioner i endpointet /Dailytransaction/
- Brugere kan se alle deres fejl i op til en måned på /ErrorLogs/

Og langt mere

Udvikling

Værktøjer

Swagger UI og Redocs
Swagger afspejler alle skemaer som er på serveren. Swagger danner JSON-eksempler og er en god ressource til, at se hvordan de forskellige endpoints skal interageres med og man kan også teste det i browseren.

Redocs er en alternativ til swagger, hvor man gå i dybden med bl.a. valideringen af endpoints, og se hvilke krav der er til de individuelle JSON-elementer.

Web dokumentation
Dette er et alternativt værktøj for at se dataspecifikationerne, dette værktøj er målrettet til klinikkere, projekt ledere og dem som skal mappe data.

Test-miljøet

For at få adgang til test miljøet, skal man sende en e-mail til rkkp.inddata@rm.dk med oplysninger omkring den ansvarlige kontaktperson:

- Fuldt navn
- E-mail,
- Organisation
- Database navn(e).

RKKP vil derefter oprette jeres bruger og sende jer de nødvendige informationer om relevante endpoints og kodeord.

Test-miljøet er ikke på SDN, og er ikke et sikkert miljø at sende personfølsomme oplysninger til. Der skal derfor benyttes genereret testdata under udvikling. Da CPR-nummer bliver valideret som format kan det anbefales at bruge et officielt test CPR-nummer som f.eks. ”2311143995”.

Prod-miljøet

For at få adgang til produktionsmiljøet påkræves det at der oprettes en service aftale med sundhedsdatanettet. Service aftale nummeret er #3539 TCS-SDN-Services

Der er følgende oplysninger for SDN på produktion:
SDN IP: 77.243.49.187
SDN DNS-navn: https://wsapi-prod.sundk.dk
Port/Protokol: 443 / HTTPS

Verifikationsprocessen

For at kunne gå over til produktionmiljøet, skal dataleverandøreren udføre en verifikationsproces. Verifikationsprocessen har til formål at hjælpe dataleverandører med at komme rundt om alle funktionerne i WSAPI, således at de er klar over hvilke muligheder som står til rådighed til dem, samt at systemet anvendes optimalt. Det nuværende verifikationsdokument er vedhæftet e-mailen.

Endpoint Opbygning

Alle endpoints følger navnestandarden ***/Databasenavn/DatabaseDomain/DatabaseVersion/***

Desuden har alle endpoints de samme funktioner tilgængeligt:

Request Method	**Endpoint**	**Formål**
POST	*/*	Indsend indberetning i JSON format
POST	*/Batch/*	Indsend et batch a flere indberetninger
GET	*/*	Få en liste af data_entry_ids i en given periode
GET	*/entry_id*	Udtræk af en specifikt data_entry_id
PUT	*/entry_id*	Opdater en indberetning (komplet) i WSAPI ved hjælp af en komplet JSON struktur
PATCH	*/entry_id*	Opdater en indberetning (delvist) i WSAPI ved hjælp af en delvis JSON struktur
DEL	*/entry_id*	Slet en indberetning
GET	*/SanityCheck/*	Få en row count i et specifik dato interval
GET	*/DailyTransactions/*	Få en liste over de succesfulde transaktioner en given dag
GET	*/Schema/*	Få det bagliggende POST validerings skema tilbage
GET	*/ErrorLogs/*	Få en liste af fejl tilbage

JSON-web-token (JWT)

Systemet bruger JSON-web-tokens til brugerstyring. Der findes 2 forskellige tokens:

- Access-token (gyldig i 3 minutter)
- Refresh-token (gyldig i 30 minutter)

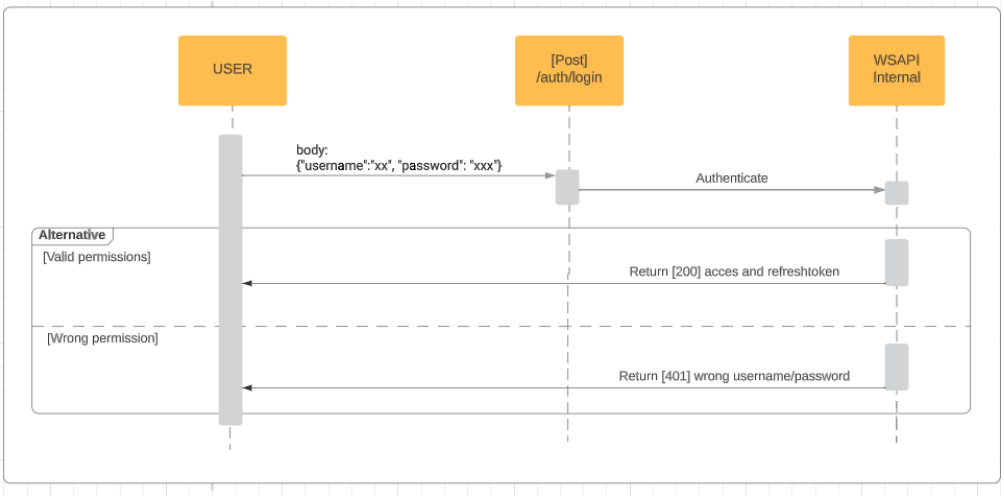
Alle interaktioner med et database endpoint påkræver en gyldig JWT, som sendes med i headeren i http requesten. Det kan se således ud i curl, ved f.eks. afsendelse af en indberetning:

```
curl -X 'POST' \
  'https://rkkp-wsapi.test____' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer Access-token \
  -H 'Content-Type: application/json' \
  -d '{ "Report": ____ }
```

RKKP har valgt at anvende ”stateless authentication”. Det vil sige at efter omkring 150 sekunder skal der laves et API kald til auth/refresh for at anmode om en ny access- og refresh-token. Ved denne implementering kan systemet permanent lave indberetninger. Vær opmærksom på at refresh-token er valid i 30 minutter.

Access-token flow

Herunder kan flowet for anmodning om en access-token ses.



For at anmode om en access-token skal systemet sende en POST-request til WSAPI's auth endpoint: /auth/login. I requesten tilføjes en JSON-struktur med email, og password. Hvis disse er korrekte vil WSAPI returnere en access- og refresh-token. De efterfølgende kodeeksempler er skrevet i Python og Groovy, og bør give en god idé om hvordan det kan se ud at anmode om en access-token:

Python eksempel

```
import json, requests

url = https://________/auth/login

json_data = {"username": "xxx@example.com", "password": "xxx"}
server = requests.post(url, headers = { "Content-Type": "application/json"},
           json=json_data)

output = json.loads(server.text)
access_token = output["data"]["access_token"]
refresh_token = output["data"]["refresh_token"]
```

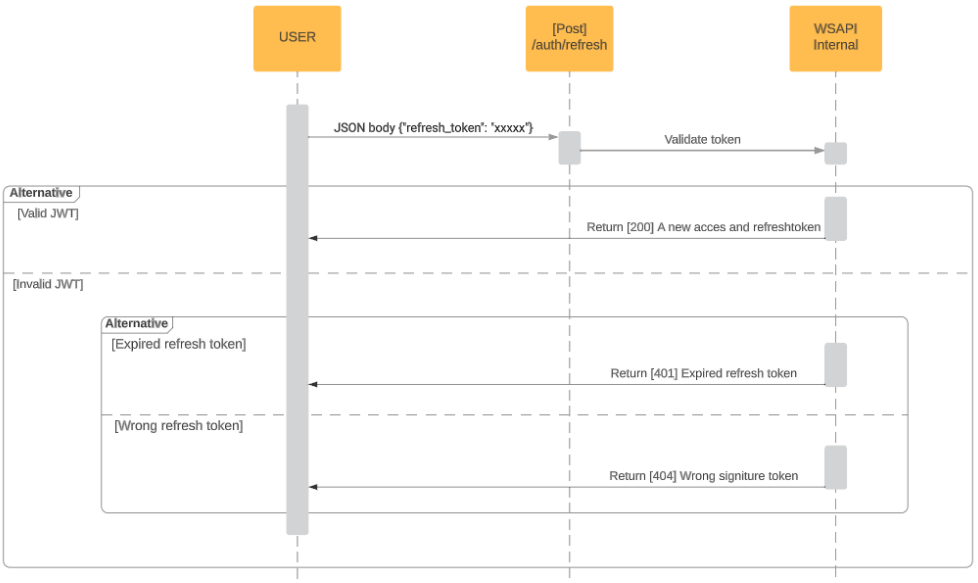
Groovy

```
RestBuilder restBuilder = new RestBuilder(connectTimetout: 2000, readTimeout: 2000)
```

```
restBuilder.restTemplate.setMessageConverters([new
StringHttpMessageConverter(Charset.defaultChatset.forName("UTF-8"))])

RestResponse response = restBuilder.post(https://_____/auth/login) {

            contentType("application/json")

            accept("application/json")

            json "{\"username\": \"userAccount\", \"password\": \"Password\"}"

}
```

Refresh-token flow

Herunder kan flowet for at anmode om en ny access- og refresh-token ved hjælp af en aktiv refresh-token.



For at anmode om nye tokens skal systemet sende en POST-request til WSAPI's auth endpoint: /auth/refresh. I requesten tilføjes en JSON-struktur med refresh_token. Hvis denne stadig er aktiv vil WSAPI returnere en access- og refresh-token.

Brug af WSAPI

Afsendelse af Data / Indberetning af data

Som beskrevet i JWT sektionen, skal der vedhæftes en JWT access-token i alle interaktioner med database endpoints.

Et kode eksempel på hvordan man kan interagere med endpoint for DAH databasen med en POST-request i Python:

```
endpoint = https://________/DAH/FoersteKontakt/1.0.0/


server = requests.post(endpoint, headers = { "Authorization": "bearer ACCESS-TOKEN", json=YOUR_DATA)

output = json.loads(server.text)
```

Data_Entry_ID (Indberetnings ID)

I det nye system har hver indberetning et ”data_entry_id”. Det er et auto genereret id til identifikation af indberetninger. Med denne ID kan i selv opdatere, slette eller delvis opdatere jeres eksisterende indberetninger. Serveren vil returnere dette ”data_entry_id” ved POST kald.

I kan til enhver tid følge med i alle af jeres transaktioner ved brug af endpointen /logs/. Der kan i søge frem på transaktioner på specifikke endpoints, på specifikke statuskoder, på request metoder, datoer etc.

Server svar

Alle svar er blevet standardiseret, og vil altid komme i det samme format:

```
"timestamp": "2023-01-01 12:00:00.000000",

"status_code": xxx,

"errors ": {”error_1”:”___”}

"data": {"Køn": "Kvinde"}
```

Status koder

Serveren kan returnere en flere statuskoder, de mest hyppige er:

200	Succesfuldt kald, og alt er gennemført som forventet.
401	JWT er udløbet
403	Adgang nægtet
404	Ressourcen findes ikke
406/409	Validerings fejl i data definitionen

REST aktioner

Vi understøtter alle standard operationer [POST, PUT, PATCH, DELETE, GET]

POST	Indsende selve indberetningen
GET	Serveren returnerer den anmodede indberetning
PUT	Opdatere all information i en indberetning. Alt data skal sendes én gang til.
PATCH	Del opdateringer på en eksisterende indberetning
DELETE	Slette en eksisterende indberetning. **Vigtigt:** dette skal kun fortages i meget særlige situationer hvor der tales om alvorlig fejl i forhold til patient oplysninger som CPR. Sletning kan kun udføres af de brugere/organisationer, som oprindeligt har indleveret indberetningen.

Formatering: JSON-format

JSON er case-sensitiv, derfor har vi valgt at alle (enums / udfald), bliver skrevet med småt.
For nogle systemer kan det desuden være relevant at sætte encoding til **utf-8**.