UTDL - API documentation:

Autentisering:

Denne API-en (**A**pplication **P**rogramming **I**nterface) bruker tokens og bruker-id for å autentisere brukere. Du får token og bruker-id når du logger inn, og disse må lagres i local storage. Pass på å sende token og bruker-id med alle andre requests til server som. Disse legges i request sin header som «Auth» og «userId».

Heroku server url:

https://to-do-list-gr1.herokuapp.com

Databaserad-eksempel:

id	title	content	owner	activated	idcount	public	contributors
1	Handleliste	{"listElements":[{"title":"Melk","duedate":"","tag":"Hjem","id":0}, {"title":"Brød","duedate":"","tag":"Hjem","id":1}, {"title":"Sukker","duedate":"","tag":"Hjem","id":2}]}	54	true	3	false	{54,65}

- id – unik id settes av databasen
- title – Listens tittel
- content – listens innhold som JSON-objekt
- owner – brukerid på den som laget listen
- activated – default = true. false hvis listen slettes
- idcount – brukes for å gi unik id til listelementer. Må økes for hvert nye element.
- Public – default = false. true hvis listen kan vises offentlig.
- Contributors – array med brukerid til brukere som kan redigere og se listen.

BRUKERHÅNDTERING:

Lage ny bruker:

METHOD: POST

ENDPOINT: /users/register

REQUEST:

Send inn et object (user) som JSON. Det skal inneholde følgende properties:

- username – Nytt brukernavn (string).
- email – Ny email (string, eksempel@hotmail.com).
- password – Nytt passord (string).

Fetch-eksempel:

```
let user = {
        username: myUsername,
        email: myEmail,
        password: myPassword
    };
fetch('https://to-do-list-gr1.herokuapp.com/users/register', {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
        },
        body: JSON.stringify(user)
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Velkommen som bruker, (myUsername)'

LOGG INN:

METHOD: POST

ENDPOINT: /users/login

REQUEST:

Send inn et object (user) som JSON. Det skal inneholde følgende properties:

- email – Email (string, eksempel@hotmail.com).
- password – Passord (string).

Fetch-eksempel:

```
let user = {
        email: myEmail,
        password: myPassword
    };
fetch('https://to-do-list-gr1.herokuapp.com/users/login, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
        },
        body: JSON.stringify(user)
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – eks. 'Hello, (myUsername)'
- username – myUsername
- id – unique user id generated by the database
- token – token used to authenticate together with id

CHANGE PASSWORD:

METHOD: POST

ENDPOINT: /users/changepassword

REQUEST:

Send inn følgende data i header:

- Auth – token du fikk ved login.
- userId – brukerid du fikk ved login.
- newPassword – myPassword (string)

Fetch-eksempel:

```
fetch(‘https://to-do-list-gr1.herokuapp.com/users/changepassword, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'userId': myUserId,
            'newPassword': myNewPassword,
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – ‘Passordet ditt er endret’

DELETE USER:

METHOD: POST

ENDPOINT: /users/delete

REQUEST:

Send inn følgende data i header:

- Auth – token du fikk ved login.
- userId – brukerid du fikk ved login.

Fetch-eksempel:

```
fetch(‘https://to-do-list-gr1.herokuapp.com/users/delete, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
```

```
            'Auth': myToken,
            'userId': myUserId,
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Deleted user (myUsername)'

LISTEHÅNDTERING:

LAGE NY LISTE:

METHOD: POST

ENDPOINT: /lists/add

REQUEST:

Send inn et object (list) som JSON. Det skal inneholde følgende properties:

- listTitle – myListTitle

Fetch-eksempel:

```
let list = {
        listTitle: myListTitle,
    };
fetch('https://to-do-list-gr1.herokuapp.com/lists/add, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId
        },
        body: JSON.stringify(list)
```

```
}).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Ny liste laget. Listenavn: (myListTitle).'

OPPDATERE LISTE:

METHOD: POST

ENDPOINT: /lists/update/list

REQUEST:

Først må man hente listen man vil oppdatere fra databasen (se «Hente enkelt liste»). Deretter oppdaterer man dataen på klientsiden og sender denne tilbake på følgende måte:

Header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id.
- NewIdCount – listens verdi fra databasekolonnen 'idcount' + 1

Body:

- listElements – objekt som sendes som JSON

Fetch-eksempel:

```
let listElements = {
        title: myListTitle (string),
        duedate: myDueDate (eks. 11/04/2018),
        tag: myTag (string),
        id: listElementId (int),
    };
fetch('https://to-do-list-gr1.herokuapp.com/lists/update/list, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
```

```
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId,
            'NewIdCount': newIdCount
        },
        body: JSON.stringify(listElements)
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Listen er oppdatert'

OPPDATERE LISTETITTEL:

METHOD: POST

ENDPOINT: /lists/update/listtitle

REQUEST:

Først må man hente listen man vil oppdatere fra databasen (se «Hente enkelt liste»). Deretter oppdaterer man dataen på klientsiden og sender denne tilbake på følgende måte:

Header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id.
- NewTitle – myNewTitle

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/update/listtitle,
{
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
```

```
            'UserId': myUserId,
            'ListId': myListId,
            'NewTitle': myNewTitle
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Listetittelen er oppdatert'

SLETTE LISTE:

METHOD: POST

ENDPOINT: /lists/delete

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id.

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/delete, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Deleted list 'myListTitle'

SLETTE LISTEELEMENT:

METHOD: POST

ENDPOINT: /lists/delete/element

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id
- listElementId – listelementet du ønsker å slette sin id

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/delete/element, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId,
            'listElementId': listElementId
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Deleted listelement'

VEKSLE MELLOM OFFENTLIG/PRIVAT LISTE:

METHOD: POST

ENDPOINT: /lists/togglepublic

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/togglepublic, {
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Listen er gjort offentlig/privat'

LEGGE TIL/FJERNE BRUKERTILGANG:

METHOD: POST

ENDPOINT: /lists/togglecontributor

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.

- UserId – brukerid du fikk ved login.
- ListId – listens id
- Email – email til bruker du ønsker å legge til/fjerne

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/togglecontributor,
{
        method: "POST",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId,
            'email': email
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- msg – 'Brukeren kan ikke lengre se denne listen.'/' Brukeren kan nå se og endre denne listen.'

HENTE ALLE BRUKERENS LISTER:

METHOD: GET

ENDPOINT: /lists/getall

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/getall, {
```

```
        method: "GET",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- lists – radene fra databasen hvor din brukerid finnes i «contributor»-kolonnen

HENTE ENKELT LISTE:

METHOD: GET

ENDPOINT: /lists/get

REQUEST:

Send følgende informasjon i header:

- Auth – token du fikk ved login.
- UserId – brukerid du fikk ved login.
- ListId – listens id

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/get, {
        method: "GET",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json',
            'Auth': myToken,
            'UserId': myUserId,
            'ListId': myListId
        }
```

```
}).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- datarow – raden fra databasen.

HENTE ALLE OFFENTLIGE LISTER:

METHOD: GET

ENDPOINT: /lists/getpublic

Fetch-eksempel:

```
fetch('https://to-do-list-gr1.herokuapp.com/lists/getpublic, {
        method: "GET",
        headers: {
            'Content-Type': 'application/json; charset=utf-8',
            'Accept': 'application/json'
        }
    }).then...
```

Response:

Et suksessfullt svar fra serveren vil inneholde (JSON):

- lists – alle rader fra databasen (lister) hvor kolonnen «public» = true.