

POST localhost:3000/user

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	Key	Value
<input checked="" type="checkbox"/>	username	admin
<input checked="" type="checkbox"/>	password	1234
	Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview JSON

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 1,
5   "info": "",
6   "serverStatus": 2,
7   "warningStatus": 0,
8   "changedRows": 0
9 }
```

admin käyttäjä lisätty:

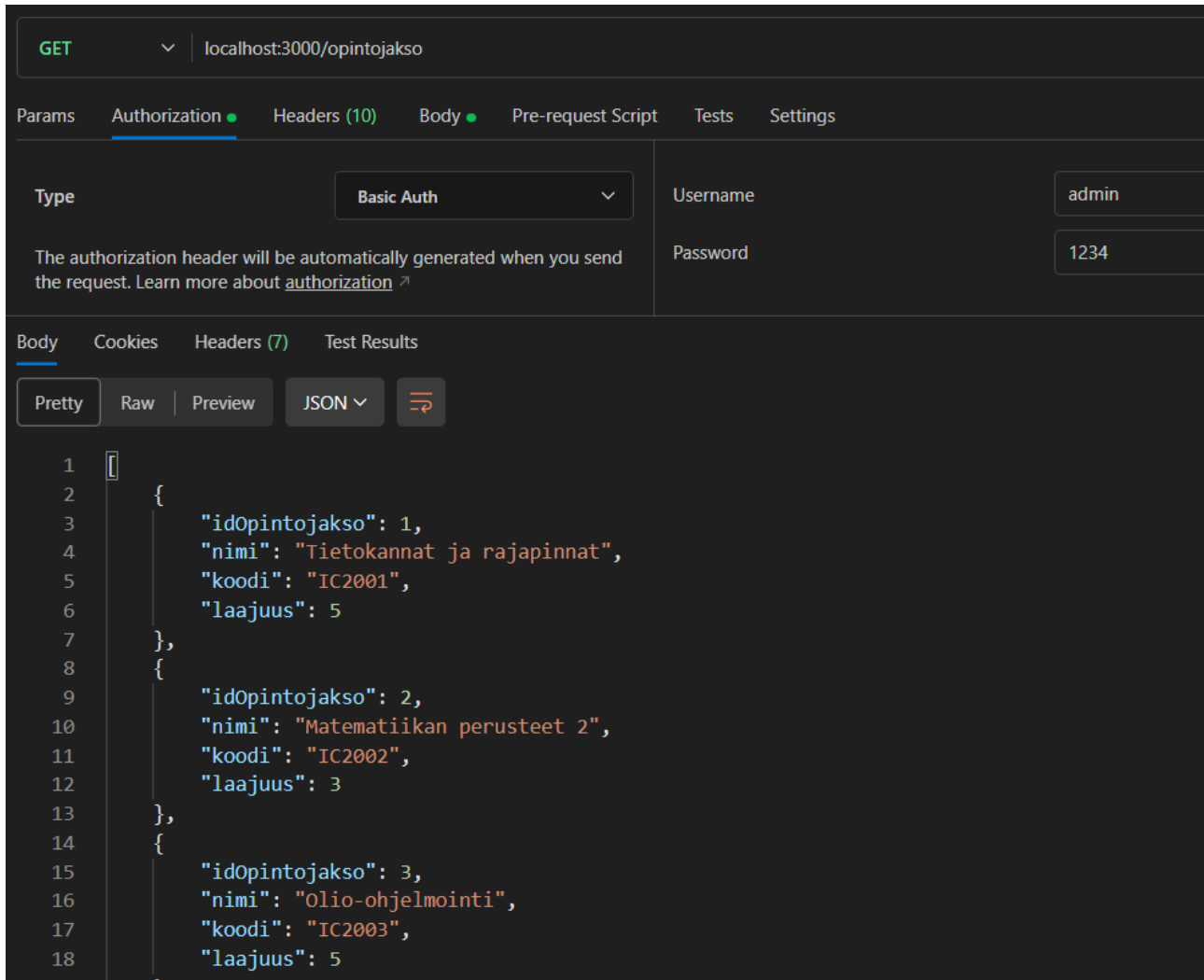
```
1 [
2   {
3     "id_user": 1,
4     "username": "admin",
5     "password": "$2b$10$CrXSTdv3f18oinE/Vv2E3u45Tn6KTxpGjcYSWyq8GgNw3AvhIZvUC"
6   }
7 ]
```

Kuten kuvassa näkyy: opintojakso, user ja login ovat salauksen takana

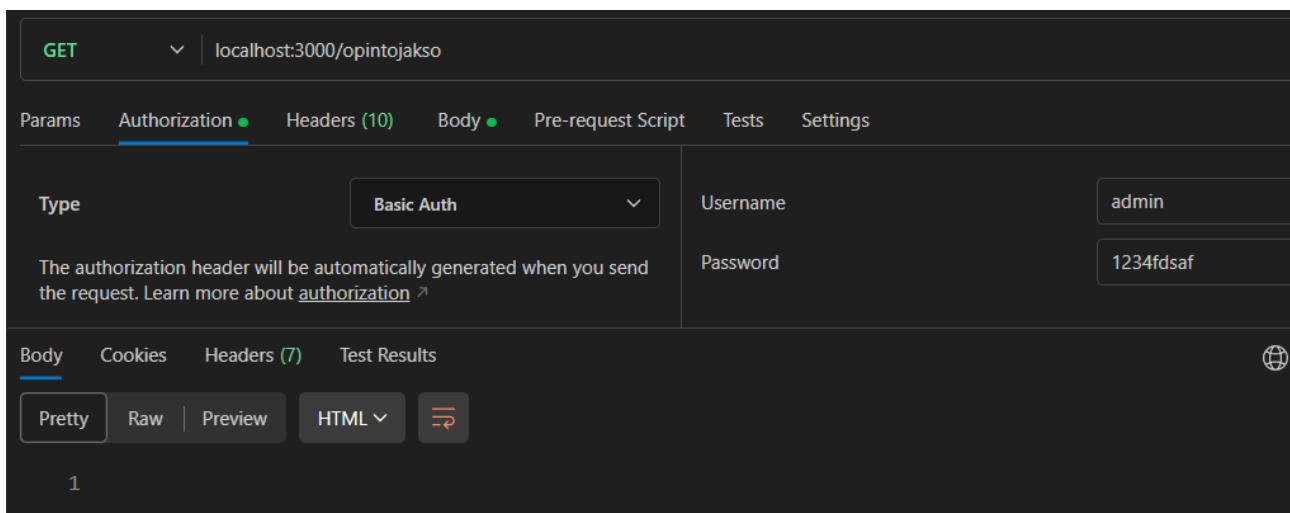
```
10 var arviointiRouter = require('./routes/arviointi');
11 var opiskelijaRouter = require('./routes/opiskelija');
12 var opintojaksoRouter = require('./routes/opintojakso');
13 var userRouter = require('./routes/user');
14 var loginRouter = require('./routes/login');
15
16
17
18
19 var app = express();
20
21 app.use(logger('dev'));
22 app.use(express.json());
23 app.use(express.urlencoded({ extended: false }));
24 app.use(cookieParser());
25 app.use(express.static(path.join(__dirname, 'public')));
26
27
28 app.use('/arviointi', arviointiRouter);
29 app.use('/opiskelija', opiskelijaRouter);
30
31 app.use(basicAuth( { authorizer: myAuthorizer, authorizeAsync:true, } ))
32
33 app.use('/opintojakso', opintojaksoRouter);
34 app.use('/user', userRouter);
35 app.use('/login', loginRouter);
36
37 function myAuthorizer(username, password,cb){
38     db.query('SELECT password FROM user WHERE username = ?', [username],
39         function(dbError, dbResults, fields) {
40             if(dbError){
41                 response.json(dbError);
42             }
43         }
44     );
45 }
```

Tässä pari kuvaa toiminnallisuudesta:

Tiedot tulostuivat opintojaksoista koska username ja password ovat oikein



Tässä sama uudestaan mutta väärällä salasanalla, joten ei tulostu mitään



```
wrong password  
GET /opintojakso 401 106.900 ms - 0
```

Tässä taas toimii vaikka salasana on väärin koska opiskelija app.use tiedosto on ennen salausta/autentikointia

The screenshot shows a REST client interface with the following details:

- Request:** GET localhost:3000/opiskelija
- Authorization:** Basic Auth (admin/1234fdsaf)
- Response Body (JSON):**

```
[  
  {  
    "idOpiskelija": 1,  
    "etunimi": "Pekka",  
    "sukunimi": "Pouta",  
    "luokkatunnus": "TVT24SPL",  
    "osoite": "Pekankuja 1"  
  },  
  {  
    "idOpiskelija": 2,  
    "etunimi": "Teppo",  
    "sukunimi": "Tulppu",  
    "luokkatunnus": "TVT24SPL",  
    "osoite": "Tulpunkuja 1"  
  },  
  {  
    "idOpiskelija": 3,  
    "etunimi": "Pelle",  
    "sukunimi": "Peloton",  
    "luokkatunnus": "TVT24SPL",  
    "osoite": "Pellonkatu 1"  
  }  
]
```

Tietojen tulostus yritys käyttäjällä, jota ei ole olemassa(tiedot salauksen/autentikoinnin takana:

The screenshot shows a REST client interface with a dark theme. At the top, the method is 'GET' and the URL is 'localhost:3000/user'. Below this is a tabbed interface with 'Params', 'Authorization', 'Headers (10)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Authorization' tab is selected, showing 'Type' as 'Basic Auth'. Below this, a note states: 'The authorization header will be automatically generated when you send the request. Learn more about [authorization](#)'. To the right, the 'Username' field contains 'kayttaja' and the 'Password' field contains '1234'. Below the 'Authorization' tab is another tabbed interface with 'Body', 'Cookies', 'Headers (7)', and 'Test Results'. The 'Body' tab is selected, showing a 'Pretty' button and a 'Raw' button. Below these buttons, the text '1' is visible.

```
user does not exists
GET /user 401 10.205 ms - 0
```