

# 17 Opintorekisteri

## Opiskelijataulun luku GET:llä

The screenshot displays a development environment with a REST client (Postman) testing a GET endpoint. The terminal shows a successful response with a JSON array of student data. The code editor shows the Express.js server code.

**Terminal Output:**

```
GET /opiskelija 200 50.225 ms - 804
```

**JSON Response:**

```
[{"idOpiskelija":1,"Etunimi":"Aku","Sukunimi":"Ankka","Opiskelijanumero":"1313","Luokkatunnus":"TVT22SPL","Osoite":"Ankkalinn 313"}, {"idOpiskelija":2,"Etunimi":"Roope","Sukunimi":"Ankka","Opiskelijanumero":"1414","Luokkatunnus":"TVT22SPL","Osoite":"Ankkalinn 314"}, {"idOpiskelija":3,"Etunimi":"Iina","Sukunimi":"Ankka","Opiskelijanumero":"1515","Luokkatunnus":"TVT22SPL","Osoite":"Ankkalinn 315"}, {"idOpiskelija":4,"Etunimi":"Mikku","Sukunimi":"Hini","Opiskelijanumero":"1616","Luokkatunnus":"TVT22SPO","Osoite":"Ankkalinn 316"}, {"idOpiskelija":5,"Etunimi":"Hessu","Sukunimi":"Hopo","Opiskelijanumero":"1717","Luokkatunnus":"TVT22SPO","Osoite":"Ankkalinn 317"}, {"idOpiskelija":6,"Etunimi":"Pelle","Sukunimi":"Pelton","Opiskelijanumero":"1818","Luokkatunnus":"TVT22SPL","Osoite":"Ankkalinn 318"}]
```

**Code Editor (app.js):**

```
1 var express = require('express');
2 var path = require('path');
3 var cookieParser = require('cookie-parser');
4 var logger = require('morgan');
5 const db = require('./database');
6 const bcrypt = require('bcryptjs');
7 const basicAuth = require('express-basic-auth');
8
9 var arviointiRouter = require('./routes/arviointi');
10 var opiskelijaRouter = require('./routes/opiskelija');
11 var opintojaksoRouter = require('./routes/opintojakso');
12 var userRouter = require('./routes/user');
13
```

# Opintojaksotaulun luku GET:llä

The screenshot displays a development environment with Visual Studio Code and Postman. The VS Code editor shows a Node.js application with the following code in `app.js`:

```
1 var express = require('express');
2 var path = require('path');
3 var cookieParser = require('cookie-parser');
4 var logger = require('morgan');
5 const db = require('./database');
6 const bcrypt = require('bcryptjs');
7 const basicAuth = require('express-basic-auth');
8
9 var arviointiRouter = require('./routes/arviointi');
10 var opiskelijaRouter = require('./routes/opiskelija');
11 var opintojaksoRouter = require('./routes/opintojakso');
12 var userRouter = require('./routes/user');
13
14 app.use(logger('dev'));
15 app.use(cookieParser());
16 app.use(express.json());
17 app.use(express.urlencoded({ extended: false }));
18 app.use(basicAuth({ username: 'admin', password: 'salainen' }));
19
20 app.use(arviointiRouter);
21 app.use(opiskelijaRouter);
22 app.use(opintojaksoRouter);
23 app.use(userRouter);
24
25 app.listen(3000, () => {
26   console.log('Server is running on port 3000');
27 });
```

The Postman interface shows a GET request to `localhost:3000/opintojakso`. The response is a JSON array of study courses:

```
[{"idOpintojakso":1,"Nimi":"Olio-ohjelmointi","Koodi":"YNG22229","Laajuus":5}, {"idOpintojakso":2,"Nimi":"Tietokannat ja rajapinnat","Koodi":"YNG22230","Laajuus":5}, {"idOpintojakso":3,"Nimi":"Johdatus ohjelmointiin","Koodi":"INO2234","Laajuus":5}, {"idOpintojakso":4,"Nimi":"Engineer English","Koodi":"YNG22231","Laajuus":5}, {"idOpintojakso":5,"Nimi":"Soveltava matematiikka ja fysiikka","Koodi":"YNG22232","Laajuus":5}, {"idOpintojakso":6,"Nimi":"Tietotekniikan sovellusprojekti","Koodi":"YNG22234","Laajuus":15}, {"idOpintojakso":7,"Nimi":"VHDL-kieli","Koodi":"YNG22235","Laajuus":3}]
```

The VS Code Explorer shows the project structure:

- bin
- models
- arviointi\_modeljs
- login\_modeljs
- opintojakso\_modeljs
- opiskelija\_modeljs
- user\_modeljs
- node\_modules
- public
- routes
- arviointijs

## Arvointitaulun luku GET:llä

The screenshot shows a Visual Studio Code editor with a REST client interface. The active request is a GET request to `localhost:3000/arviointi`. The response is a large JSON array containing 16 assessment records. Each record includes fields like `idArviointi`, `Arvosana`, `Paivays`, `idOpiskelija`, and `idOpintojakso`.

```
GET /arviointi 200 8.260 ms - 1345
```

```
RowDataPacket {
  idArviointi: 12,
  Arvosana: 4,
  Paivays: '2023-01-29T22:00:00.000Z',
  idOpiskelija: 5,
  idOpintojakso: 2
},
RowDataPacket {
  idArviointi: 13,
  Arvosana: 5,
  Paivays: '2023-01-29T22:00:00.000Z',
  idOpiskelija: 5,
  idOpintojakso: 4
},
RowDataPacket {
  idArviointi: 16,
  Arvosana: 3,
  Paivays: '2023-02-09T22:00:00.000Z',
  idOpiskelija: 3,
  idOpintojakso: 1
}
```

The REST client interface shows the following details:

- Method: GET
- URL: localhost:3000/arviointi
- Status: 200 OK
- Time: 17 ms
- Size: 1.55 KB

The response body is a large JSON array of 16 assessment records, each containing fields like `idArviointi`, `Arvosana`, `Paivays`, `idOpiskelija`, and `idOpintojakso`.

## Lisätään Opiskelija POST:lla

The screenshot shows a Visual Studio Code editor with a REST client interface. The active request is a POST request to `localhost:3000/opiskelija`. The response is a 200 OK status.

```
POST /opintojakso/ 200 6.693 ms - 649
```

```
RowDataPacket {
  idOpintojakso: 9,
  Nimi: 'Matti166',
  Koodi: 'matti166',
  Laajuus: 5
}
```

The REST client interface shows the following details:

- Method: POST
- URL: localhost:3000/opiskelija
- Status: 200 OK
- Time: 17 ms
- Size: 373 B

The request body is a JSON object with the following fields:

```
{
  "idOpintojakso": "NULL",
  "Etunimi": "Ville",
  "Sukunimi": "Vallaton",
  "Opiskelijanumero": "9999",
  "Luokkatunnus": "TVTXXXX",
  "Osoite": "Mehukuja 1"
}
```

Nähdään GET:llä että lisäys onnistui ja id:ski tuli 11 koska välistä on deletoitu opiskelijoita

The screenshot shows a Visual Studio Code editor on the left with a file named `opiskelija_model.js` containing a `RowDataPacket` array. The array has two objects, each representing a student with fields like `idOpiskelija`, `Etunimi`, `Sukunimi`, `Opiskelijanumero`, `Luokkatunnus`, and `Osoite`. The first object has `idOpiskelija: 6` and the second has `idOpiskelija: 11`. On the right, the Postman application shows a GET request to `localhost:3000/opiskelija` with a status of 200 OK. The response body is a JSON array with two objects, matching the data in the code editor.

```
RowDataPacket {  
  idOpiskelija: 6,  
  Etunimi: 'Pelle',  
  Sukunimi: 'Peltonen',  
  Opiskelijanumero: '1818',  
  Luokkatunnus: 'TVT225PL',  
  Osoite: 'Ankkalinnä 317'  
},  
RowDataPacket {  
  idOpiskelija: 11,  
  Etunimi: 'Ville',  
  Sukunimi: 'Vallaton',  
  Opiskelijanumero: '9999',  
  Luokkatunnus: 'TVTXXXX',  
  Osoite: 'Mehukuja 1'  
}
```

GET /opiskelija 200 12.099 ms - 937

localhost:3000/opiskelija

GET localhost:3000

Params Authorization Headers (10) Body Pre-request Script Tests Settings

KEY VALUE DESCRIPTION

<input checked="" type="checkbox"/>	idOpintojakso	NULL	
<input checked="" type="checkbox"/>	Etunimi	Ville	
<input checked="" type="checkbox"/>	Sukunimi	Vallaton	
<input checked="" type="checkbox"/>	Opiskelijanumero	9999	
<input checked="" type="checkbox"/>	Luokkatunnus	TVTXXXX	
<input checked="" type="checkbox"/>	Osoite	Mehukuja 1	
	Key	Value	Description

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
48 {  
49   "Osoite": "Ankkalinnä 317"  
50 },  
51 {  
52   "idOpiskelija": 6,  
53   "Etunimi": "Pelle",  
54   "Sukunimi": "Peltonen",  
55   "Opiskelijanumero": "1818",  
56   "Luokkatunnus": "TVT225PL",  
57   "Osoite": "Ankkalinnä 318"  
58 }  
59 },  
60 {  
61   "idOpiskelija": 11,  
62   "Etunimi": "Ville",  
63   "Sukunimi": "Vallaton",  
64   "Opiskelijanumero": "9999",  
65   "Luokkatunnus": "TVTXXXX",  
66   "Osoite": "Mehukuja 1"  
67 }  
68 }
```

Päivitetään Villen luokkatunnus PUT:lla ja tarkistetaan GET:llä

The screenshot shows the same Visual Studio Code editor on the left. On the right, the Postman application shows a PUT request to `localhost:3000/opiskelija/11` with a status of 200 OK. The response body is a JSON object with fields like `fieldCount`, `affectedRows`, `insertId`, `serverStatus`, `warningCount`, `message`, `protocol41`, and `changedRows`.

```
RowDataPacket {  
  idOpiskelija: 11,  
  Etunimi: 'Pelle',  
  Sukunimi: 'Peltonen',  
  Opiskelijanumero: '1818',  
  Luokkatunnus: 'TVT225PL',  
  Osoite: 'Ankkalinnä 318'  
},  
RowDataPacket {  
  idOpiskelija: 11,  
  Etunimi: 'Ville',  
  Sukunimi: 'Vallaton',  
  Opiskelijanumero: '9999',  
  Luokkatunnus: 'TVTXXXX',  
  Osoite: 'Mehukuja 1'  
}
```

GET /opiskelija 200 12.099 ms - 937

PUT /opiskelija/11 200 9.077 ms - 161

localhost:3000/opiskelija/11

PUT localhost:3000

Params Authorization Headers (10) Body Pre-request Script Tests Settings

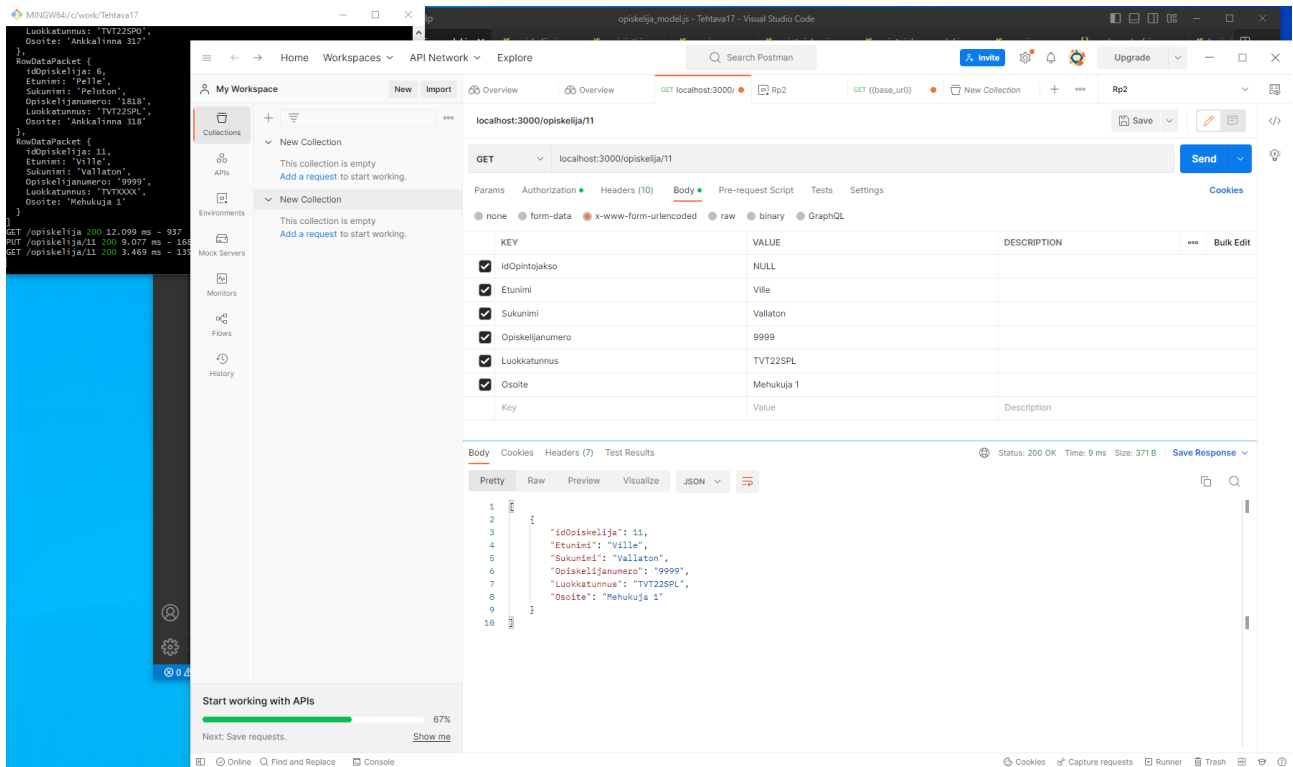
KEY VALUE DESCRIPTION

<input checked="" type="checkbox"/>	idOpintojakso	NULL	
<input checked="" type="checkbox"/>	Etunimi	Ville	
<input checked="" type="checkbox"/>	Sukunimi	Vallaton	
<input checked="" type="checkbox"/>	Opiskelijanumero	9999	
<input checked="" type="checkbox"/>	Luokkatunnus	TVT225PL	
<input checked="" type="checkbox"/>	Osoite	Mehukuja 1	
	Key	Value	Description

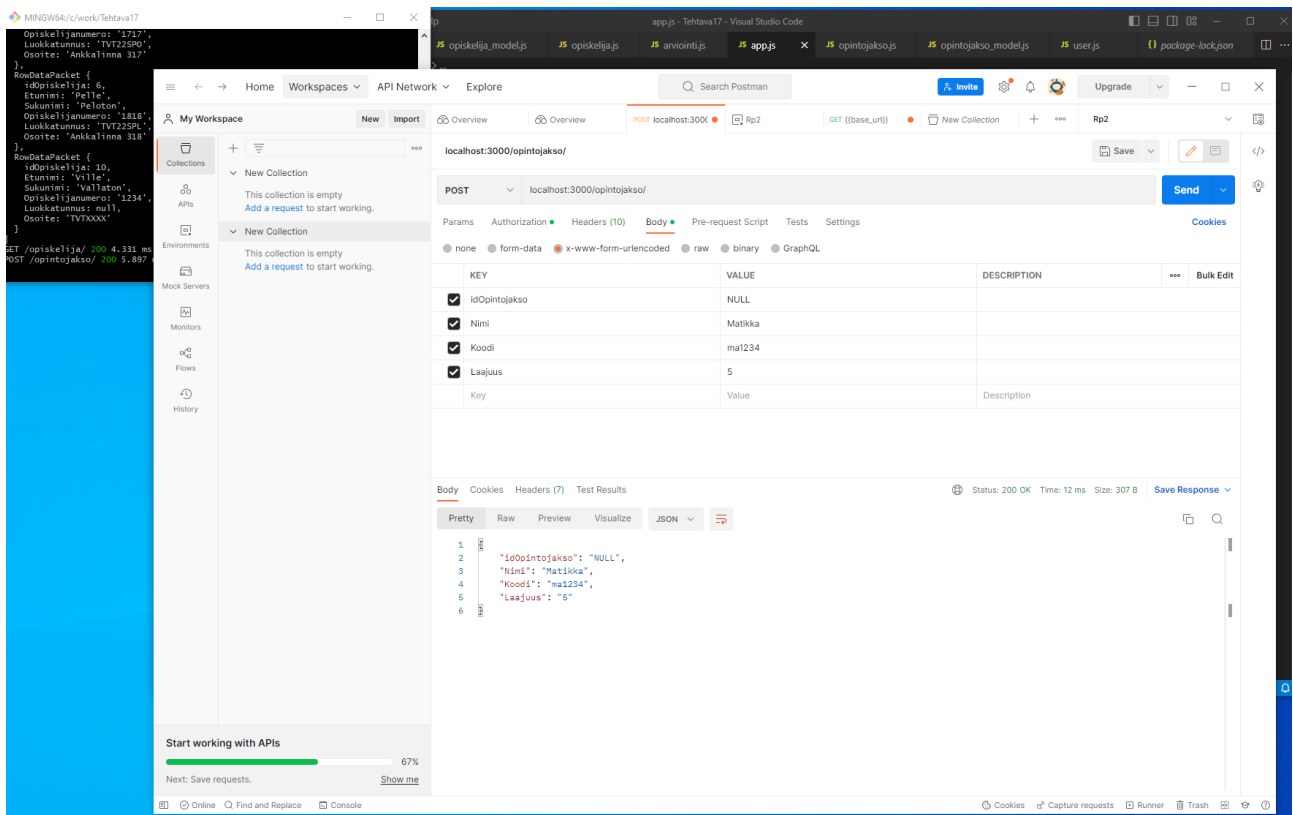
Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

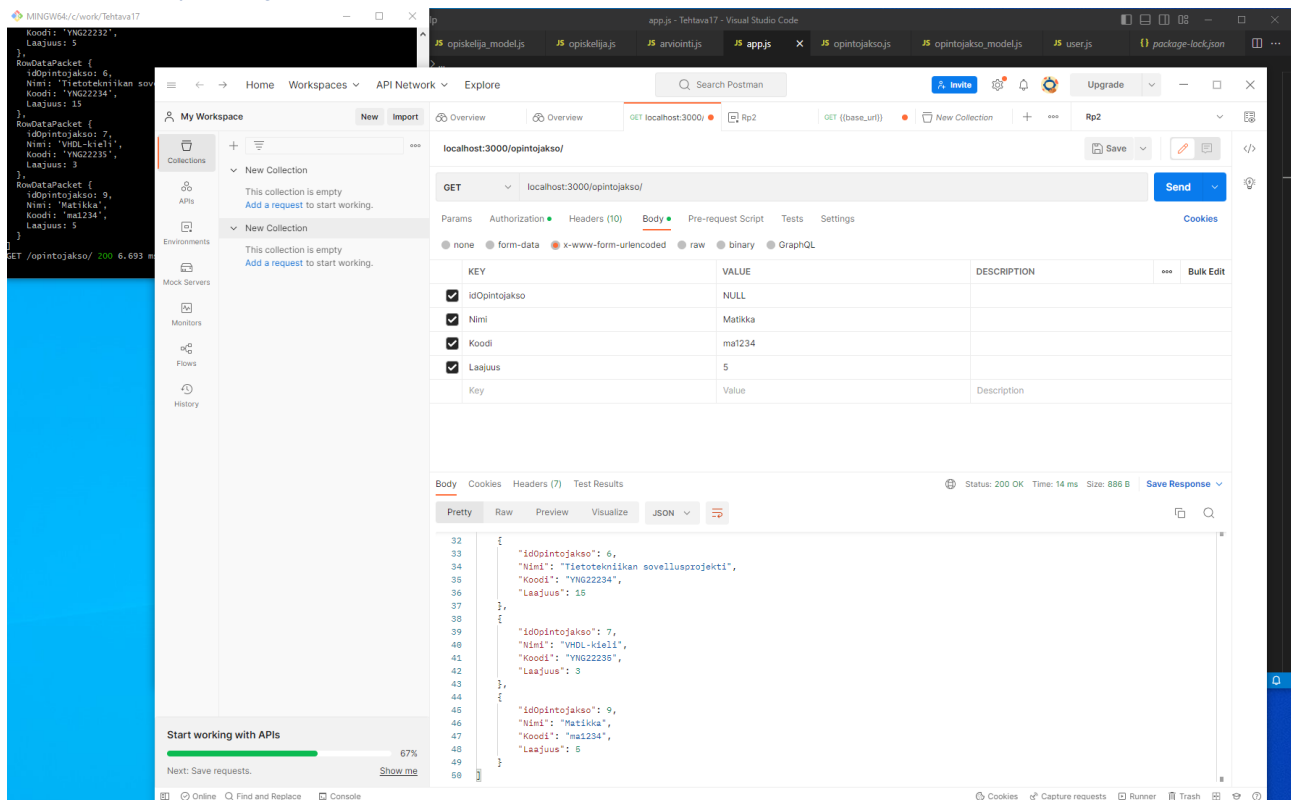
```
1 {  
2   "fieldCount": 0,  
3   "affectedRows": 1,  
4   "insertId": 0,  
5   "serverStatus": 2,  
6   "warningCount": 0,  
7   "message": "(Rows matched: 1 Changed: 1 Warnings: 0",  
8   "protocol41": true,  
9   "changedRows": 1  
10 }
```



## Lisätään opintojako POST:lla



## Luetaan opintojaksot GET:llä



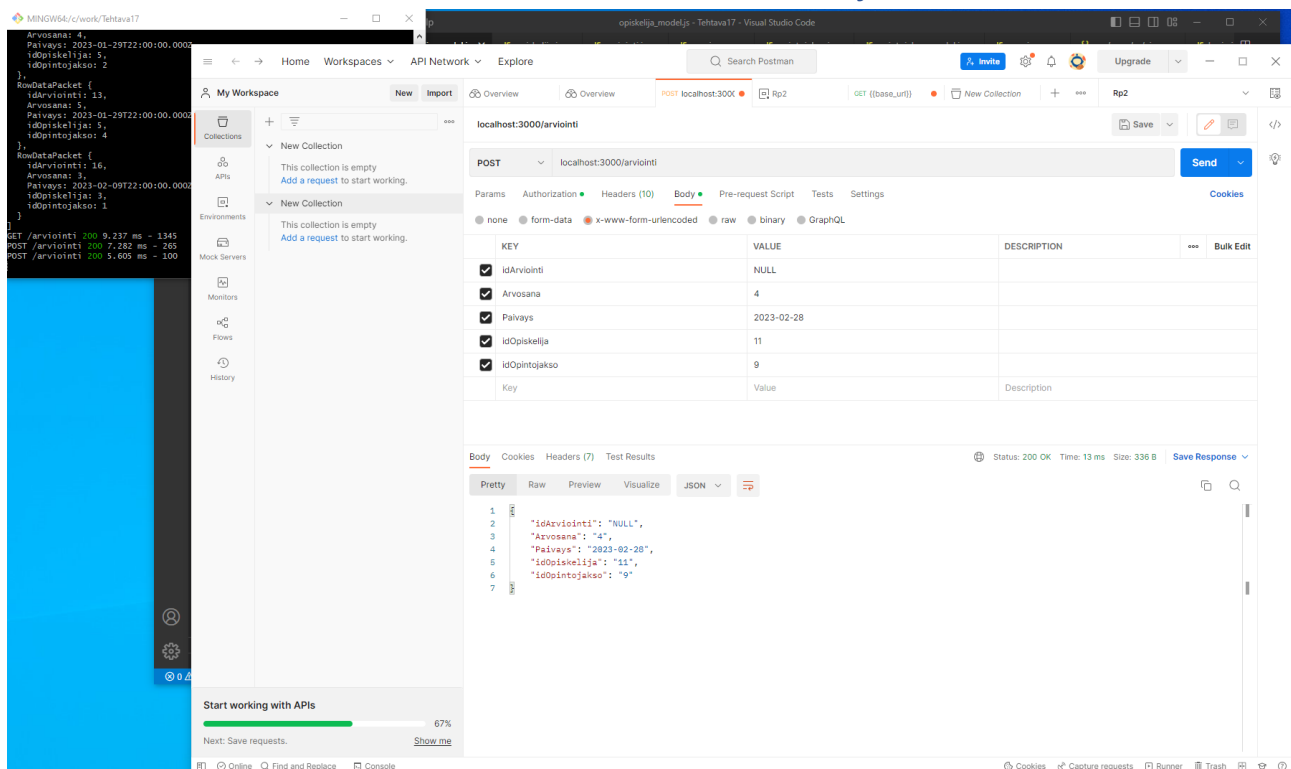
The screenshot shows the Postman interface with a GET request to `localhost:3000/opintojako/`. The request parameters are:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> idOpintojakso	NULL	
<input checked="" type="checkbox"/> nimi	Matikka	
<input checked="" type="checkbox"/> koodi	mat1234	
<input checked="" type="checkbox"/> laajuus	5	
Key	Value	Description

The response body is a JSON array of course objects:

```
32 [
33   {
34     "idOpintojakso": 6,
35     "nimi": "Tietotekniikan sovellusprojekti",
36     "koodi": "YNG22234",
37     "laajuus": 15
38   },
39   {
40     "idOpintojakso": 7,
41     "nimi": "YNDL-kielii",
42     "koodi": "YNG22235",
43     "laajuus": 3
44   },
45   {
46     "idOpintojakso": 9,
47     "nimi": "Matikka",
48     "koodi": "mat1234",
49     "laajuus": 5
50   }
51 ]
```

## Lisätään Villelle suoritus POST:lla arviointitauluun ja tarkistetaan GET:llä

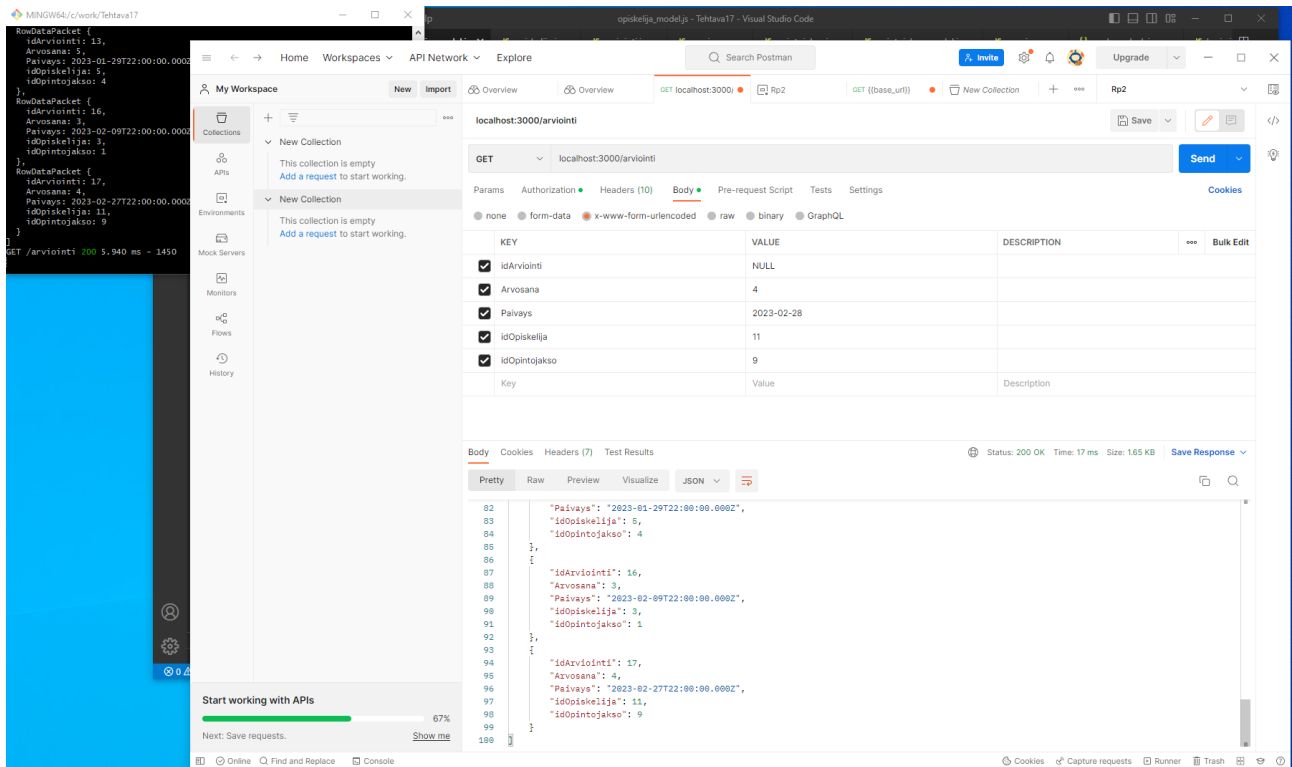


The screenshot shows the Postman interface with a POST request to `localhost:3000/arviointi`. The request parameters are:

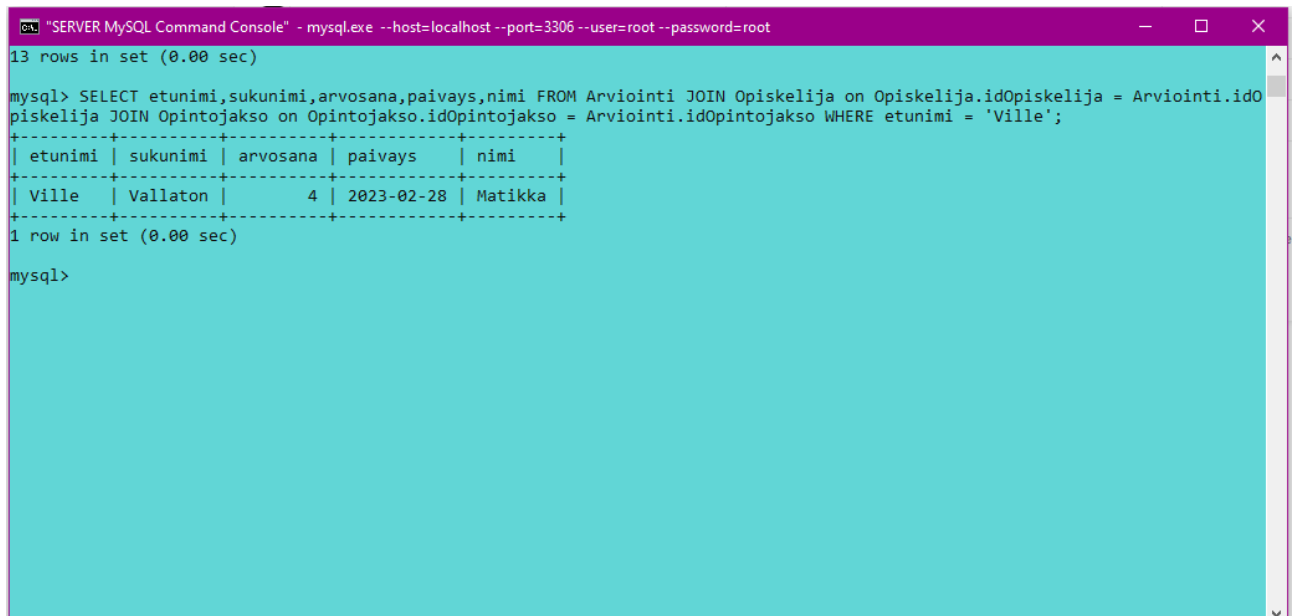
KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> idArviointi	NULL	
<input checked="" type="checkbox"/> Arvosana	4	
<input checked="" type="checkbox"/> Paivays	2023-02-28	
<input checked="" type="checkbox"/> idOpiskelija	11	
<input checked="" type="checkbox"/> idOpintojakso	9	
Key	Value	Description

The response body is a JSON object:

```
1 {
2   "idArviointi": "NULL",
3   "Arvosana": "4",
4   "Paivays": "2023-02-28",
5   "idOpiskelija": "11",
6   "idOpintojakso": "9"
7 }
```



## Testi vielä komentorivillä



## Deletoidaan Villen arviointi

The screenshot shows the Postman interface with a DELETE request to `localhost:3000/arviointi/17`. The response body is a JSON object with the following structure:

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "",
8   "protocol41": true,
9   "changedRows": 0
10 }
```

The status bar indicates `Status: 200 OK`, `Time: 12 ms`, and `Size: 363 B`. The response is saved as `Save Response`.

## Deleteoidaan opintojakso Matikka

The screenshot shows the Postman interface with a DELETE request to `localhost:3000/opintojakso/9`. The response body is a JSON object with the following structure:

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "",
8   "protocol41": true,
9   "changedRows": 0
10 }
```

The status bar indicates `Status: 200 OK`, `Time: 20 ms`, and `Size: 363 B`. The response is saved as `Save Response`.



## Deletoidaan opiskelija Ville

The screenshot shows the Postman interface with a DELETE request to `localhost:3000/opiskelija/11`. The request is successful, returning a 200 OK status. The response body is a JSON object:

```
{  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 2,  "warningCount": 0,  "message": "",  "protocol41": true,  "changedRows": 0}
```

The left sidebar shows a terminal window with the following commands and output:

```
GET /opintojakso 200 3.425 ms
DELETE /opintojakso/9 200 10.88 ms
DELETE /opiskelija/11 200 6.380 ms
```

## Testataan että autetikointia väärällä salasanalla ja nähdään että se pelittää

The screenshot shows the Postman interface with a DELETE request to `localhost:3000/opiskelija/`. The request is unauthorized, returning a 401 Unauthorized status. The response body is a single character: `1`.

The left sidebar shows the same terminal window as the previous screenshot, with the following commands and output:

```
GET /opintojakso 200 3.425 ms
DELETE /opintojakso/9 200 10.88 ms
DELETE /opiskelija/ 401 6.380 ms
```