

TEHTÄVÄNANTO: Lisää edelliseen tehtävään taulut **user** ja **login**. User sisältää käyttäjätunnuksen ja salasanan. Kryptaa salasanat käyttäen **bcrypt.js**:ää. Tee toiminnallisuus, jossa esim. Postmanilla annettu käyttäjätunnus ja salaus tarkistetaan user-tilusta (myAuthorizer-funktio).

1. user-tilun lisäys opintorekisteri-kantaan

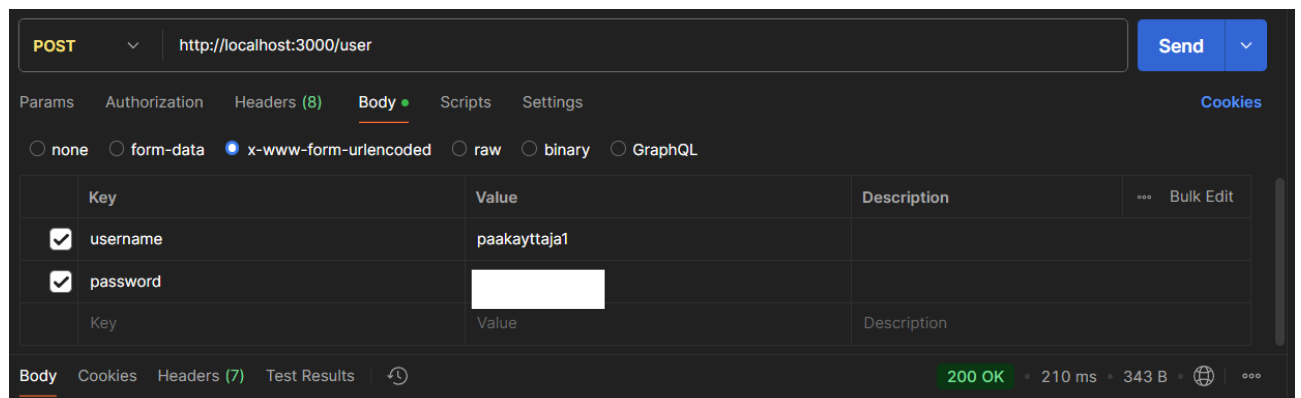
```
mysql> CREATE TABLE user(  
  -> idUser INT primary key auto_increment,  
  -> username VARCHAR(20),  
  -> password VARCHAR(255),  
  -> UNIQUE (username)  
  -> );  
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> desc user;
```

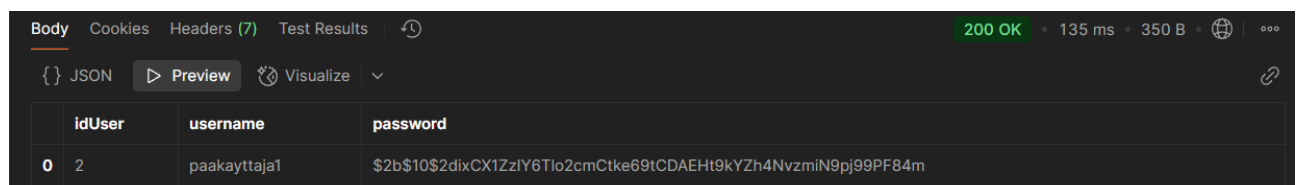
Field	Type	Null	Key	Default	Extra
idUser	int	NO	PRI	NULL	auto_increment
username	varchar(20)	YES	UNI	NULL	
password	varchar(255)	YES		NULL	

3 rows in set (0.02 sec)

2. Lisätään käyttäjä user-tiluun Postmanin POST:illa.



Tarkistetaan, että käyttäjä lisättiin user-tiluun onnistuneesti GET:illä. Ohjelma bcrypt.js salaa salasanan hashaamalla.



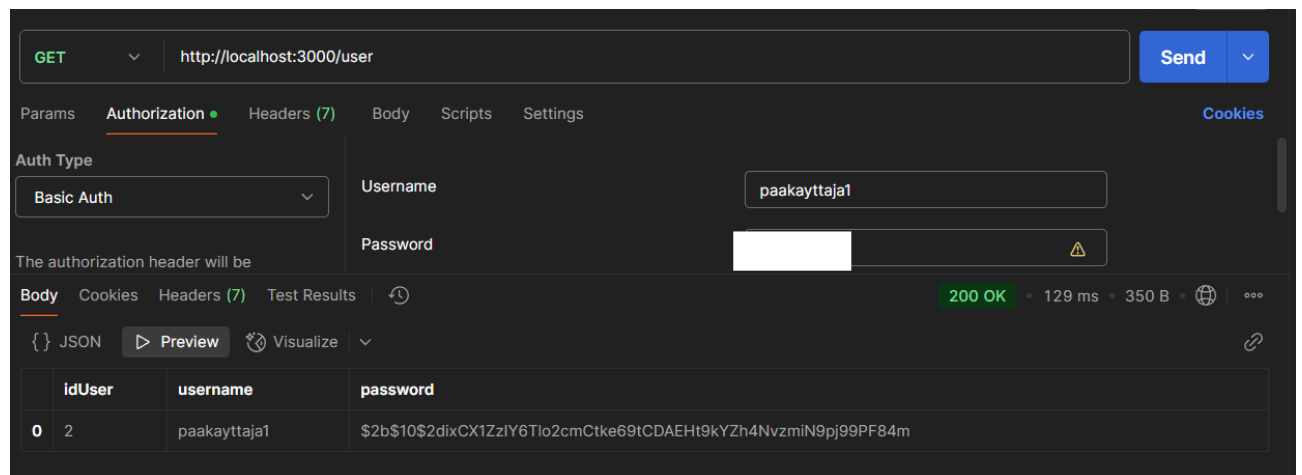
3. Järjestetään basicauthilla salattavat taulut seuraavasti:
- vas. **vihreällä** merkitty taulun app.use-middleware on kaikkien saatavilla.
 - vas. **punaisella** merkittujen taulujen app.use-middlewareet ovat saatavilla vain oikeilla tunnuksilla.

```
29 {
30   app.use("/opintojakso", opintojaksoRouter); // available to anyone
31 }
32 app.use(basicAuth({ authorizer: myAuthorizer, authorizeAsync: true }));
33
34 {
35   app.use("/opiskelija", opiskelijaRouter); // available only for authenticated users
36   app.use("/arviointi", arviointiRouter);
37   app.use("/user", userRouter);
38   app.use("/login", loginRouter);
39 }
```

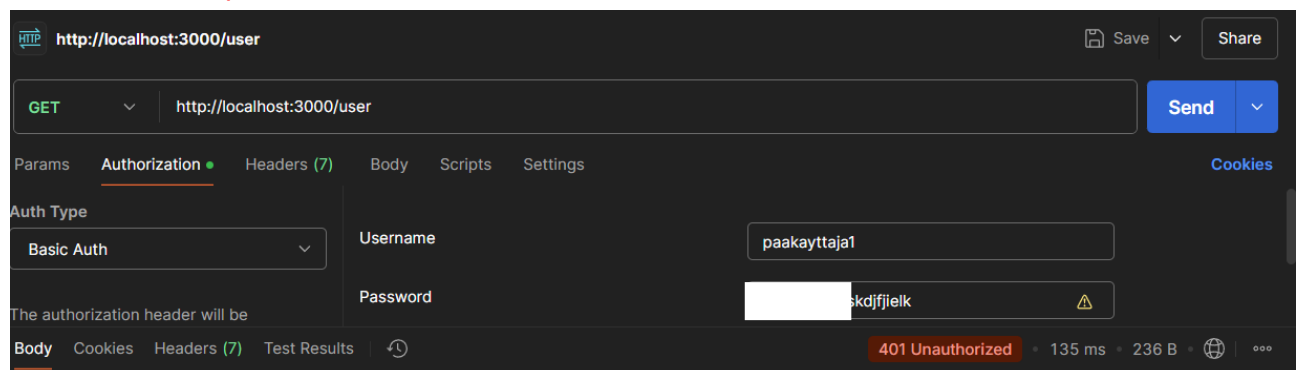
Tämä tarkoittaa sitä, että yleinen pääsy opintojakso-tauluun voidaan sallia, mutta myAuthorizer-funktion jälkeen esiintyviin tauluihin (opiskelija, arviointi, user ja login) ei pääse. Dokumentin kohdassa 5. on esitelty taulujen salausta käytännössä.

4. Käyttäjätunnusten testaus Postman-sovelluksen Authorization välilehdessä.

Demonstroitu **onnistunut** haku oikeilla tunnuksilla salatusta user-taulusta:



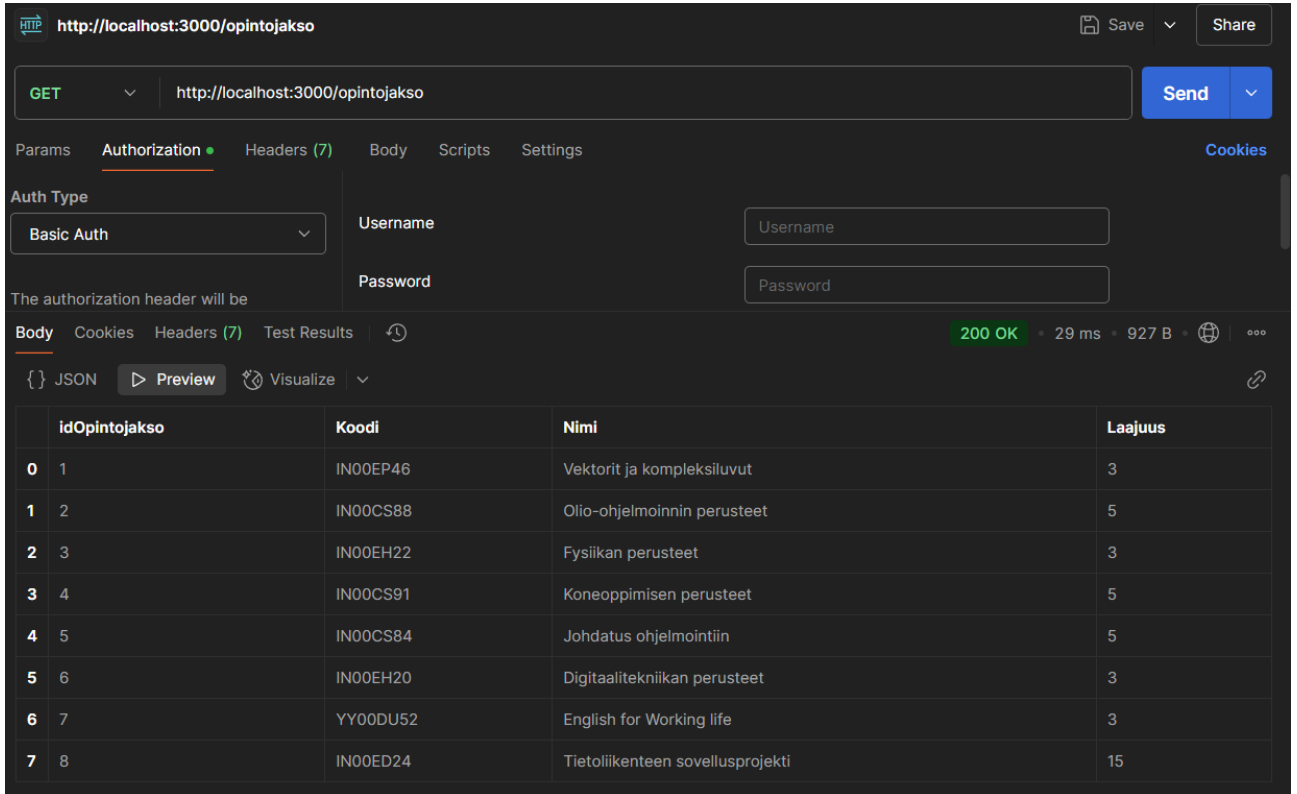
Demonstroitu **epäonnistunut** haku väärillä tunnuksilla salatusta user-taulusta:



5. Postman-sovelluksen testaus **ilman** Authorization-käyttäjätunnusta (yleinen käyttäjä).

Tässä osuudessa toteutetaan dokumentin kohdassa 3. esitettyä teoriaa salatuista tauluista.

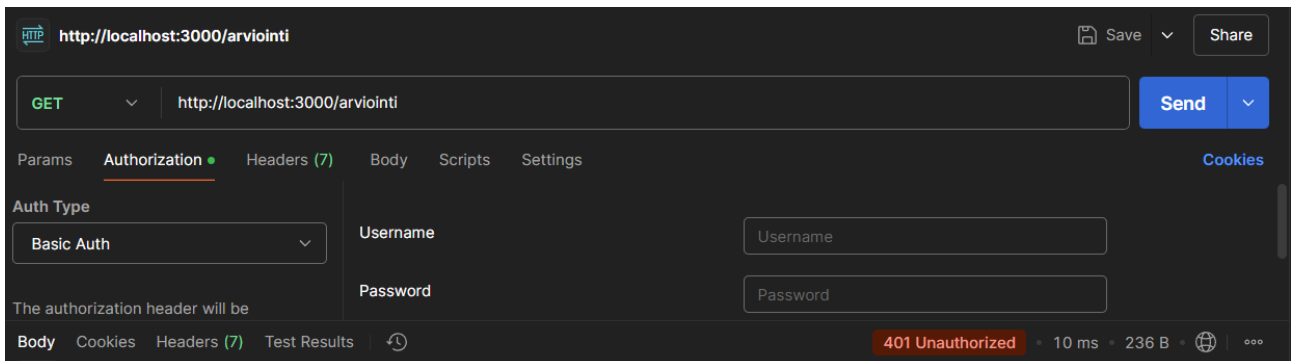
Taulun opintojakso haku *yleisellä käyttäjällä* **onnistuu**, koska se ei ole salattu:



The screenshot shows a web browser interface for an HTTP client. The URL bar displays `http://localhost:3000/opintojakso`. The method is set to `GET`. The response status is `200 OK` with a response time of 29 ms and a body size of 927 B. The response body is a JSON array of course objects.

	idOpintojakso	Koodi	Nimi	Laajuus
0	1	IN00EP46	Vektorit ja kompleksiluvut	3
1	2	IN00CS88	Olio-ohjelmoinnin perusteet	5
2	3	IN00EH22	Fysiikan perusteet	3
3	4	IN00CS91	Koneoppimisen perusteet	5
4	5	IN00CS84	Johdatus ohjelmointiin	5
5	6	IN00EH20	Digitaalitekniikan perusteet	3
6	7	YY00DU52	English for Working life	3
7	8	IN00ED24	Tietoliikenteen sovellusprojekti	15

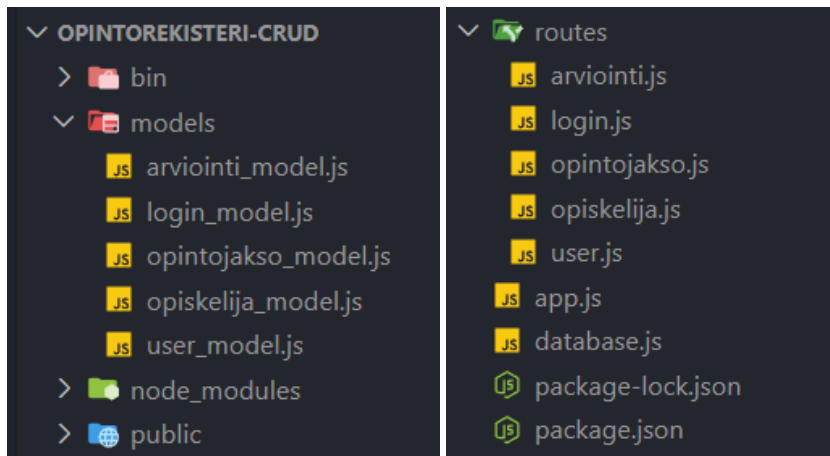
Taulun arviointi haku *yleisellä käyttäjällä* **epäonnistuu**, sillä se on salattu:



The screenshot shows a web browser interface for an HTTP client. The URL bar displays `http://localhost:3000/arviointi`. The method is set to `GET`. The response status is `401 Unauthorized` with a response time of 10 ms and a body size of 236 B.

6. Muuta palautettavaa:

Opintorekisteri-hakemisto:



myAuthorizer-funktio:

```
39 function myAuthorizer(username, password, cb) {
40   db.query(
41     "SELECT password FROM user WHERE username = ?",
42     [username],
43     function (dbError, dbResults, fields) {
44       if (dbError) {
45         response.json(dbError);
46       } else {
47         if (dbResults.length > 0) {
48           bcrypt.compare(password, dbResults[0].password, function (err, res) {
49             if (res) {
50               console.log("success");
51               return cb(null, true);
52             } else {
53               console.log("wrong password");
54               return cb(null, false);
55             }
56             response.end();
57           });
58         } else {
59           console.log("user does not exists");
60           return cb(null, false);
61         }
62       }
63     }
64   );
65 }
66
67 module.exports = app;
```