/\*

Question 1

Part 1:

List all users email and sex in order of most recently created. Do not include password hash in your output

\*/

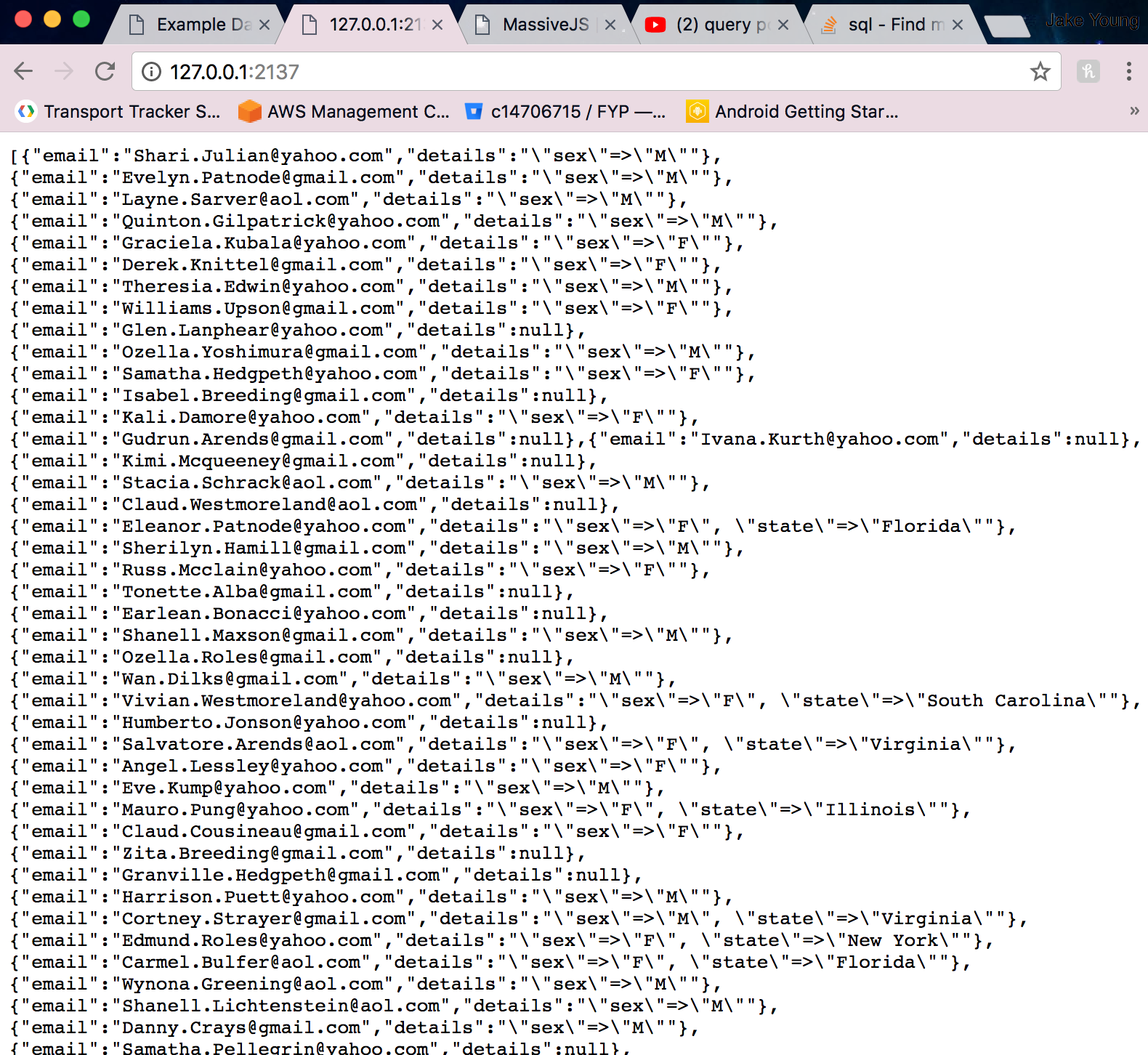
app.get('/users', function(req, res) {

db.run("Select email, details from users order by created\_at desc",function(err, result) {

res.send(result);

});

});



/\*

Question 1

Part 2:

Show above details of the specified user

\*/

app.get('/users/:id', function(req, res) {

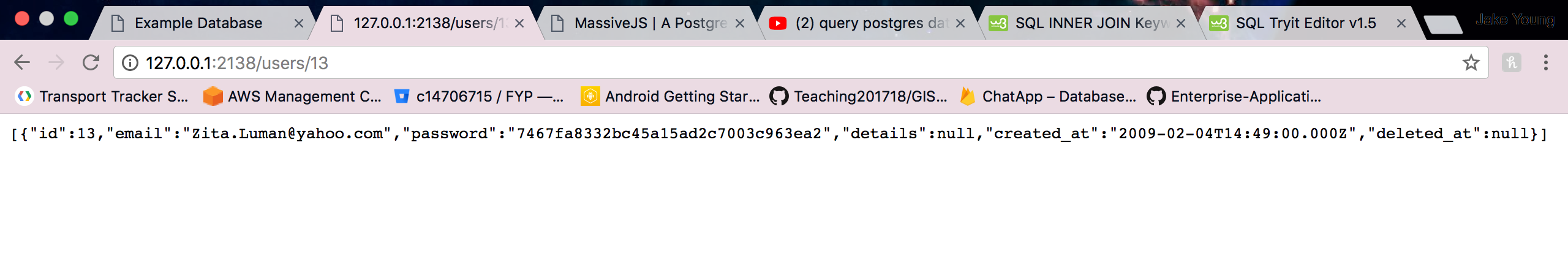
let id = req.params.id;

db.users.find({id: id}, function (err, result) {

res.send(result)

});

});



/\*

Question 1

Part 3:

List all products in ascending order of price

\*/

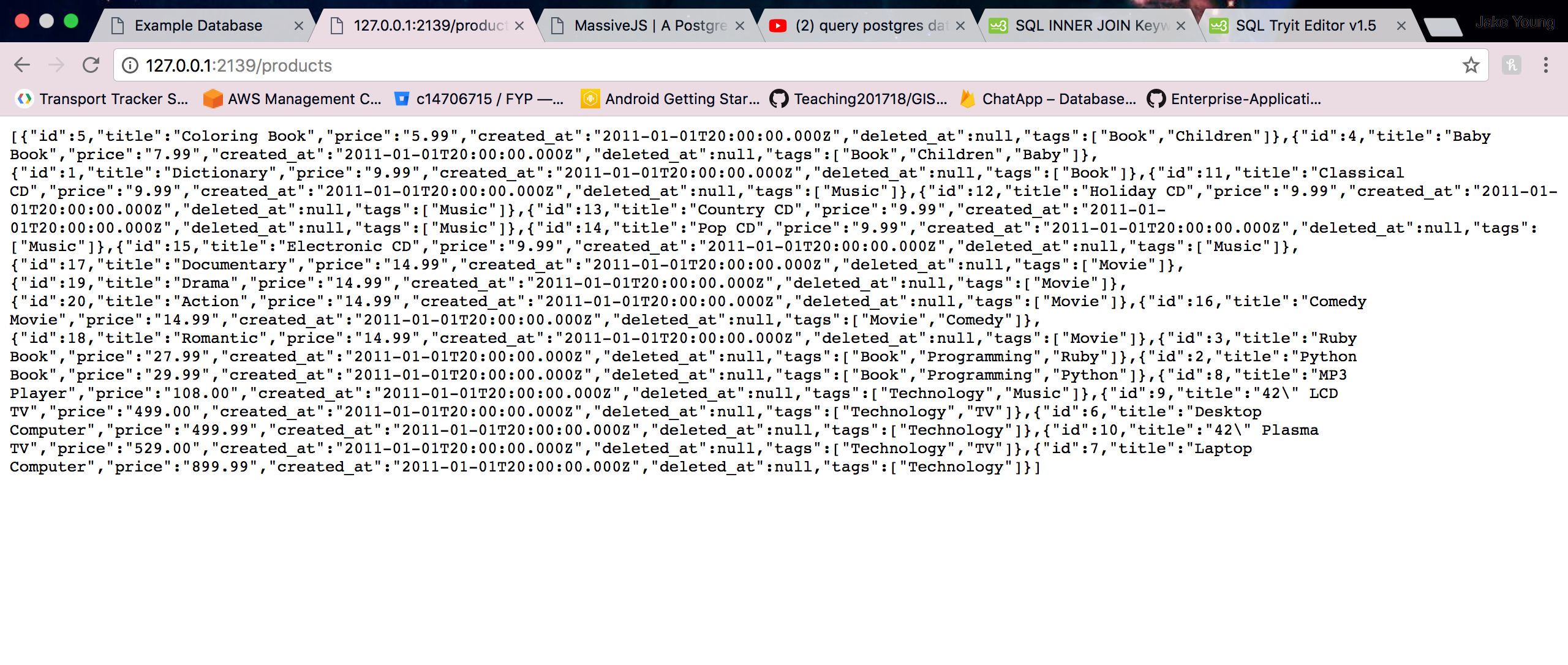
app.get('/products', function(req, res) {

db.run("select \* from products order by price asc",function(err, result) {

res.send(result);

});

});



/\*

Question 1

Part 4:

Show details of the specified products

\*/

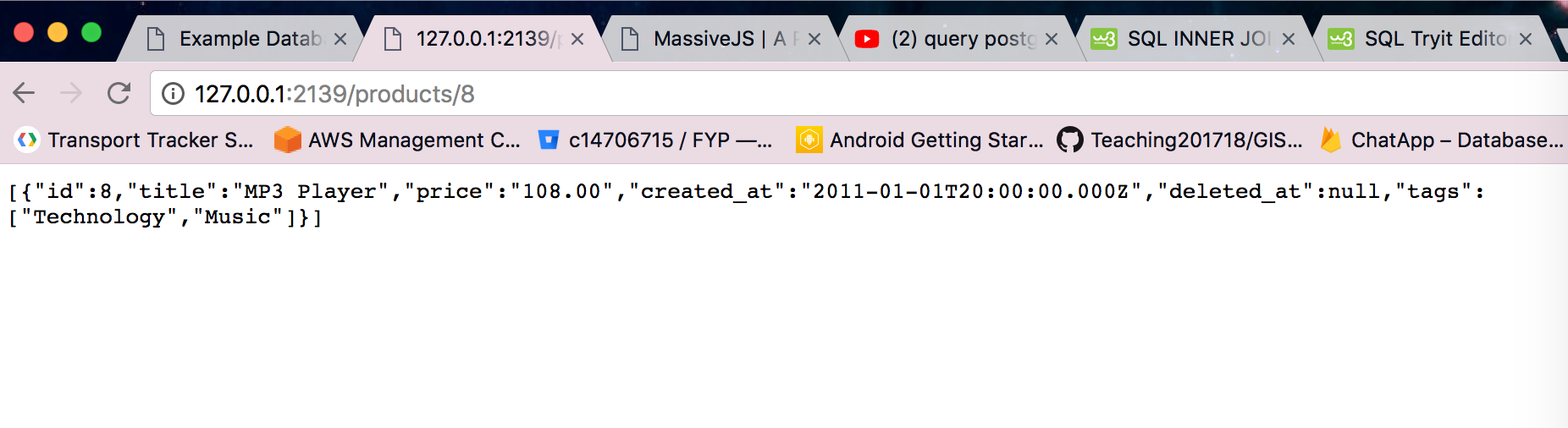
app.get('/products/:id', function(req, res) {

let id = req.params.id;

db.products.find({id: id}, function (err, result) {

res.send(result)

});

});

/\*

Question 1

Part 5:

List purchase items to include the receiver’s name and, address, the purchaser’s email address and the price, quantity and delivery status of the purchased item. Order by price in descending order

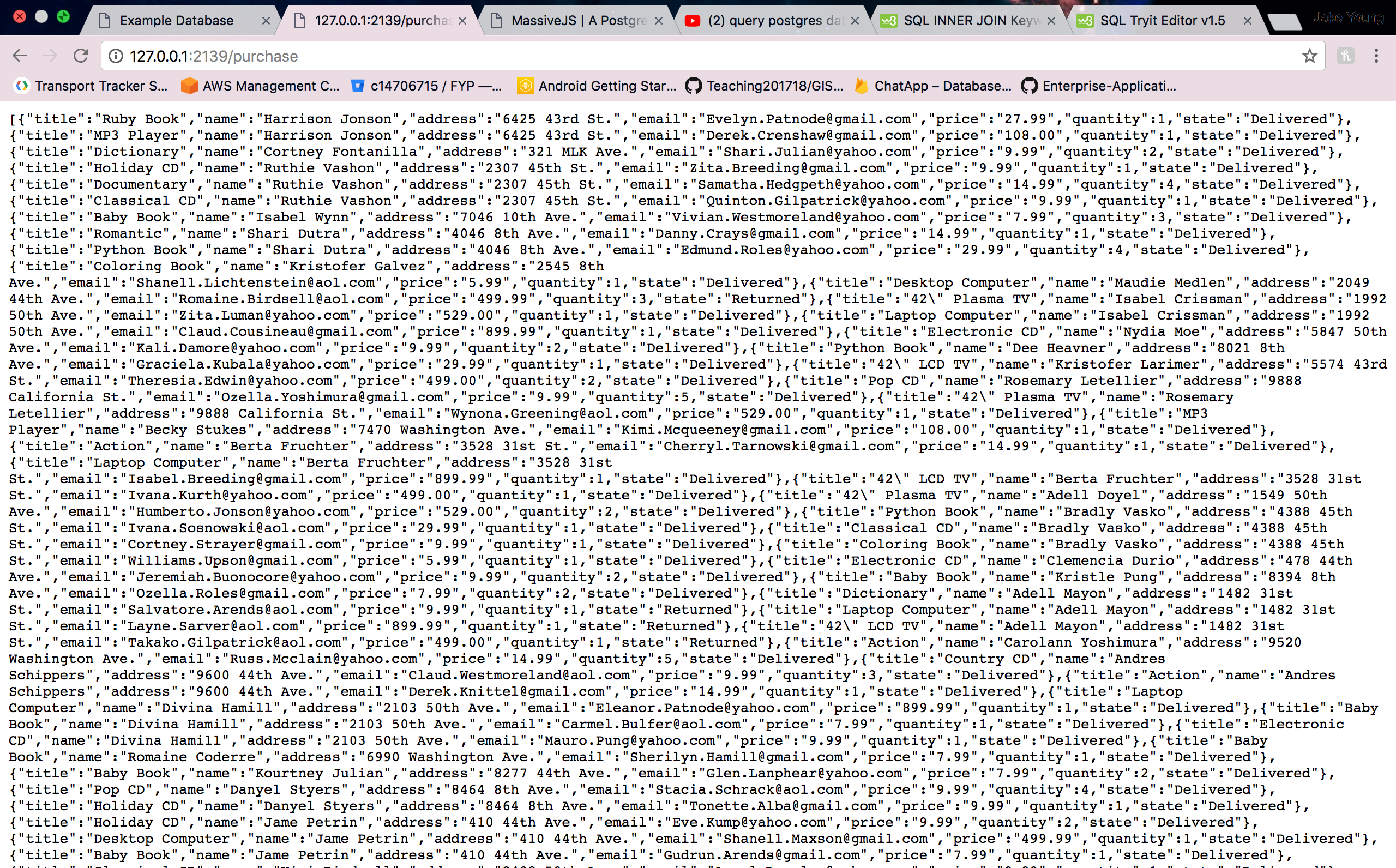
\*/

app.get('/purchase', function(req, res) {

db.run("select title, name, address, email, purchase\_items.price, quantity, purchase\_items.state from products join purchase\_items on products.id = purchase\_items.product\_id join purchases on purchase\_items.purchase\_id = purchases.id join users on purchase\_items.id = users.id",function(err, result) {

res.send(result);

});

});

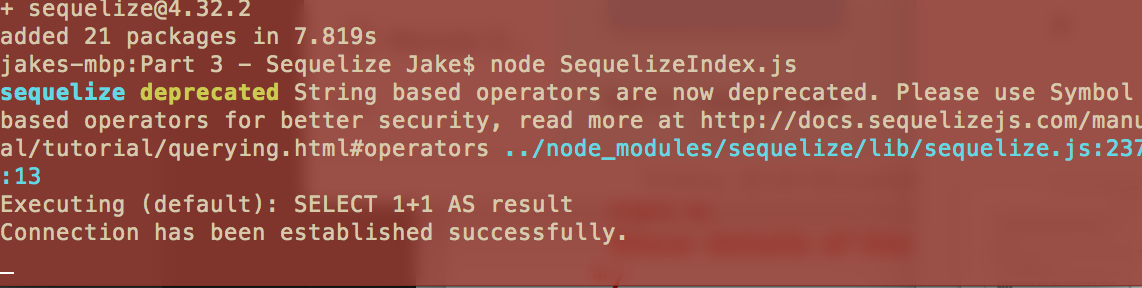
/\*

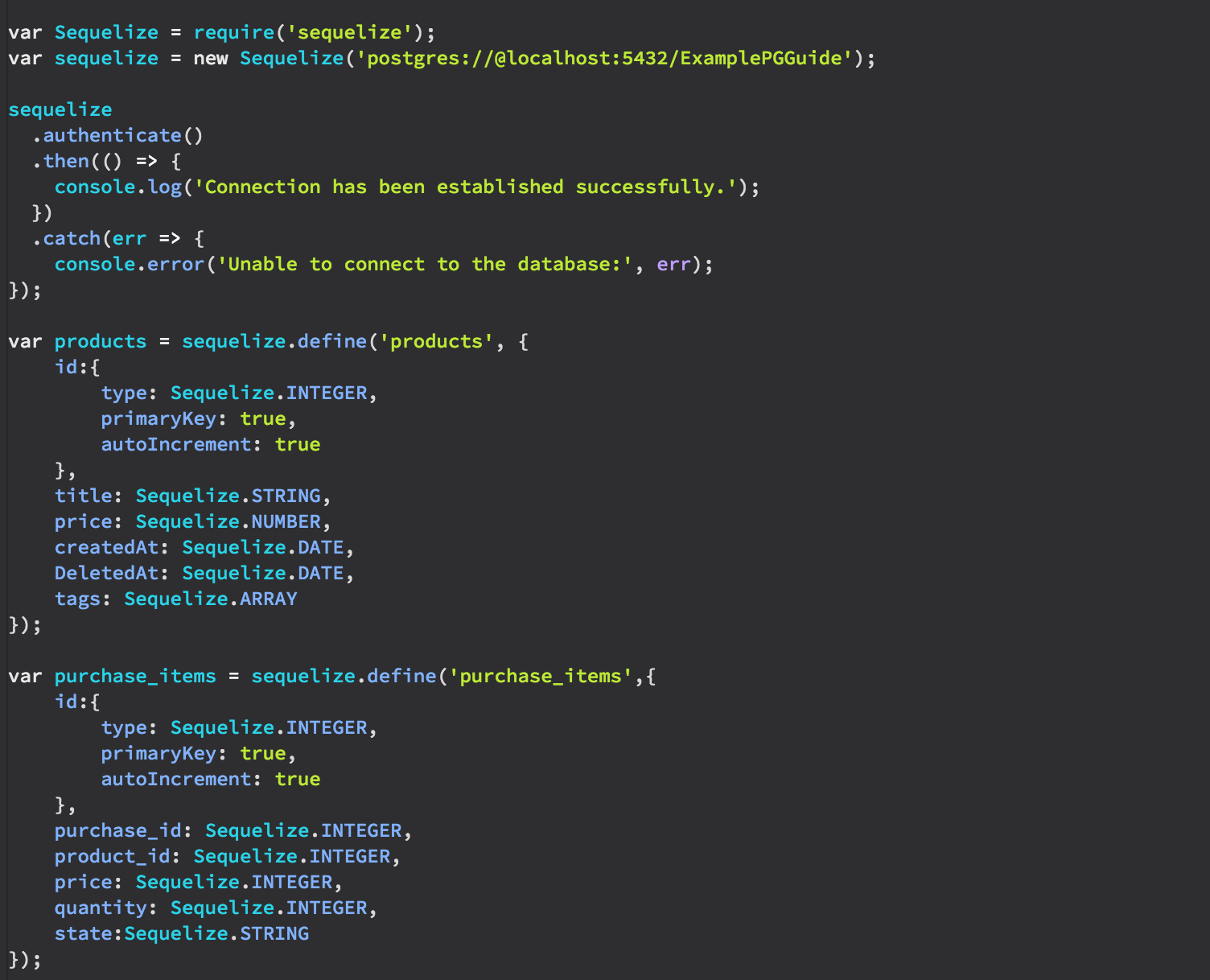
Question 4

Create a brand new Express project using the Sequelize ORM. Install and configure Sequelize and wire it up to the pgguide database. Verify that you have basic connectivity before proceeding.

Create Sequelize migrations for the pgguide sample database

Ensure that the appropriate associations and referential integrity checking are set up in your models

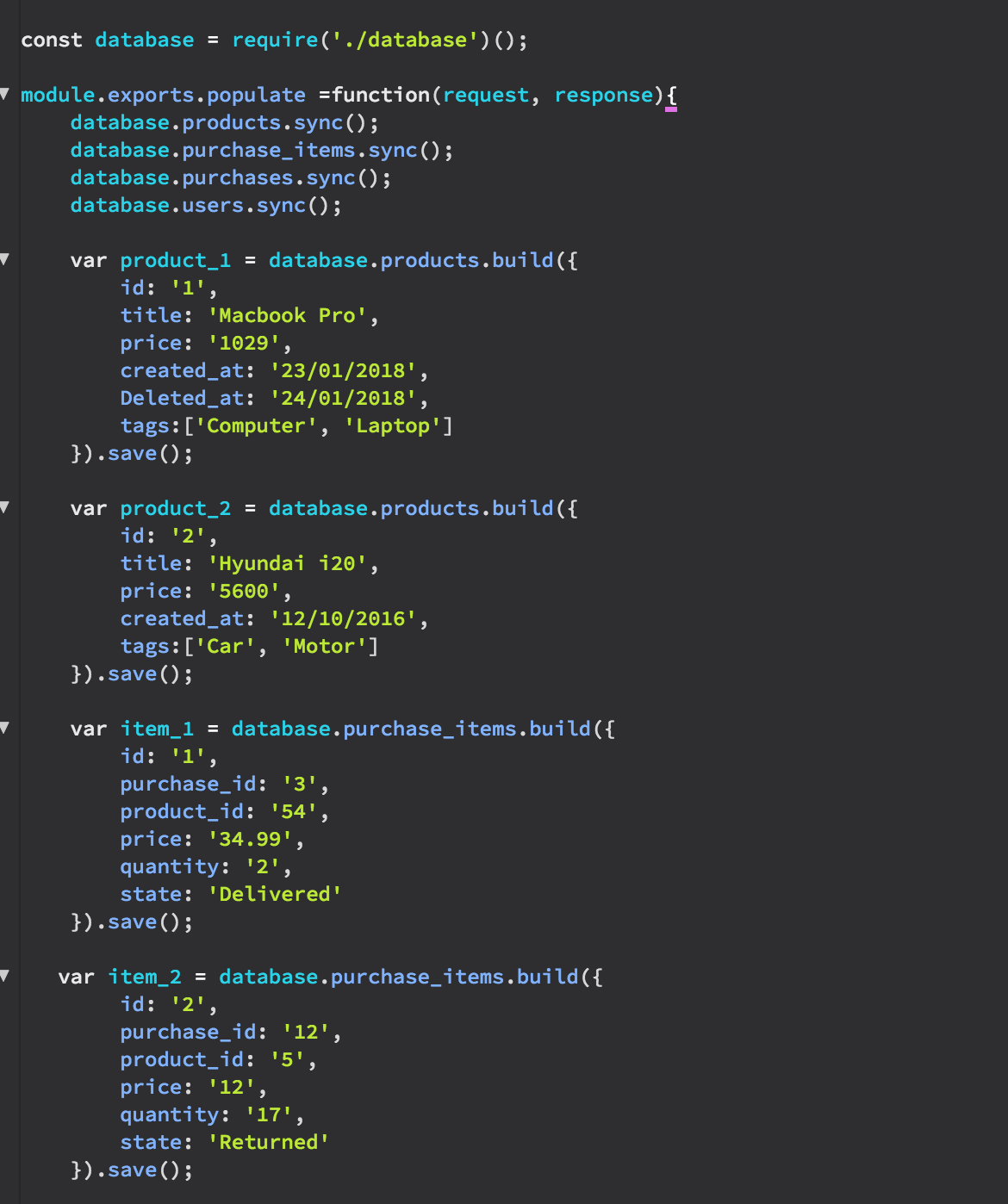
\*/ 



/\*

Question 5

Use your models and Javascript code to populate the database with some additional test data for all of the models above

\*/



/\*

Question 6

Reimplement the RESTful API using Sequelize and Express for your system. Your API should support the following CRUD operations as follows, returning JSON responses

|  |  |
| --- | --- |
| GET /products[?name=string] | List all products |
| GET /products/:id | Show details of the specified products |
| POST /products | Create a new product instance |
| PUT /products/:id | Update an existing product |
| DELETE /products/:id | Remove an existing product |

Show test cases for each of the API endpoint REST operations

\*/