

REST API, SQL and ORM

Code for parts 1 ,2 and 3

```
1  const express = require('express');
2  const http = require('http');
3  const massive = require('massive');
4  var hstore = require('pg-hstore')();
5
6  const app = express()
7  const port = 3000
8
9
10 massive({
11   host: '127.0.0.1',
12   port: 5432,
13   database: 'pgguide',
14   user: 'pgpaul',
15   password: 'password'
16 }).then(instance => {
17   app.set('db', instance);
18
19   /*******
20   app.get('/', (req, res) => res.send('Lab1'))
21   /*******
22
23   // Get all users
24   ..app.get("/users", (req, res) => {
25     .. instance.query("SELECT email, details->'sex' as sex FROM users ORDER BY created_at
26       DESC").then(query => {
27       res.json(query);
28     })
29   });
30
31   /*******
32
33   // Get a specific user
34   app.get("/users/:id", (req, res) => {
35     instance.query("select email, details->'sex' as sex from users where id = ${id}
36       order by created_at DESC",
37     {id: req.params.id}).then(query => {
38       res.json(query);
39     })
40   });
41
42   /*******
```

```

//*****

// Get all products
// app.get('/products', (req, res) => {
//   instance.query( "Select * from products order by price ASC").then(items => {
//     res.json(items);
//   });
// });

// -----
// Inject arbitrary SQL code into query execution
// -----

//http://127.0.0.1:3000/products?name=p
//http://127.0.0.1:3000/products?name=p%27%3B%20SELECT%20*%20FROM%20purchase_items%3B%20--
//http://127.0.0.1:3000/products?
//   name=p%27%3BDELETE%20FROM%products%20WHERE%20product_id=1000%3B%20-- ;

app.get('/products', (req, res) => {
  search = (req.query.name == undefined) ? '' : req.query.name;
  instance.query("SELECT * FROM products WHERE LOWER(title) LIKE '" + search + "%' " +
    "ORDER BY price ASC").then(items => {
    res.json(items);
  });
});

// -----
//Using a parameterised query to eliminate the SQL Injection possibilities
// -----
//   app.get('/products', (req, res) => {
//     search = (req.query.name == undefined) ? '' : req.query.name;
//     instance.query("SELECT * FROM products WHERE LOWER(title) LIKE $1 " +
//       "ORDER BY price ASC", [ search + "%" ]).then(items => {
//       res.json(items);
//     });
//   });

//Using a stored procedure to eliminate the SQL Injection
//http://127.0.0.1:3000/products?name=p
//http://127.0.0.1:3000/products?name=p%27%3B%20SELECT%20*%20FROM%20users%3B%20--
//http://127.0.0.1:3000/products?name=p'; DELETE FROM purchase_items WHERE product_id =
//   1222;
// app.get("/products", (req, res) => {
//   instance.query('select * from products(${title})', {title: req.query.name}).then(query
//     => {
//       res.json(query);
//     });
// });

```

```

//*****

·// Get product by id
app.get("/products/:id", (req, res) => {
  instance.query("select * from products where id = ${id}",
    {id: req.params.id}).then(query => {
      res.json(query);
    })
});

//*****

// Get purchases
app.get("/purchases", (req, res) => {
  instance.query("select purchases.name, purchases.address, users.email,
    purchase_items.price, purchase_items.quantity,"
    + " purchase_items.state from purchases join users on purchases.user_id = users.id
    "
    + "join purchase_items on purchases.id = purchase_items.purchase_id order by
    purchase_items.price DESC").then(query => {
      res.json(query);
    })
});

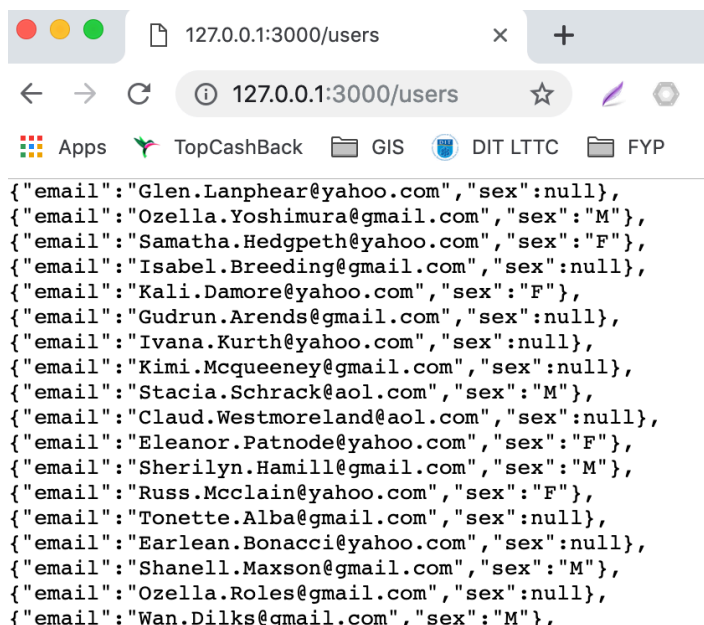
//*****

app.listen(port, () => console.log(`Example app listening on port ${port}!`))

});

```

Get all users



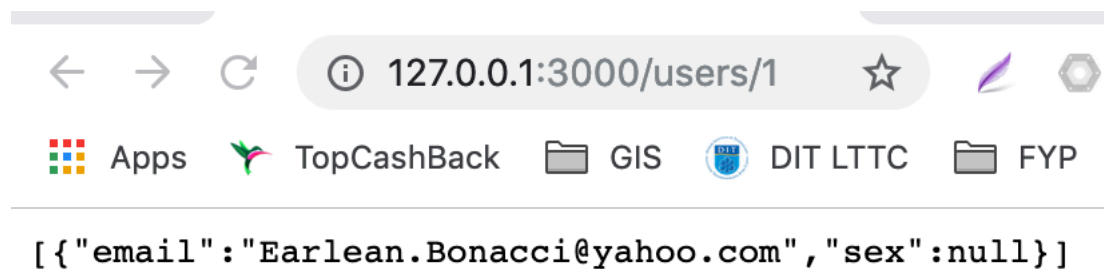
The screenshot shows a web browser window with the address bar displaying "127.0.0.1:3000/users". The page content is a JSON array of 20 user objects, each containing an "email" and a "sex" field. The browser's taskbar at the bottom shows several open applications: "Apps", "TopCashBack", "GIS", "DIT LTTC", and "FYP".

```

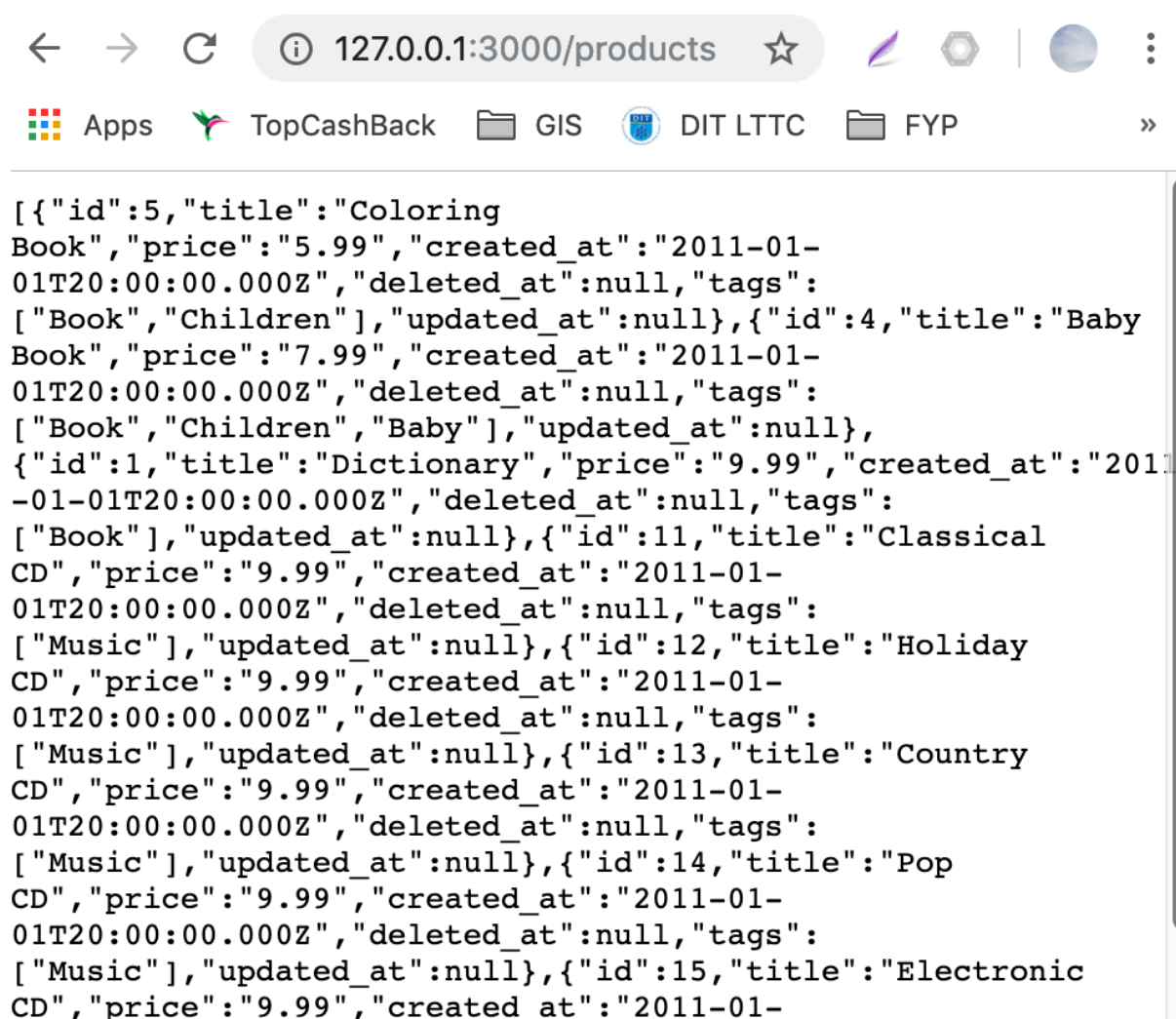
{"email": "Glen.Lanphear@yahoo.com", "sex": null},
{"email": "Ozella.Yoshimura@gmail.com", "sex": "M"},
{"email": "Samatha.Hedgpeth@yahoo.com", "sex": "F"},
{"email": "Isabel.Breeding@gmail.com", "sex": null},
{"email": "Kali.Damore@yahoo.com", "sex": "F"},
{"email": "Gudrun.Arends@gmail.com", "sex": null},
{"email": "Ivana.Kurth@yahoo.com", "sex": null},
{"email": "Kimi.Mcqueeney@gmail.com", "sex": null},
{"email": "Stacia.Schrack@aol.com", "sex": "M"},
{"email": "Claud.Westmoreland@aol.com", "sex": null},
{"email": "Eleanor.Patnode@yahoo.com", "sex": "F"},
{"email": "Sherilyn.Hamill@gmail.com", "sex": "M"},
{"email": "Russ.Mcclain@yahoo.com", "sex": "F"},
{"email": "Tonette.Alba@gmail.com", "sex": null},
{"email": "Earlean.Bonacci@yahoo.com", "sex": null},
{"email": "Shanell.Maxson@gmail.com", "sex": "M"},
{"email": "Ozella.Roles@gmail.com", "sex": null},
{"email": "Wan.Dilks@gmail.com", "sex": "M"},

```

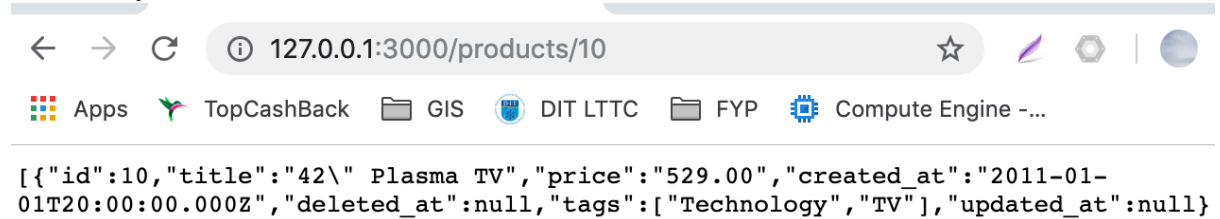
User. by id :



All products

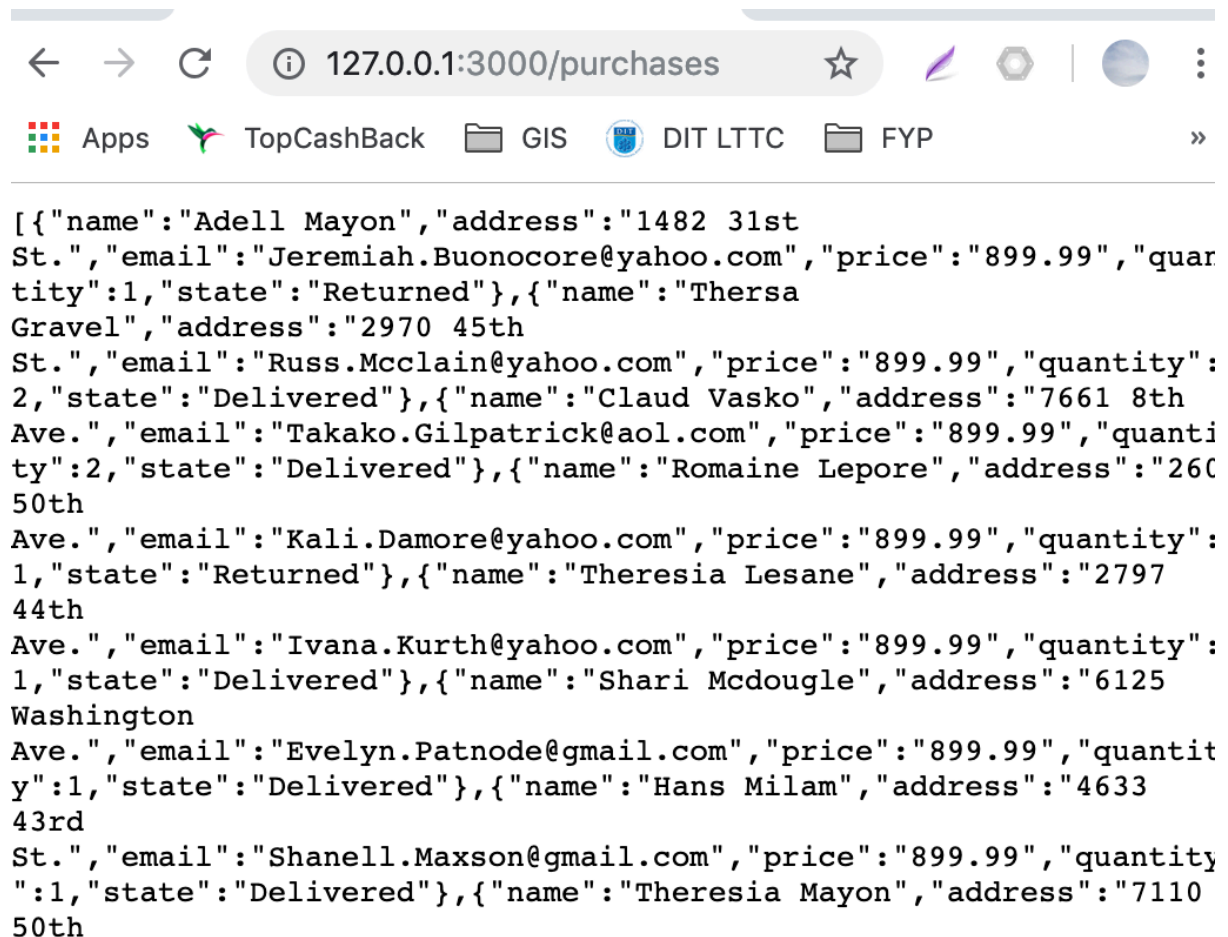


Product by ID



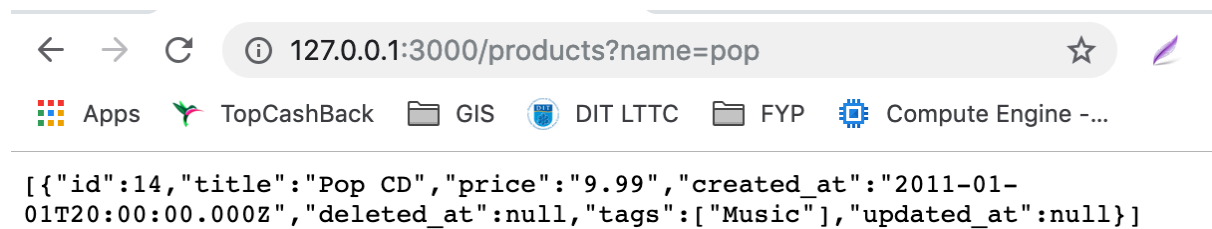
```
[{"id":10,"title":"42\" Plasma TV","price":"529.00","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology","TV"],"updated_at":null}]
```

Purchases

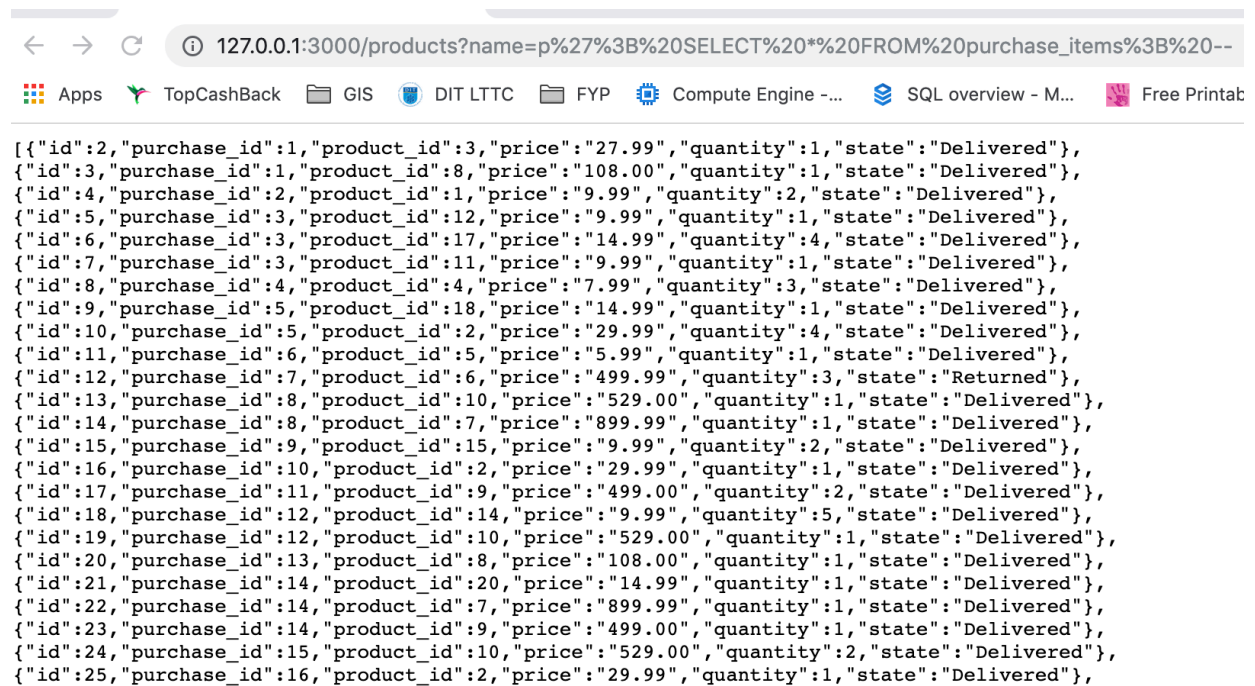


```
[{"name":"Adell Mayon","address":"1482 31st St. ","email":"Jeremiah.Buonocore@yahoo.com","price":"899.99","quantity":1,"state":"Returned"}, {"name":"Thersa Gravel","address":"2970 45th St. ","email":"Russ.Mcclain@yahoo.com","price":"899.99","quantity":2,"state":"Delivered"}, {"name":"Claud Vasko","address":"7661 8th Ave. ","email":"Takako.Gilpatrick@aol.com","price":"899.99","quantity":2,"state":"Delivered"}, {"name":"Romaine Lepore","address":"260 50th Ave. ","email":"Kali.Damore@yahoo.com","price":"899.99","quantity":1,"state":"Returned"}, {"name":"Theresia Lesane","address":"2797 44th Ave. ","email":"Ivana.Kurth@yahoo.com","price":"899.99","quantity":1,"state":"Delivered"}, {"name":"Shari Mcdougale","address":"6125 Washington Ave. ","email":"Evelyn.Patnode@gmail.com","price":"899.99","quantity":1,"state":"Delivered"}, {"name":"Hans Milam","address":"4633 43rd St. ","email":"Shanell.Maxson@gmail.com","price":"899.99","quantity":1,"state":"Delivered"}, {"name":"Theresia Mayon","address":"7110 50th"}]
```

Part 2 – get products by name



SQL injection get all data from purchases table



SQL injection to delete data

```
pggguide=# select * from products;
```

id	title	price	created_at	deleted_at	tags	updated_at
1	Dictionary	9.99	2011-01-01 20:00:00+00		{Book}	
3	Ruby Book	27.99	2011-01-01 20:00:00+00		{Book,Programming,Ruby}	
4	Baby Book	7.99	2011-01-01 20:00:00+00		{Book,Children,Baby}	
5	Coloring Book	5.99	2011-01-01 20:00:00+00		{Book,Children}	
6	Desktop Computer	499.99	2011-01-01 20:00:00+00		{Technology}	
7	Laptop Computer	899.99	2011-01-01 20:00:00+00		{Technology}	
8	MP3 Player	108.00	2011-01-01 20:00:00+00		{Technology,Music}	
9	42" LCD TV	499.00	2011-01-01 20:00:00+00		{Technology,TV}	
10	42" Plasma TV	529.00	2011-01-01 20:00:00+00		{Technology,TV}	
11	Classical CD	9.99	2011-01-01 20:00:00+00		{Music}	
12	Holiday CD	9.99	2011-01-01 20:00:00+00		{Music}	
13	Country CD	9.99	2011-01-01 20:00:00+00		{Music}	
14	Pop CD	9.99	2011-01-01 20:00:00+00		{Music}	
15	Electronic CD	9.99	2011-01-01 20:00:00+00		{Music}	
16	Comedy Movie	14.99	2011-01-01 20:00:00+00		{Movie,Comedy}	
17	Documentary	14.99	2011-01-01 20:00:00+00		{Movie}	
18	Romantic	14.99	2011-01-01 20:00:00+00		{Movie}	
19	Drama	14.99	2011-01-01 20:00:00+00		{Movie}	
20	Action	14.99	2011-01-01 20:00:00+00		{Movie}	
21	television	228.99	2019-02-17 16:16:26.275+00		{tv,television}	
22	television	228.99	2019-02-17 16:41:48.185+00		{tv,television}	2019-02-17
23	television	228.99	2019-02-17 16:42:07.882+00		{tv,television}	2019-02-17
24			2019-02-17 17:08:59.053+00			2019-02-17
2	Python Book	29.99	2011-01-01 20:00:00+00		{Book,Programming,Python}	2019-02-17

<http://127.0.0.1:3000/products?name=p%27;%20DELETE%20FROM%20products%20WHERE%20id%20=%2022>

We can see product with id 22 has been removed

1	Dictionary	9.99	2011-01-01 20:00:00+00		{Book}	
3	Ruby Book	27.99	2011-01-01 20:00:00+00		{Book,Programming,Ruby}	
4	Baby Book	7.99	2011-01-01 20:00:00+00		{Book,Children,Baby}	
5	Coloring Book	5.99	2011-01-01 20:00:00+00		{Book,Children}	
6	Desktop Computer	499.99	2011-01-01 20:00:00+00		{Technology}	
7	Laptop Computer	899.99	2011-01-01 20:00:00+00		{Technology}	
8	MP3 Player	108.00	2011-01-01 20:00:00+00		{Technology,Music}	
9	42" LCD TV	499.00	2011-01-01 20:00:00+00		{Technology,TV}	
10	42" Plasma TV	529.00	2011-01-01 20:00:00+00		{Technology,TV}	
11	Classical CD	9.99	2011-01-01 20:00:00+00		{Music}	
12	Holiday CD	9.99	2011-01-01 20:00:00+00		{Music}	
13	Country CD	9.99	2011-01-01 20:00:00+00		{Music}	
14	Pop CD	9.99	2011-01-01 20:00:00+00		{Music}	
15	Electronic CD	9.99	2011-01-01 20:00:00+00		{Music}	
16	Comedy Movie	14.99	2011-01-01 20:00:00+00		{Movie,Comedy}	
17	Documentary	14.99	2011-01-01 20:00:00+00		{Movie}	
18	Romantic	14.99	2011-01-01 20:00:00+00		{Movie}	
19	Drama	14.99	2011-01-01 20:00:00+00		{Movie}	
20	Action	14.99	2011-01-01 20:00:00+00		{Movie}	
21	television	228.99	2019-02-17 16:16:26.275+00		{tv,television}	
23	television	228.99	2019-02-17 16:42:07.882+00		{tv,television}	2019-02-17
24			2019-02-17 17:08:59.053+00			2019-02-17
2	Python Book	29.99	2011-01-01 20:00:00+00		{Book,Programming,Python}	2019-02-17

(23 rows)

- Using a parameterised query

```
//Using a parameterised query to eliminate the SQL Injection possibilities
// -----
app.get('/products', (req, res) => {
  search = (req.query.name == undefined) ? '' : req.query.name;
  instance.query("SELECT * FROM products WHERE LOWER(title) LIKE $1 " +
    "ORDER BY price ASC", [ search + "%" ]).then(items => {
    res.json(items);
  });
});
```

- Using a stored procedure using PLPGSQL

```
1 //run this function in Postgres
2 CREATE OR REPLACE FUNCTION products(IN t_title VARCHAR(255))
3 RETURNS setof products
4 AS $BODY$
5 BEGIN RETURN QUERY
6 SELECT * FROM PRODUCTS WHERE TITLE LIKE '%' || t_title || '%' ORDER BY PRICE ASC; END;
7 $BODY$ LANGUAGE plpgsql;
8
```

```
•//Using a stored procedure to eliminate the SQL Injection
//http://127.0.0.1:3000/products?name=p
//http://127.0.0.1:3000/products?name=p%27%3B%20SELECT%20*%20FROM%20users%3B%20--
//http://127.0.0.1:3000/products?name=p'; DELETE FROM purchase_items WHERE product_id =
22;
app.get("/products", (req, res) => {
  instance.query('select * from products(${title})', {title: req.query.name}).then(query
    => {
      res.json(query);
    });
});
});
```


4. New Sequelize project

Create sequelize models

```
//Products
node_modules/.bin/sequelize model:generate --name products --attributes
title:string,price:float,created_at:date,deleted_at:date,tags:array:string --underscored
//Purchase items
node_modules/.bin/sequelize model:generate --name purchase_items --attributes
purchase_id:integer,product_id:integer,price:float,quantity:integer,state:string --
underscored
//Purchases
node_modules/.bin/sequelize model:generate --name purchases --attributes
created_at:date,name:string,address:string,state:string,zipcode:integer,user_id:integer --
underscored
//Users
node_modules/.bin/sequelize model:generate --name users --attributes
email:string,password:string,details:array:string,created_at:date,deleted_at:date --
underscored
```

Migrate new models and add new data using commands:

```
node_modules/.bin/sequelize db:migrate
```

Create and run seeds:

```
node_modules/.bin/sequelize seed:generate --name demo-user
node_modules/.bin/sequelize seed:generate --name demo-products
node_modules/.bin/sequelize seed:generate --name demo-purchaseitems
node_modules/.bin/sequelize seed:generate --name demo-purchases
```

```
node_modules/.bin/sequelize db:seed:all
```

List of relations			
Schema	Name	Type	Owner
public	SequelizeMeta	table	pgpaul
public	products	table	paulmac
public	purchase_items	table	paulmac
public	purchases	table	paulmac
public	users	table	paulmac
(5 rows)			

5. Insert data

New user

```
'use strict';

module.exports = {
  up: (queryInterface, Sequelize) => {
    return queryInterface.bulkInsert('users', [{
      email: 'Paul22@test.com',
      password: 'abcdef3efgh',
      created_at: new Date()
    }], {});
  },
  down: (queryInterface, Sequelize) => {
    return queryInterface.bulkDelete('users', { where: { email: 'Paul22@test.com' } }, {});
  }
};
```

49		Samatha.Pellegrin@yahoo.com		4e2a5f4b462636dbc7519bc49f841822				2009-03-25 20:17:00+00
50		Wan.Dilks@gmail.com		0650f5923e2abce41721d3d9ab37cc54		"sex"=>"M"		2009-10-08 23:43:00+01
51		Paul22@test.com		abcdef3efgh				2019-02-17 16:16:26.26+00
51	rows)							

New Product

```
'use strict';

module.exports = {
  up: (queryInterface, Sequelize) => {
    return queryInterface.bulkInsert('products', [{
      title: 'television',
      price: 228.99,
      tags: '{"tv", "television"}',
      created_at: new Date()
    }], {});
  },
  down: (queryInterface, Sequelize) => {
    return queryInterface.bulkDelete('products', { where: { title: 'television' } }, {});
  }
};
```

19		Drama		14.99		2011-01-01 20:00:00+00				{Movie}
20		Action		14.99		2011-01-01 20:00:00+00				{Movie}
21		television		228.99		2019-02-17 16:16:26.275+00				{tv,television}

New Purchase Item

```
'use strict';

module.exports = {
  up: (queryInterface, Sequelize) => {
    return queryInterface.bulkInsert('purchase_items', [{
      purchase_id: 1000,
      product_id: 1,
      price: 29.99,
      quantity: 999,
      state: 'Delivered'
    }], {});
  },

  down: (queryInterface, Sequelize) => {
    return queryInterface.bulkDelete('purchase_items', { where: { quantity: 999 } }, {});
  }
};
```

1458	999	16	14.99	1	Delivered
1459	1000	1	29.99	999	Delivered

New Purchase

```
'use strict';

module.exports = {
  up: (queryInterface, Sequelize) => {
    return queryInterface.bulkInsert('purchases', [{
      name: 'purchase101',
      address: 'main street',
      state: 'MH',
      zipcode: 11111,
      user_id: 1,
      created_at: new Date()
    }], {});
  },

  down: (queryInterface, Sequelize) => {
    return queryInterface.bulkDelete('purchases', { where: { name: 'purchase101' } }, {});
  }
};
```

1000	2011-12-03 20:36:00+00	Hiedi Heavner	5977 35th St.42nd Ave.	CO	14885	42
1001	2019-02-17 16:16:26.36+00	purchase101	main street	MH	11111	1

Part 6 :

Get all products

```
← → ↻ ⓘ 127.0.0.1:3000/products ☆ 🖋️ 🌐 🔄

📱 Apps 🌿 TopCashBack 📁 GIS 🇵🇪 DIT LTTC 📁 FYP ⚙️ Compute Engine -... 📊 SQL overview - M...

[{"id":1,"title":"Dictionary","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book"],"updated_at":null}, {"id":3,"title":"Ruby Book","price":"27.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book","Programming","Ruby"],"updated_at":null}, {"id":4,"title":"Baby Book","price":"7.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book","Children","Baby"],"updated_at":null}, {"id":5,"title":"Coloring Book","price":"5.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book","Children"],"updated_at":null}, {"id":6,"title":"Desktop Computer","price":"499.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology"],"updated_at":null}, {"id":7,"title":"Laptop Computer","price":"899.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology"],"updated_at":null}, {"id":8,"title":"MP3 Player","price":"108.00","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology","Music"],"updated_at":null}, {"id":9,"title":"42\ LCD TV","price":"499.00","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology","TV"],"updated_at":null}, {"id":10,"title":"42\ Plasma TV","price":"529.00","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Technology","TV"],"updated_at":null}, {"id":11,"title":"Classical CD","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Music"],"updated_at":null}, {"id":12,"title":"Holiday CD","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Music"],"updated_at":null}, {"id":13,"title":"Country CD","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Music"],"updated_at":null}, {"id":14,"title":"Pop CD","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Music"],"updated_at":null}, {"id":15,"title":"Electronic CD","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Music"],"updated_at":null}, {"id":16,"title":"Comedy Movie","price":"14.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Movie","Comedy"],"updated_at":null}, {"id":17,"title":"Documentary","price":"14.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Movie","Comedy"],"updated_at":null}]
```

Get products by name or ID

```
← → ↻ ⓘ 127.0.0.1:3000/products?name=Dictionary ☆ 🖋️ 🌐 🔄

📱 Apps 🌿 TopCashBack 📁 GIS 🇵🇪 DIT LTTC 📁 FYP ⚙️ Compute Engine -...

[{"id":1,"title":"Dictionary","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book"],"updated_at":null}]
```

```
🔴 🟡 🟢 127.0.0.1:3000/products/1 × +

← → ↻ ⓘ 127.0.0.1:3000/products/1 📱 Apps 🌿 TopCashBack 📁 GIS 🇵🇪 DIT LTTC 📁 FYP ⚙️ Compute Engine -... 📊 SQL overview - M...

[{"id":1,"title":"Dictionary","price":"9.99","created_at":"2011-01-01T20:00:00.000Z","deleted_at":null,"tags":["Book"],"updated_at":null}]
```

POST : <http://127.0.0.1:3000/products?title=new Product>

```
pgguid=# select * from products;
```

id	title	price	created_at	deleted_at	tags	updated_at
1	Dictionary	9.99	2011-01-01 20:00:00+00		{Book}	
2	Python Book	29.99	2011-01-01 20:00:00+00		{Book,Programming,Python}	
3	Ruby Book	27.99	2011-01-01 20:00:00+00		{Book,Programming,Ruby}	
4	Baby Book	7.99	2011-01-01 20:00:00+00		{Book,Children,Baby}	
5	Coloring Book	5.99	2011-01-01 20:00:00+00		{Book,Children}	
6	Desktop Computer	499.99	2011-01-01 20:00:00+00		{Technology}	
7	Laptop Computer	899.99	2011-01-01 20:00:00+00		{Technology}	
8	MP3 Player	108.00	2011-01-01 20:00:00+00		{Technology,Music}	
9	42" LCD TV	499.00	2011-01-01 20:00:00+00		{Technology,TV}	
10	42" Plasma TV	529.00	2011-01-01 20:00:00+00		{Technology,TV}	
11	Classical CD	9.99	2011-01-01 20:00:00+00		{Music}	
12	Holiday CD	9.99	2011-01-01 20:00:00+00		{Music}	
13	Country CD	9.99	2011-01-01 20:00:00+00		{Music}	
14	Pop CD	9.99	2011-01-01 20:00:00+00		{Music}	
15	Electronic CD	9.99	2011-01-01 20:00:00+00		{Music}	
16	Comedy Movie	14.99	2011-01-01 20:00:00+00		{Movie,Comedy}	
17	Documentary	14.99	2011-01-01 20:00:00+00		{Movie}	
18	Romantic	14.99	2011-01-01 20:00:00+00		{Movie}	
19	Drama	14.99	2011-01-01 20:00:00+00		{Movie}	
20	Action	14.99	2011-01-01 20:00:00+00		{Movie}	
21	television	228.99	2019-02-17 16:16:26.275+00		{tv,television}	
22	television	228.99	2019-02-17 16:41:48.185+00		{tv,television}	2019-02-17
23	television	228.99	2019-02-17 16:42:07.882+00		{tv,television}	2019-02-17
24			2019-02-17 17:08:59.053+00			2019-02-17

(24 rows)

PUT – used postman we can see the product with id number 2 gets updated

PUT

<http://127.0.0.1:3000/products/2>

Send

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies (1)

KEY	VALUE	DESCRIPTION
<input type="checkbox"/> title	New Dict	
<input type="checkbox"/> Key	Value	Description

Body

Cookies (1)

Headers (6)

Test Results

Status: 200 OK

Time: 100 ms

Size: 219 B

Pretty

Raw

Preview

Product updated

```
pgguid=# select * from products;
```

id	title	price	created_at	deleted_at	tags	updated_at
1	Dictionary	9.99	2011-01-01 20:00:00+00		{Book}	
3	Ruby Book	27.99	2011-01-01 20:00:00+00		{Book,Programming,Ruby}	
4	Baby Book	7.99	2011-01-01 20:00:00+00		{Book,Children,Baby}	
5	Coloring Book	5.99	2011-01-01 20:00:00+00		{Book,Children}	
6	Desktop Computer	499.99	2011-01-01 20:00:00+00		{Technology}	
7	Laptop Computer	899.99	2011-01-01 20:00:00+00		{Technology}	
8	MP3 Player	108.00	2011-01-01 20:00:00+00		{Technology,Music}	
9	42" LCD TV	499.00	2011-01-01 20:00:00+00		{Technology,TV}	
10	42" Plasma TV	529.00	2011-01-01 20:00:00+00		{Technology,TV}	
11	Classical CD	9.99	2011-01-01 20:00:00+00		{Music}	
12	Holiday CD	9.99	2011-01-01 20:00:00+00		{Music}	
13	Country CD	9.99	2011-01-01 20:00:00+00		{Music}	
14	Pop CD	9.99	2011-01-01 20:00:00+00		{Music}	
15	Electronic CD	9.99	2011-01-01 20:00:00+00		{Music}	
16	Comedy Movie	14.99	2011-01-01 20:00:00+00		{Movie,Comedy}	
17	Documentary	14.99	2011-01-01 20:00:00+00		{Movie}	
18	Romantic	14.99	2011-01-01 20:00:00+00		{Movie}	
19	Drama	14.99	2011-01-01 20:00:00+00		{Movie}	
20	Action	14.99	2011-01-01 20:00:00+00		{Movie}	
21	television	228.99	2019-02-17 16:16:26.275+00		{tv,television}	
22	television	228.99	2019-02-17 16:41:48.185+00		{tv,television}	2019-02-17
23	television	228.99	2019-02-17 16:42:07.882+00		{tv,television}	2019-02-17
24			2019-02-17 17:08:59.053+00			2019-02-17
2	Python Book	29.99	2011-01-01 20:00:00+00		{Book,Programming,Python}	2019-02-17

DELETE – using Postman we can see product with id 22 has been removed from database.

DELETE

http://127.0.0.1:3000/products/22

Send

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies

Cod

	KEY	VALUE	DESCRIPTION	...
<input type="checkbox"/>	title	New Dict		
	Key	Value	Description	

Body

Cookies (1)

Headers (6)

Test Results

Status: 200 OK

Time: 55 ms

Size: 219 B

Pretty

Raw

Preview

Product deleted

1	Dictionary	9.99	2011-01-01 20:00:00+00	{Book}	
3	Ruby Book	27.99	2011-01-01 20:00:00+00	{Book,Programming,Ruby}	
4	Baby Book	7.99	2011-01-01 20:00:00+00	{Book,Children,Baby}	
5	Coloring Book	5.99	2011-01-01 20:00:00+00	{Book,Children}	
6	Desktop Computer	499.99	2011-01-01 20:00:00+00	{Technology}	
7	Laptop Computer	899.99	2011-01-01 20:00:00+00	{Technology}	
8	MP3 Player	108.00	2011-01-01 20:00:00+00	{Technology,Music}	
9	42" LCD TV	499.00	2011-01-01 20:00:00+00	{Technology,TV}	
10	42" Plasma TV	529.00	2011-01-01 20:00:00+00	{Technology,TV}	
11	Classical CD	9.99	2011-01-01 20:00:00+00	{Music}	
12	Holiday CD	9.99	2011-01-01 20:00:00+00	{Music}	
13	Country CD	9.99	2011-01-01 20:00:00+00	{Music}	
14	Pop CD	9.99	2011-01-01 20:00:00+00	{Music}	
15	Electronic CD	9.99	2011-01-01 20:00:00+00	{Music}	
16	Comedy Movie	14.99	2011-01-01 20:00:00+00	{Movie,Comedy}	
17	Documentary	14.99	2011-01-01 20:00:00+00	{Movie}	
18	Romantic	14.99	2011-01-01 20:00:00+00	{Movie}	
19	Drama	14.99	2011-01-01 20:00:00+00	{Movie}	
20	Action	14.99	2011-01-01 20:00:00+00	{Movie}	
21	television	228.99	2019-02-17 16:16:26.275+00	{tv,television}	
23	television	228.99	2019-02-17 16:42:07.882+00	{tv,television}	2019-02-17
24			2019-02-17 17:08:59.053+00		2019-02-17
2	Python Book	29.99	2011-01-01 20:00:00+00	{Book,Programming,Python}	2019-02-17
(23 rows)					