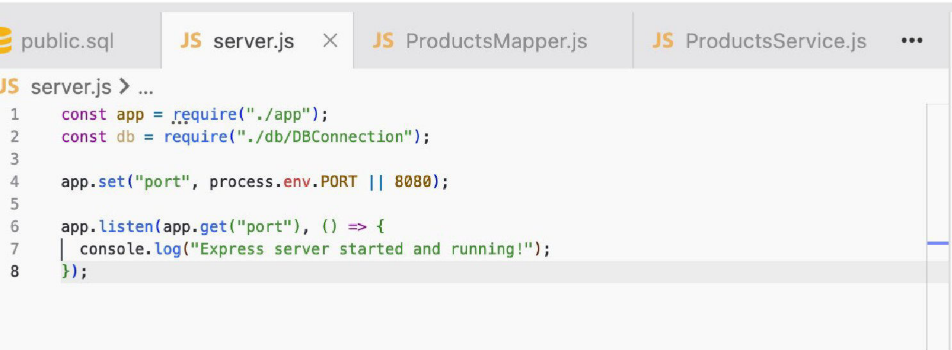
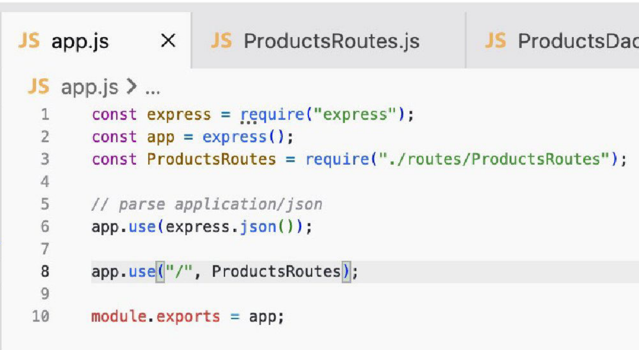
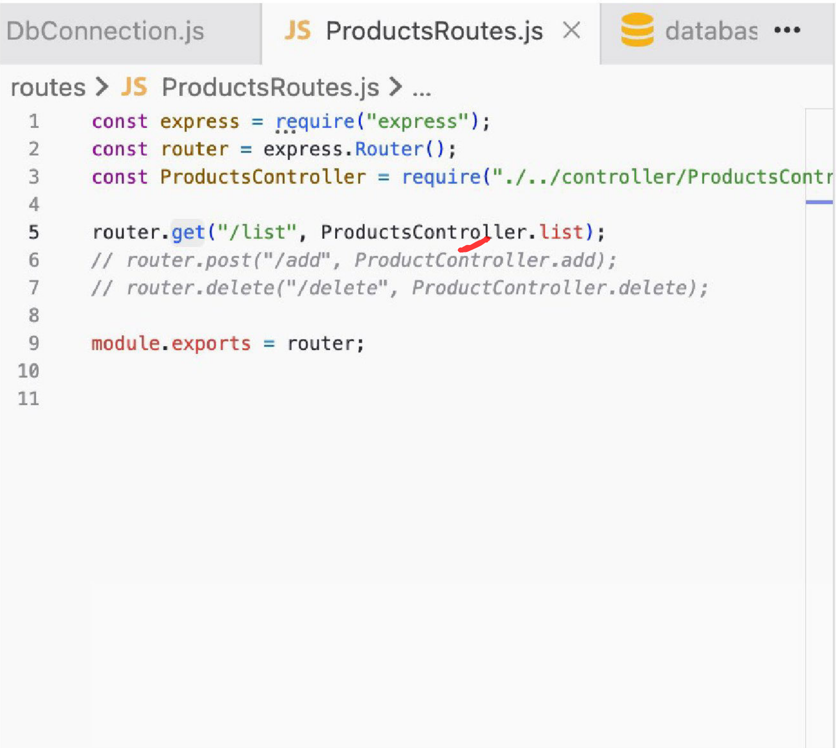


```js
const app = require("./app");
const db = require("./db/DBConnection");

app.set("port", process.env.PORT || 8080);

app.listen(app.get("port"), () => {
  console.log("Express server started and running!");
});
```

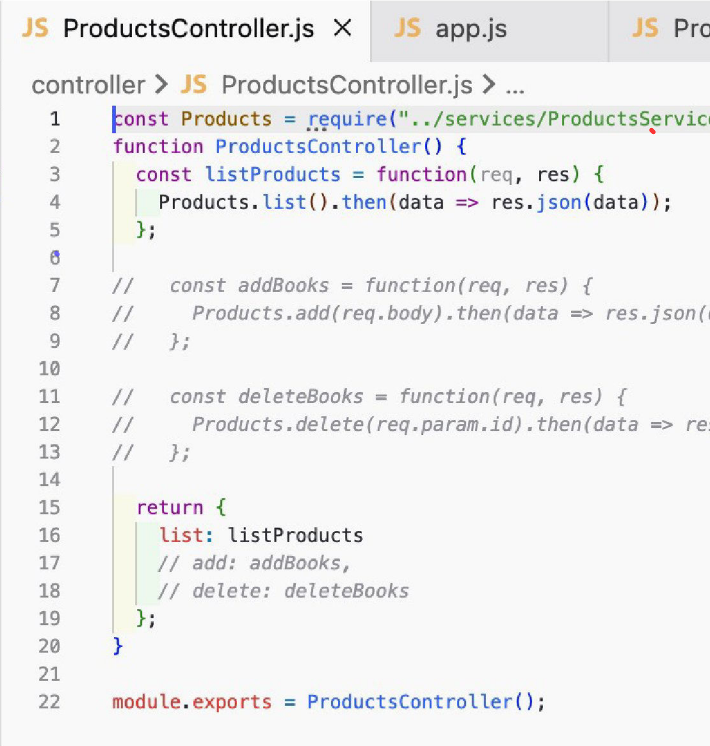


```js
const express = require("express");
const app = express();
const ProductsRoutes = require("./routes/ProductsRoutes");

// parse application/json
app.use(express.json());

app.use("/", ProductsRoutes);

module.exports = app;
```

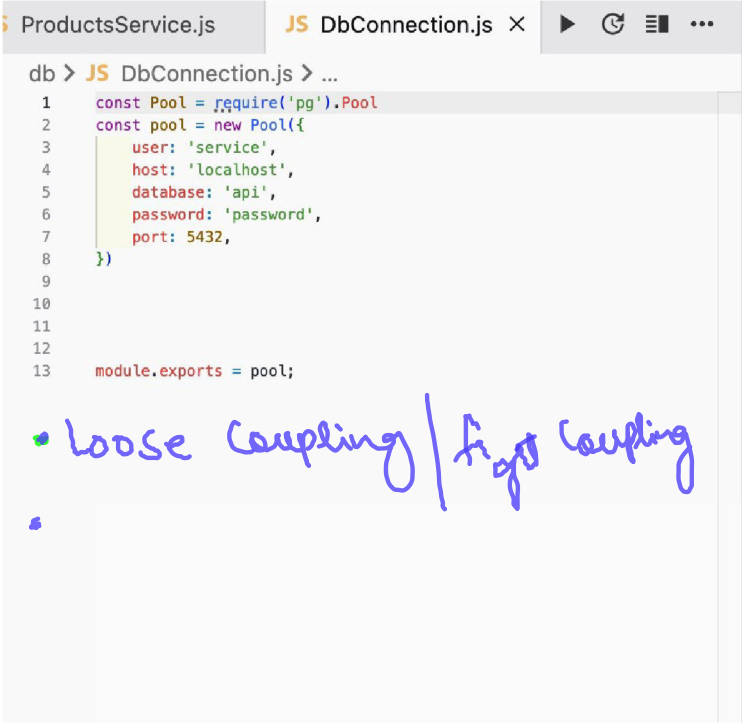
Uygi enco



```js
const express = require("express");
const router = express.Router();
const ProductsController = require("./../controller/ProductsContr

router.get("/list", ProductsController.list);
// router.post("/add", ProductController.add);
// router.delete("/delete", ProductController.delete);

module.exports = router;
```

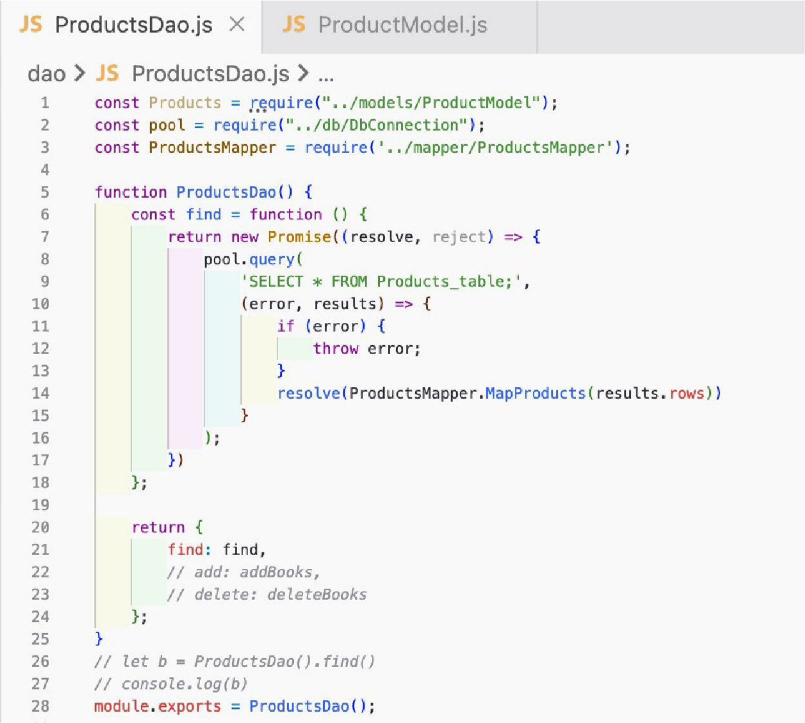


```js
const Products = require("../services/ProductsServic
function ProductsController() {
  const listProducts = function(req, res) {
    Products.list().then(data => res.json(data));
  };

//   const addBooks = function(req, res) {
//     Products.add(req.body).then(data => res.json(
//   };

//   const deleteBooks = function(req, res) {
//     Products.delete(req.param.id).then(data => re
//   };

  return {
    list: listProducts
    // add: addBooks,
    // delete: deleteBooks
  };
}

module.exports = ProductsController();
```

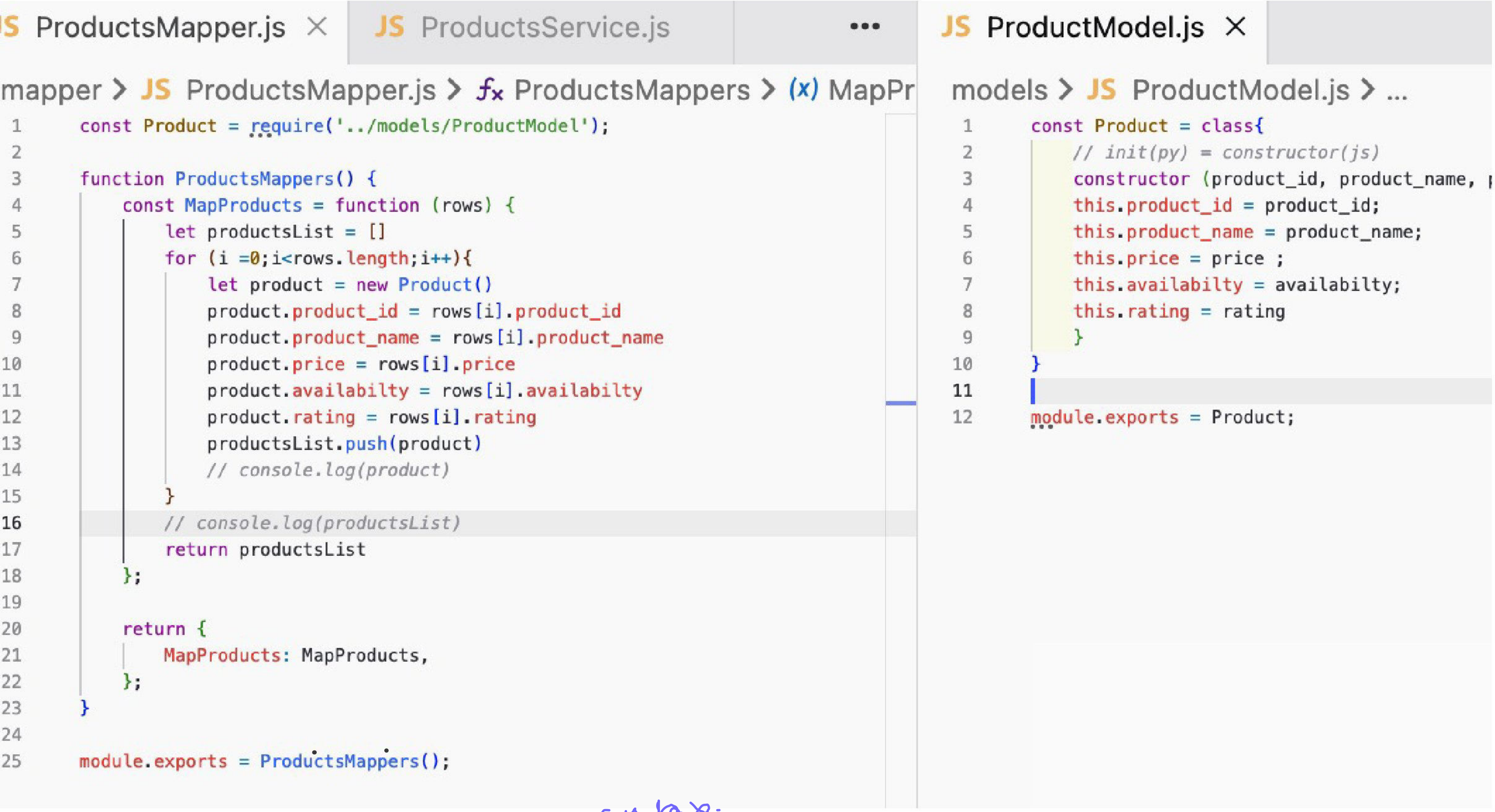


```js
const Pool = require('pg').Pool
const pool = new Pool({
    user: 'service',
    host: 'localhost',
    database: 'api',
    password: 'password',
    port: 5432,
})




module.exports = pool;
```

Loose Coupling | tight Coupling

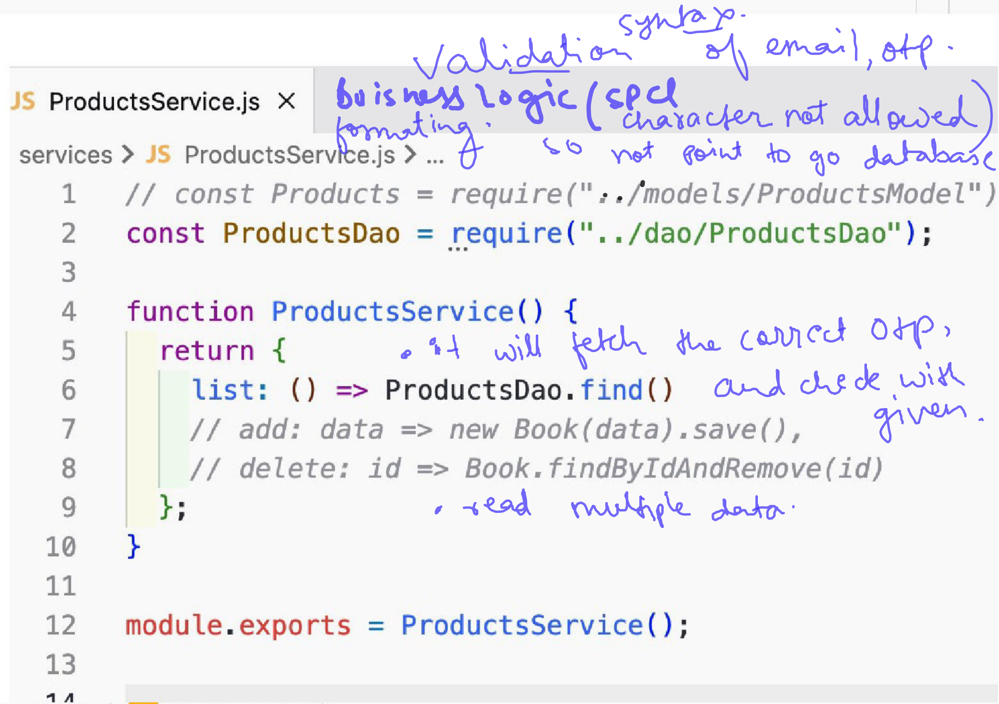


```js
const Products = require("../models/ProductModel");
const pool = require("../db/DbConnection");
const ProductsMapper = require('../mapper/ProductsMapper');

function ProductsDao() {
    const find = function () {
        return new Promise((resolve, reject) => {
            pool.query(
                'SELECT * FROM Products_table;',
                (error, results) => {
                    if (error) {
                        throw error;
                    }
                    resolve(ProductsMapper.MapProducts(results.rows))
                }
            );
        })
    };

    return {
        find: find,
        // add: addBooks,
        // delete: deleteBooks
    };
}
// let b = ProductsDao().find()
// console.log(b)
module.exports = ProductsDao();
```


ProductsMapper.js
ProductsService.js
ProductModel.js
mapper > ProductsMapper.js > ProductsMappers > MapPr
models > ProductModel.js > ...

```javascript
const Product = require('../models/ProductModel');

function ProductsMappers() {
    const MapProducts = function (rows) {
        let productsList = []
        for (i =0;i<rows.length;i++){
            let product = new Product()
            product.product_id = rows[i].product_id
            product.product_name = rows[i].product_name
            product.price = rows[i].price
            product.availabilty = rows[i].availabilty
            product.rating = rows[i].rating
            productsList.push(product)
            // console.log(product)
        }
        // console.log(productsList)
        return productsList
    };

    return {
        MapProducts: MapProducts,
    };
}

module.exports = ProductsMappers();
```

```javascript
const Product = class{
    // init(py) = constructor(js)
    constructor (product_id, product_name, 
    this.product_id = product_id;
    this.product_name = product_name;
    this.price = price ;
    this.availabilty = availabilty;
    this.rating = rating
    }
}

module.exports = Product;
```



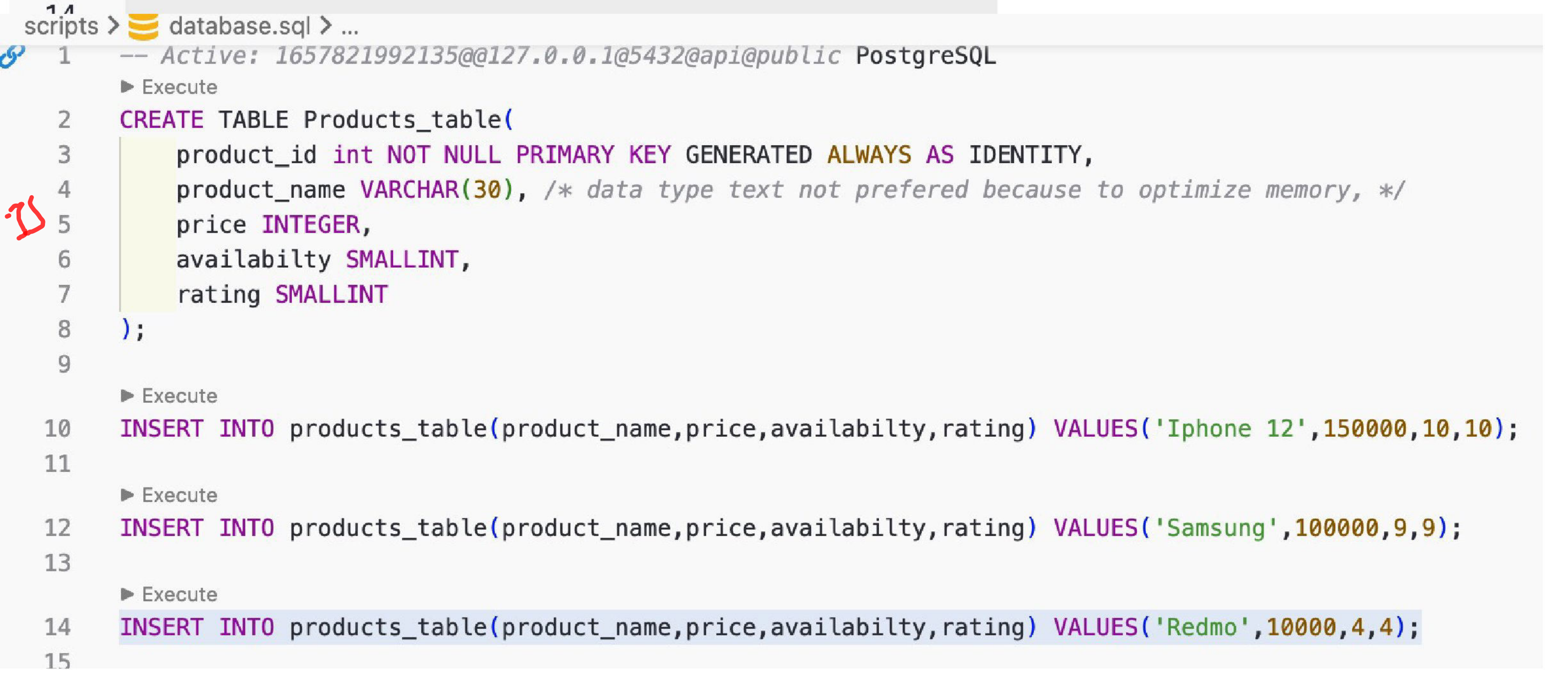
Validation syntax of email, otp.

buisness Logic (spcl character not allowed) formating. so not point to go database

```javascript
// const Products = require("../models/ProductsModel")
const ProductsDao = require("../dao/ProductsDao");

function ProductsService() {
  return {
    list: () => ProductsDao.find()
    // add: data => new Book(data).save(),
    // delete: id => Book.findByIdAndRemove(id)
  };
}

module.exports = ProductsService();
```

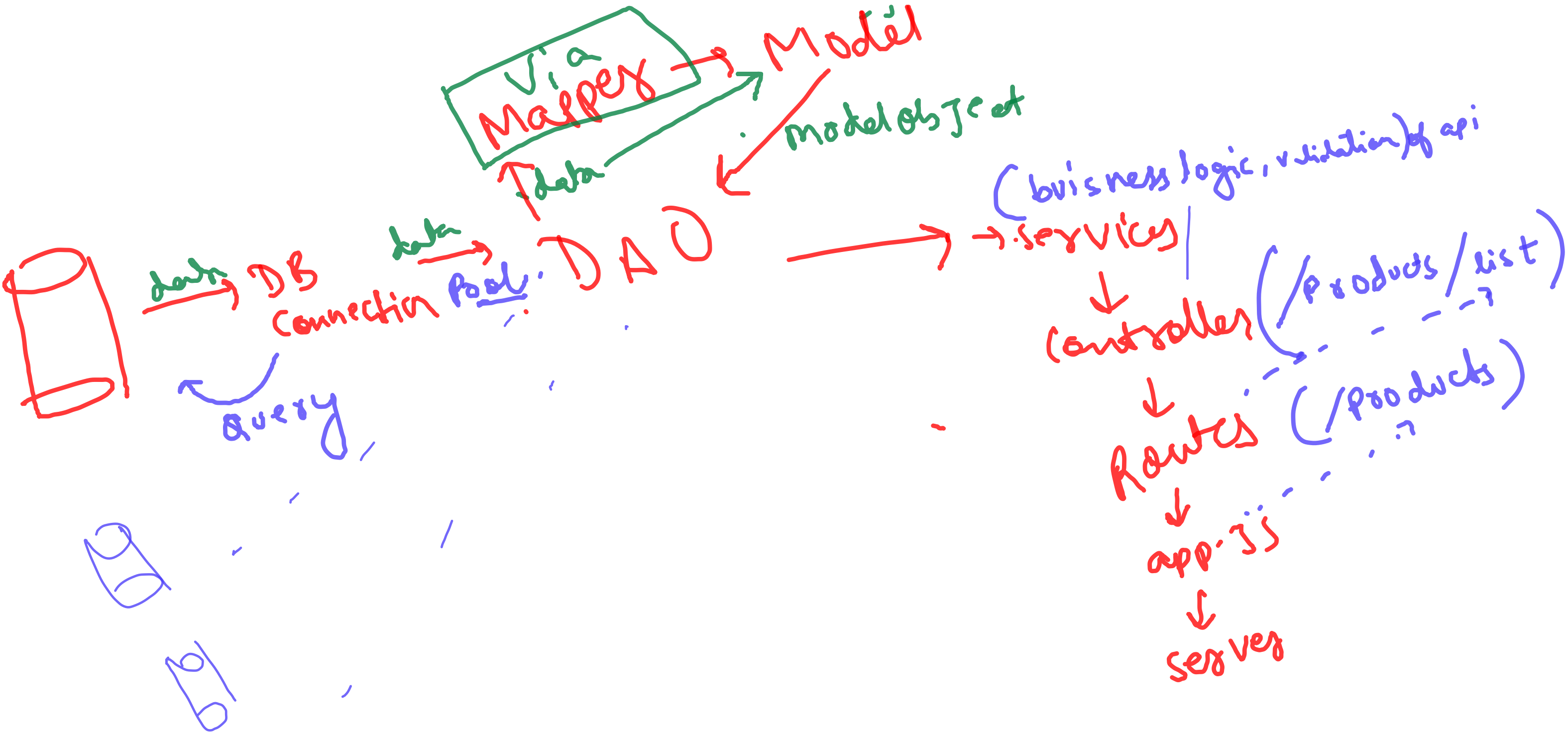
• it will fetch the correct otp, and check with given.

• read multiple data.



```sql
-- Active: 1657821992135@@127.0.0.1@5432@api@public PostgreSQL
CREATE TABLE Products_table(
    product_id int NOT NULL PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
    product_name VARCHAR(30), /* data type text not prefered because to optimize memory, */
    price INTEGER,
    availabilty SMALLINT,
    rating SMALLINT
);

INSERT INTO products_table(product_name,price,availabilty,rating) VALUES('Iphone 12',150000,10,10);

INSERT INTO products_table(product_name,price,availabilty,rating) VALUES('Samsung',100000,9,9);

INSERT INTO products_table(product_name,price,availabilty,rating) VALUES('Redmo',10000,4,4);
```

Microservice (no view)


Via
Mapper
Model
Model object
data
data
data
DB Connection Pool
DAO
query
(buisness logic, validation) of api
services
controller
(/products/list)
Routes
(/products)
app.js
server