



wawiwa

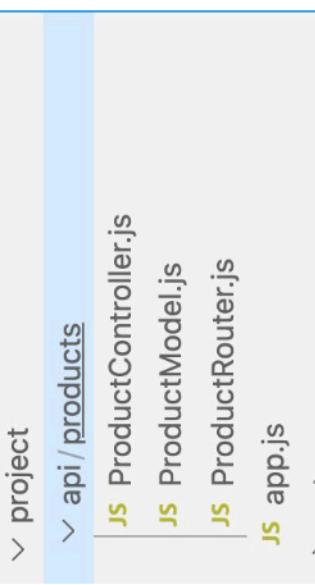
REFACTORIZANDO TU PROYECTO MONGOOSE NODE.JS

Refactorizar

La refactorización de código se define como el proceso de reestructurar el código de una computadora sin cambiar ni aumentar su comportamiento y funcionalidad externos. Hay muchas formas de refactorizar, pero la mayoría de las veces consiste en aplicar una serie de microrefactorizaciones básicas estandarizadas, cada una de las cuales es (generalmente) un pequeño cambio en el código fuente de un programa de computadora que preserva el comportamiento del software o al menos no modifica su conformidad con los requisitos funcionales.

Crea carpetas

1. Crea la carpeta “project”
2. En la carpeta “project” crea la carpeta “api”
 - a. En la carpeta “folder”
 - i. Crea el archivo `productController.js`
 - ii. Crea el archivo `productRoute.js`
 - iii. Crea el archivo `productModel.js`
 - b. Agrega `app.js` a project



productRouter.js

project/api/product/productRouter.js

```
const express = require('express')
const productRouter = express.Router()
const productController = require('../ProductController')

productRouter.get('/', productController.getProducts);
productRouter.post('/ ', productController.createProduct);
productRouter.get('/:id', productController.getProductById);
productRouter.patch('/:id', productController.updateProductById);
productRouter.delete('/:id', productController.deleteProductById);
productRouter.get('/get/statistic', productController.getStatistic);

module.exports = productRouter;
```

productController.js

project/api/product/productController.js

```
1 var CurrentProduct = require('./ProductModel')
2 > exports.createProduct = async function(req, res, next) { ...
17 }
18 //read by id
19 > exports.getProductById = function(req, res, next){ ...
32 }
33 //update
34 > exports.updateProductById= function(req, res, next){ ...
48 }
49 //delete
50 > exports.deleteProductById = function(req, res, next){ ...
64 }
65 //get
66 > exports.getProducts = async function(req, res, next){ ...
110 }
111 > exports.getStatistic = function(req, res, next){ ...
140 }
```

ProductModel.js

project/api/product/productModel.js

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;
const ProductSchema = new Schema({
  title: {
    type: String,
    required: [true, 'A product must have title'],
    unique: true,
    trim: true
  },
  description: {
    type: String,
    minlength: [5, 'Description is minimum 20 characters'],
    maxlength: [1000, 'Description is maximum 1000 characters']
  },
  price: {
    type: Number,
    required: [true, 'A product must have price'],
    min: [0, 'price must be above 0'],
    max: [10000, 'price must be below 10000']
  },
  created: Date
});

module.exports = mongoose.model('product', ProductSchema);
```

app.js

project/app.js

Use express.json for
body content

```
const express = require('express');
const app = express();
app.use(express.json());

const mongoose = require('mongoose');
const productRouter = require("./api/products/productRouter");

app.use('/api/v1/products', productRouter);

const strConnect = "mongodb+srv://asaf:asaf@cluster0.hgfhv.mongodb.net/myFirstDatabase?retryWrites=true&w=majority";
const OPT = {
    useNewUrlParser: true
};

mongoose.connect(strConnect, OPT);

const port = process.env.PORT || 3000;
app.listen(port, function() {
    console.log("Running on port " + port);
})
```

Ejercicio de refactorizacion 1-8 - para personRouter.js, personmodel y personController.js

Solicitud	Respuesta	Método	Comentario
/api/v1/players	Todos los "players"	Get	Obtener todos los "players" de players.json
/api/v1/players	El "player creado"	Post	Crea un nuevo Player
/api/v1/players/:id	"Player" por su id	Get	Obtener el "player" por su id
/api/v1/players/:id	Actualizar "player"	Patch	Actualizar "player" por su id
/api/v1/players/:id	Borrar "player"	Delete	Borrar "player" por su id

wawiwa

¿Preguntas?