

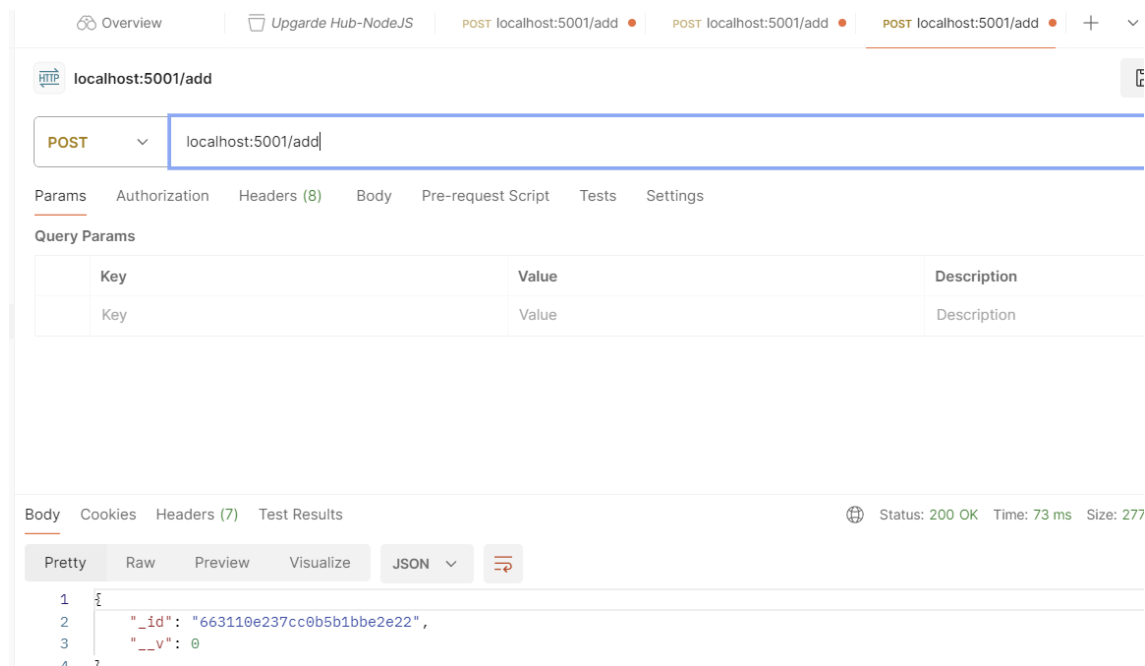
## Proyecto REST API con NodeJS-UpgradeHub Fullstack Developer Bootcamp

Author: Bilyana Ancheva

Simulaciones y pruebas de los endpoints CRUD a través de REST API en Postman y MongoDB

### Probar el postman para POST

localhost:5001/add



En Postman voy a Body-raw-json y creo el modelo de dato que tengo en models

```
name: { type: String, require: true },
color: { type: String },
model: { type: Number },
availability: { type: Boolean },
quantity: { type: Number }
```

El formato del JSON tiene que ser este:

```
{
  "name": "candle",
  "color": "red",
  "model": 467772,
  "availability": true,
  "quantity": 3,
  "_id": "663114969d51afb51404c7f8",
  "__v": 0
}
```

Esto ha creado UNA colección Product en la base UpgradeHub de MongoDB

The image shows two screenshots. The top screenshot is from the MongoDB Compass interface, displaying the 'product' collection in the 'upgradeHub' database. It shows two documents. The first document has an empty object body. The second document is a product entry:

```
{
  "_id": ObjectId('663114969d51afb51404c7f8'),
  "name": "candle",
  "color": "red",
  "model": 467772,
  "availability": true,
  "quantity": 3,
  "__v": 0
}
```

The bottom screenshot is from the Postman application, showing a POST request to 'localhost:5001/add'. The request body is a JSON object:

```
{
  "name": "candle",
  "color": "red",
  "model": 467772,
  "availability": true,
  "quantity": 3
}
```

The response status is 200 OK, and the response body is a JSON object with the same data as the request, plus an '\_id' and '\_\_v' field:

```
{
  "name": "candle",
  "color": "red",
  "model": 467772,
  "availability": true,
  "quantity": 3,
  "_id": "663114969d51afb51404c7f8",
  "__v": 0
}
```

CREATE endpoint SELECT de tipo GET

EN Postman es GET localhost:5001/select

OverviewUpgarde Hub-NodeJSPOST localhost:5001/addPOST localhost:5001/addGET localhost:5001/select+No envirom

localhost:5001/selectSave

GETlocalhost:5001/select

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

Query Params

Key	Value	Description	...
-----	-------	-------------	-----

BodyCookiesHeaders (7)Test ResultsStatus: 200 OKTime: 335 msSize: 662 BSave a

PrettyRawPreviewVisualizeJSON

```
3      "_id": "663119e237cc0b5d1dbe2e22",
4      "__v": 0
5    },
6    {
7      "_id": "663114969d51afb51404c7f8",
8      "name": "candle",
9      "color": "red",
10     "model": 467772,
11     "availability": true,
12     "quantity": 3,
13     "__v": 0
14   },
15   {
16     "_id": "663118c49d51afb51404c7fa",
17     "name": "golden rings",
18     "color": "white gold",
19     "model": 669543,
20     "availability": true,
21     "quantity": 10,
```

Crear otro enpoint selectProduct

Ruta para Postman GET localhost:5001/selectProduct/663114969d51afb51404c7f8

Este es el id de uno de los productos de la base de datos 663114969d51afb51404c7f8 que estoy seleccionando como producto con este endpoint

localhost:5001/selectproduct/663114969d51afb51404c7f8

GETlocalhost:5001/selectproduct/663114969d51afb51404c7f8

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

Query Params

Key	Value	Description
Key	Value	Description

BodyCookiesHeaders (7)Test ResultsStatus: 200 OKTime: 5

PrettyRawPreviewVisualizeJSON

```
1  [
2    {
3      "_id": "663114969d51afb51404c7f8",
4      "name": "candle",
5      "color": "red",
6      "model": 467772,
7      "availability": true,
8      "quantity": 3,
9      "__v": 0
10   }
```

Falta el similar el update

PUT localhost:5001/update/place the id you want to update

HTTP localhost:5001/update/663114969d51afb51404c7f8 Save

PUT localhost:5001/update/66311b7c73442578ddcad701

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "name": "towels",
3   "color": "brown",
4   "model": 699883,
5   "availability": false,
6   "quantity": 0
7 }
```

Body Cookies Headers (7) Test Results 🌐 Status: 200 OK Time: 51 ms Size: 359 B 📄

Pretty Raw Preview Visualize **JSON** ▾ 🔍

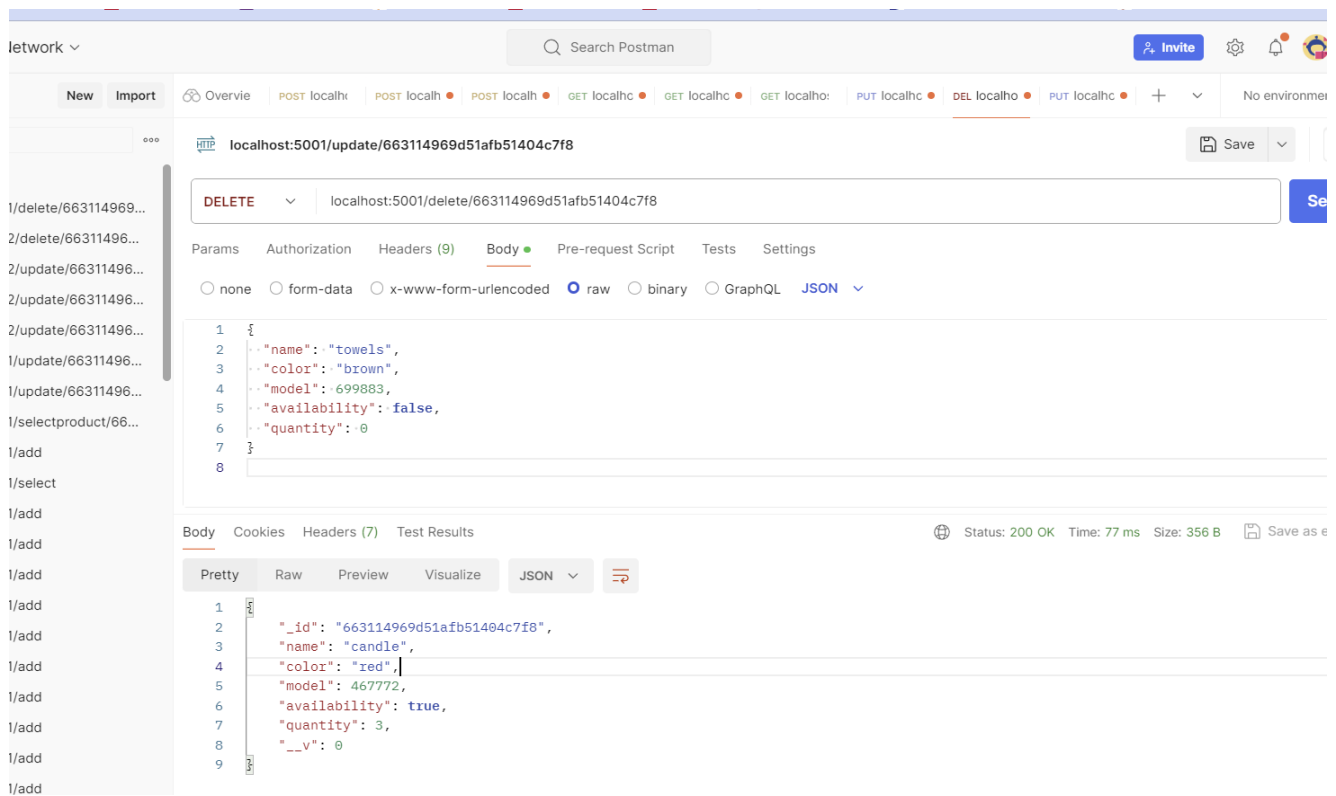
```
1 {
2   "_id": "66311b7c73442578ddcad701",
3   "name": "towels",
4   "color": "brown",
5   "model": 699883,
6   "availability": false,
7   "quantity": 0,
8   "__v": 0
9 }
```

```
{
  {
    name: 'mirror',
    color: 'white',
    model: 6558993,
    availability: true,
    quantity: 10
  }
  []
  {
    _id: new ObjectId('66311b7c73442578ddcad701'),
    name: 'towels',
    color: 'brown',
    model: 699883,
    availability: false,
    quantity: 0,
    __v: 0
  }
}
```

endpoint DELETE

Delete

localhost:5001/delete/663114969d51afb51404c7f8



## Render con la repo de GitHub

The screenshot shows the Render dashboard for a project named "api-nodejs-project-bilyana-1.onrender.com". The top navigation bar includes links for Dashboard, Blueprints, Env Groups, Docs, Community, and Help. A "New +" button and a user profile for "Bilyana Ancheva" are also visible. The main content area shows the project's status as "Live" and a message about the free instance spinning down with inactivity. Below this, a "Shell" tab is open, displaying a log of the deployment process. The logs show the build being successful, the deployment starting, and the service being live on port 5001. The logs also mention the Node version (20.12.2) and the Bun version (1.1.0).

## Prueba con la ruta de Render asociada al repo en GitHub

Usuario nuevo registrado con éxito poe el endpoint <https://api-nodejs-project-bilyana-1.onrender.com/user/register> con metodo POST

The screenshot shows a REST client interface with a POST request to the endpoint `https://api-nodejs-project-bilyana-1.onrender.com/user/register`. The request body is a JSON object with the following fields: `name` (John Doe), `email` (john@example.com), and `password` (Password123#). The response status is 200 OK, and the response body is a JSON object with the following fields: `success` (true), `data` (an object with `name`, `email`, `password`, `role`, `image`, `_id`, and `__v`), and `__v` (0).

## Otro user nuevo registrado

The screenshot shows a REST client interface with a POST request to `https://api-nodejs-project-bilyana-1.onrender.com/user/register`. The request body is a JSON object with the following fields:

```
1 {
2   "name": "Teodor Sanchez",
3   "email": "teo@example.com",
4   "password": "$2b$10$MD0kj/iz5swY9eDTd7VrIQ09UTxKRosFY7JIDycvp7.jT8zmVqM68y"
5 }
6
```

The response is a 200 OK status with a response time of 825 ms and a body size of 679 B. The response body is a JSON object with the following fields:

```
1 {
2   "success": true,
3   "data": {
4     "name": "Teodor Sanchez",
5     "email": "teo@example.com",
6     "password": "$2b$10$90Lt/1wXW4457feqlByuR.0MVQ1qbw5H09mqglAjQCRQ6sL2eP.10",
7     "role": "user",
8     "image": "",
9     "_id": "663371f93473a77ff27a6d1c",
10    "__v": 0
11  }
12 }
```

## Si la contraseña no cumple con el patron, lo detecta y devuelve Usuario no registrado



The screenshot shows a REST client interface with a POST request to `https://api-nodejs-project-bilyana-1.onrender.com/user/register`. The request body is a JSON object with the following fields:


```
1 {
2   "name": "John Doe",
3   "email": "jon@example.com",
4   "password": "password123"
5 }
6
```

The response is a 200 OK status with a response time of 155 ms and a body size of 541 B. The response body is a JSON object with the following fields:


```
1 {
2   "success": false,
3   "message": "La contraseña no cumple con el patron indicado"
4 }
```


Para el mismo usuario Jon si se pone una contraseña mas fuerte, ya si que se consigue registro nuevo


[HTTP](#) Upgarde Hub-NodeJS / <https://api-nodejs-project-bilyana-1.onrender.com/user/register>  Save  Share

POST 




<https://api-nodejs-project-bilyana-1.onrender.com/user/register>



Send 

Params Authorization Headers (9) **Body**  Scripts Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON  Beautify

```
1 {
2   "name": "John Doe",
3   "email": "jon@example.com",
4   "password": "$2b$10$MD0kj/iz5swY9eDTd7VrQ09UTxKRosFY7JIDycvp7.jT8zmVqM68y"
5 }
6
```

Body Cookies Headers (15) Test Results  200 OK 613 ms 673 B  Save as example 

Pretty Raw Preview Visualize JSON  

```
1 {
2   "success": true,
3   "data": {
4     "name": "John Doe",
5     "email": "jon@example.com",
6     "password": "$2b$10$qiqzfKf4NZvcigi0nCAGt0eCJj67v0z7YBea7NKZT7wsGyyKV4UzS",
7     "role": "user",
8     "image": "",
9     "_id": "663373c33473a77ff27a6d21",
10    "__v": 0
11  }
12 }
```



Login de jon con email y password inválidos, no se genera token

The screenshot shows a REST client interface with the following details:

- URL:** `https://api-nodejs-project-bilyana-1.onrender.com/user/login`
- Method:** `POST`
- Body (JSON):**

```
1 {
2   "email": "jon@example.com",
3   "password": "$2b$10$qiqzfkf4NZvcigi0nCAGt0eCJj67v0z7YBea7NKZT7wsGyyKV4UzS"
4 }
```
- Response:**
  - Status:** `200 OK`
  - Time:** `429 ms`
  - Size:** `512 B`
  - Body (JSON):**

```
1 {
2   "succes": true,
3   "message": "contraseña invalida"
4 }
```

Login de John con email y password valido, se genera token

[illegible]

HTTP

Upgrade Hub-NodeJS / https://api-nodejs-project-bilyana-1.onrender.com/user/login

Save

Share

POST

https://api-nodejs-project-bilyana-1.onrender.com/user/login

Send

Params

Authorization

Headers (9)

Body ●

Scripts

Tests

Settings

Cookies

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON ▾

Beautify

```
1 {  
2  
3   "email": "dor@example.com",  
4   "password": "$2b$23$LyoPp/Iz5swY9eDTd7VrQ09UTxKRosFY7JIDycvp7.jT8zmVqM68y"  
5 }
```

Body

Cookies

Headers (15)

Test Results

200 OK 463 ms 727 B

Save as example

Pretty

Raw

Preview

Visualize

JSON ▾

```
1 {  
2   "success": true,  
3   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
           eyJuYWllIjoiRG9yZWVuIEdyYXkiLCJlbWFnbiCI6ImRvckBleGFtcGxlLnNvbSI6IjlpZCJlcyZmZiIwOWVhOWE2ZmJh  
           NTEyNzkzMWIZYSImIhdCI6MTcxNDY0ODYzOSwiZXhwIjozNzE0NjUyMjMfQ.  
           f-sJi81F0e7VRHHrIzQ9IMbivIM5h-0Ug4LyBx12X6c"  
4 }
```

En Mongo DB

Se han registrado dinámicamente a través del endpoint probado en Postman, 3 usuarios

BILYANA'S ORG - 2024-04-01 > PROJECT 0 > DATABASES

ClusterO

Overview

Real Time

Metrics

Collections

Atlas Search

Performance Advisor

Online Archive

Cmd Line Tools

DATABASES: 4 COLLECTIONS: 14

+ Create Database

Search Namespaces

Nueva\_ejercicio0

auth

sample\_mflix

upgradeHub

courses

orders

product

students

teachers

user

upgradeHub.user

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 546B TOTAL DOCUMENTS: 3 INDEXES TOTAL SIZE: 36KB

Find

Indexes

Schema Anti-Patterns

Aggregation

Search Indexes

Generate queries from natural language in Compass

Filter

Type a query: { field: 'value' }

QUERY RESULTS: 1-3 OF 3

\_id: ObjectId('663209ea9a6fba5127921b3a')

name: "Doreen Gray"

email: "dor@example.com"

password: "\$2b\$10\$eH7x.CbRxyNET/tBAC0ih0/FBM/7KlHhZjGsGM7mISjCWLdIPoTZG"

role: "user"

image: ""

\_\_v: 0

\_id: ObjectId('663371f93473a77ff27a6d1c')

name: "Teodor Sanchez"

email: "teo@example.com"

password: "\$2b\$10\$90Lt/1wXW4457feq1ByuR.0MVQ1qbw5H09mqg1AjQCRQ6sL2eP.L0"

role: "user"

image: ""

\_\_v: 0

\_id: ObjectId('663373c33473a77ff27a6d21')

name: "John Doe"

email: "jon@example.com"

password: "\$2b\$10\$qiqzFKf4NZvcig18nCAGt0eCJj67v0z7YBea7NKZT7wsGyyKV4Uz5"

role: "user"

image: ""

\_\_v: 0