



Curso complementario Desarrollo de Back-end con Node.js - MongoDB

ING - DIEGO CASALLAS

22810019-1



www.sena.edu.co

Node.js - Mongo.db



CRUD - Task

- **Controlador de tareas**
 - addTask
 - getTaskId
 - getTasks
 - updateTask
 - addComment
 - addFile

```
Task_management > controllers > JS task.controller.js > [e] default

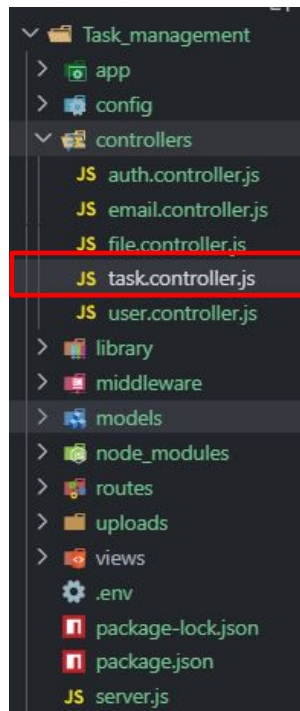
1  import TaskModel from '../models/task.model.js'; // Import the user model
2  import { mongoose } from '../config/db/connection.js'; // Import mongoose from the connection
3
4  class TaskController {
5      // Create a new task
6  >   async addTask(req, res) { ...
79   };
80
81   //show all tasks
82 >   async getTaskId(req, res) { ...
93   };
94   //show all tasks
95 >   async getTasks(req, res) { ...
105  };
106
107   // Update a task add array of managers
108 >   async updateTask(req, res) { ...
137  };
138
139   //Add comment to a task
140 >   async addComment(req, res) { ...
162  };
163
164   // Add file to a task
165 >   async addFile(req, res) { ...
192  };
193  }
194
195  export default new TaskController();
```

Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - addTask



```
Task_management > controllers > JS task.controller.js > TaskController

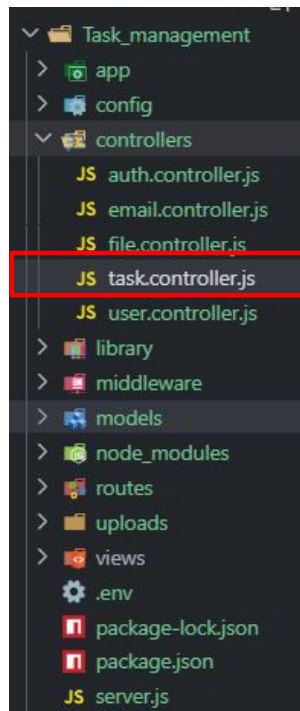
1  import TaskModel from '../models/task.model.js'; // Import the user model
2  import { mongoose } from '../config/db/connection.js'; // Import mongoose from the connection
3
4  class TaskController {
5      // Create a new task
6      async addTask(req, res) {
7          try {
8              const {
9                  userId,
10                 title,
11                 description,
12                 type,
13                 priority,
14                 color,
15                 dateStart,
16                 dateEnd,
17                 notificationEmail,
18                 assignees,
19                 createdBy,
20                 tags
21             } = req.body;
22
23             // Validate required fields
24             if (!userId) {
25                 return res.status(400).json({ error: 'User ID is required' });
26             }
27         }
28     }
29 }
```

Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - addTask



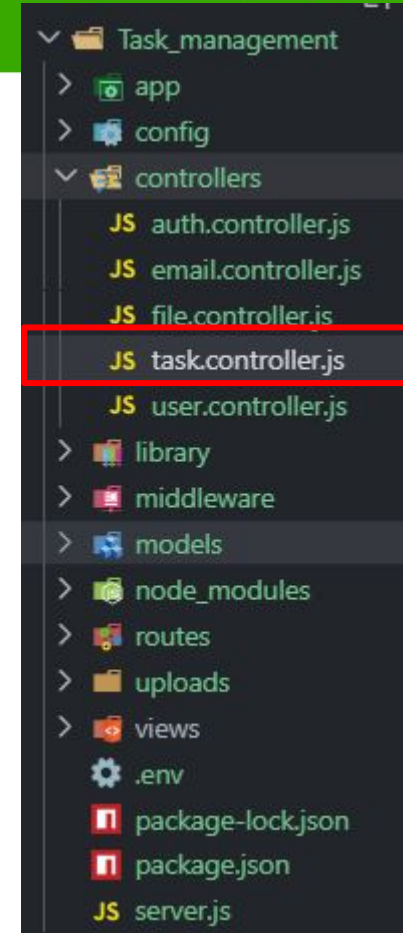
```
27
28 //Vaildate required fields
29 if (!title || !type || !dateStart || !dateEnd) {
30   return res.status(400).json({ error: 'Title, type, start date, and end date are required' });
31 }
32
33 // Convert date strings to Date objects
34 const startDate = new Date(dateStart);
35 const endDate = new Date(dateEnd);
36 if (isNaN(startDate) || isNaN(endDate)) {
37   return res.status(400).json({ error: 'Invalid date format' });
38 }
39 if (endDate < startDate) {
40   return res.status(400).json({ error: 'End date must be after start date' });
41 }
42 // Create a new task
43 const newTasks = new TaskModel({
44   title,
45   description,
46   type,
47   priority,
48   color,
49   startDate,
50   endDate,
51   notificationEmail,
52   assignees: Array.isArray(assignees)
53     ? assignees.map(assignee => ({
54       name: assignee.name,
55       email: assignee.email,
56       role: assignee.role,
57       userId: assignee.userId // Do not convert to ObjectId here
58     }))
59     : [],
60   projectId: req.body.projectId ? new mongoose.Types.ObjectId(req.body.projectId) : null, // Convert to ObjectId if provided
61   createdBy: createdBy || userId, // Use userId if createdBy is not provided
62   tags: Array.isArray(tags) ? tags.map(tag => tag.name) : []
63 });
```

Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - addTask



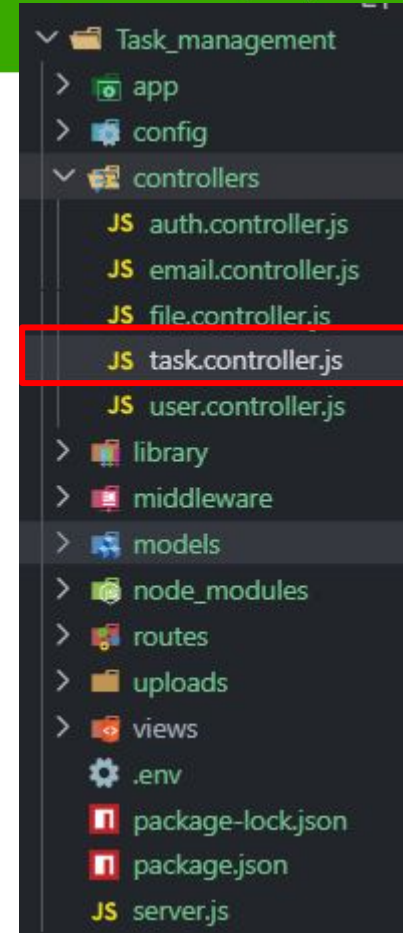
```
64
65     const savedTasks = await newTasks.save();
66     // Optional: Send notification by email
67     if (notificationEmail) {
68         // Here you would integrate the emailController
69         // emailController.sendNotificationEmail(notificationEmail, savedTasks);
70     }
71     // Return the saved task as a response
72     res.status(201).json({ "data": savedTasks });
73 } catch (error) {
74     res.status(400).json({
75         error: error.message,
76         details: error.errors
77     });
78 }
79 };
```


Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - getTaskId



```
81 //show all tasks
82 async getTaskId(req, res) {
83   try {
84     const task = await TaskModel.findById(req.params.id);
85     if (!task) {
86       return res.status(404).json({ error: 'Task not found' });
87     }
88
89     res.status(200).json(task);
90   } catch (error) {
91     res.status(500).json({ error: 'Error in get task details' });
92   }
93 };
94
```

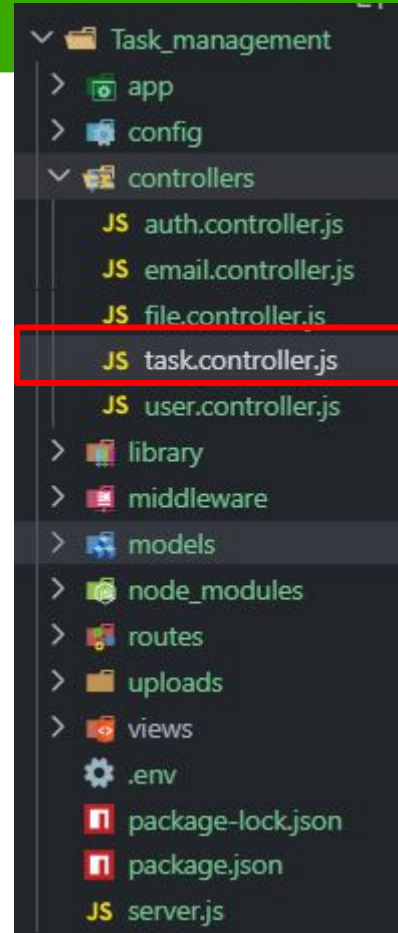
Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - getTasks

```
95 //show all tasks
96 async getTasks(req, res) {
97   try {
98     const tasks = await TaskModel.find();
99     if (!tasks || tasks.length === 0) {
100       return res.status(404).json({ error: 'No tasks found' });
101     }
102     res.status(200).json(tasks);
103   } catch (error) {
104     res.status(500).json({ error: 'Error in get task details' });
105   }
106 };
```



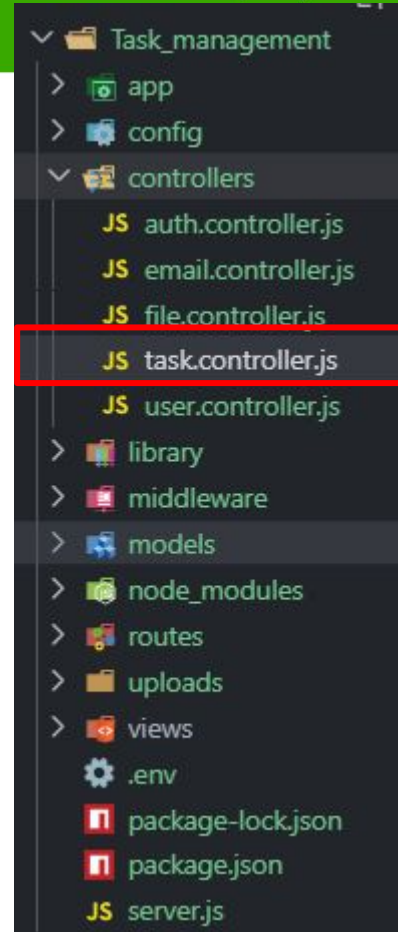
Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - updateTask

```
108 // Update a task add array of managers
109 async updateTask(req, res) {
110   try {
111     const updates = req.body;
112
113     // Convert date strings to Date objects
114     if (updates.dateStart) updates.dateStart = new Date(updates.dateStart);
115     if (updates.dateEnd) updates.dateEnd = new Date(updates.dateEnd);
116
117     // Convert ids in managers to ObjectId
118     if (updates.managers) {
119       updates.managers = updates.managers.map(inCharge => ({
120         ...inCharge,
121         userId: new mongoose.Types.ObjectId(inCharge.userId)
122       }));
123     }
124
125     const updateTask = await TaskModel.findByIdAndUpdate(
126       req.params.id,
127       { $set: updates },
128       { new: true, runValidators: true }
129     );
130
131     res.status(200).json(updateTask);
132   } catch (error) {
133     res.status(400).json({
134       error: error.message,
135       details: error.errors
136     });
137   }
138 };
139
```



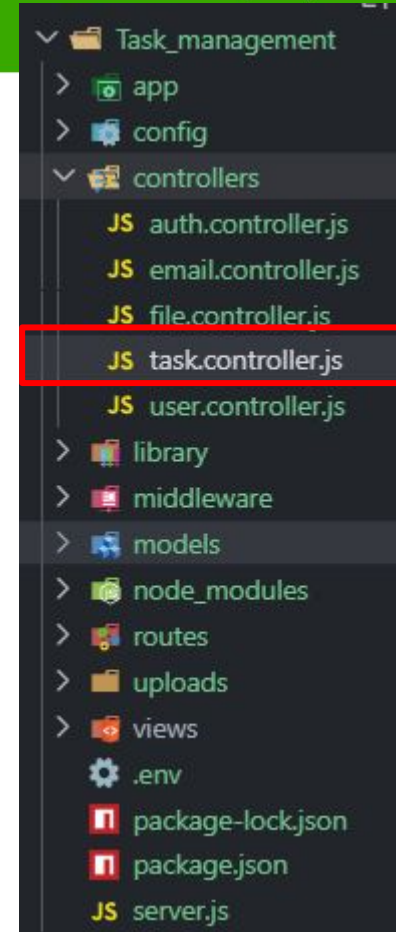
Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - addComment

```
140 //Add comment to a task
141 async addComment(req, res) {
142   try {
143     const { userId, name, text } = req.body;
144     // Validate required fields
145     if (!userId || !name || !text) {
146       return res.status(400).json({ error: 'User ID, name, and text are required' });
147     }
148     // Find the task and add the comment
149     const task = await TaskModel.findById(req.params.id);
150
151     if (!task) {
152       return res.status(404).json({ error: 'Task not found' });
153     }
154     // Check if the comment already exists
155     const existingComment = task.comments.find(comment => comment.userId.toString() === userId && comment.text === text);
156     if (existingComment) {
157       return res.status(400).json({ error: 'Comment already exists' });
158     }
159     // Add the comment
160     task.comments.push({
161       userId: userId,
162       name: name,
163       text: text,
164       date: new Date()
165     });
166     // Save the updated task
167     await task.save();
168
169     // Return the updated task
170     res.status(200).json(task);
171   } catch (error) {
172     res.status(400).json({ error: error.message });
173   }
174 }
```



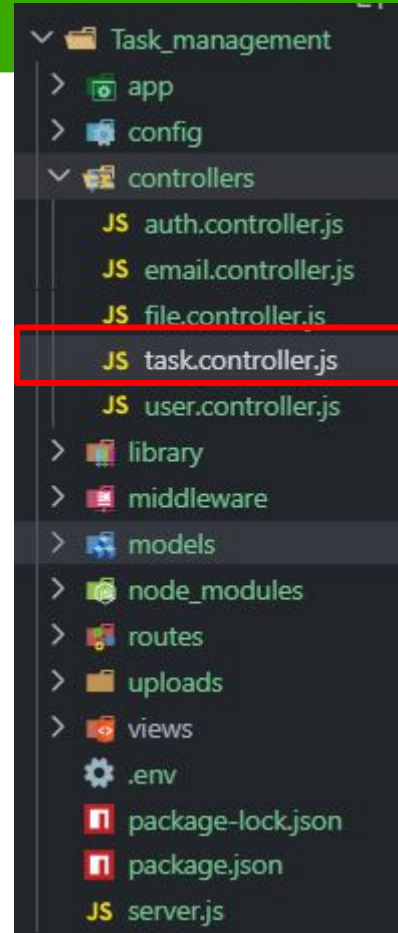
Node.js - Mongo.db



CRUD - Task

- Controlador de tareas
 - addFile

```
177 > async addFile(req, res) {
178 >   try {
179 >     // Assuming multer uploaded the file and we have the info in req.file
180 >     const { name, extension, type, size, url } = req.file;
181 >     if (!name || !extension || !type || !size || !url) {
182 >       return res.status(400).json({ error: 'File information is incomplete' });
183 >     }
184 >     const task = await TaskModel.findByIdAndUpdate(
185 >       req.params.id,
186 >       {
187 >         $push: {
188 >           files: {
189 >             name: name,
190 >             url: url,
191 >             type: type,
192 >             extension: extension,
193 >             size: size
194 >           }
195 >         },
196 >         { new: true }
197 >       );
198 >
199 >     res.status(200).json(task);
200 >   } catch (error) {
201 >     res.status(400).json({ error: error.message });
202 >   }
203 > };
204 >
205 > }
206 >
207 > export default new TaskController();
```



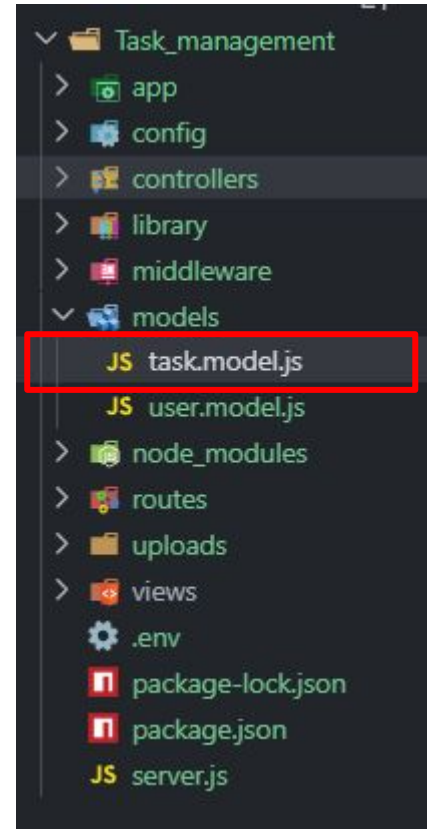
Node.js - Mongo.db



CRUD - Task

- Actualizar modelo
 - models -> task.model.js

```
12 // Subdocument for files
13 v const fileSchema = new Schema({
14     name: { type: String, required: true },
15     url: { type: String, required: true },
16     type: { type: String, enum: ['document', 'image', 'video', 'other'] },
17     extension: { type: String, required: true },
18     size: { type: Number }, // in KB
19     uploadDate: { type: Date, default: Date.now }
20 });
```



Node.js - Mongo.db



CRUD - Task

- Actualizar rutas
 - task.routes.js

Task_management > routes > JS task.routes.js > default

```
1 import { Router } from "express";
2 import TaskController from '../controllers/task.controller.js';
3 import { verifyToken } from '../middleware/authMiddleware.js';
4
5 const router = Router();
6 const name = "/task";
7 const nameComment = "/task/comment";
8 const nameFiles = "/task/files";
9 // Verify token middleware
10 router.use(verifyToken);
11 // Route for user registration and list
12 router.route(name)
13   .post(TaskController.addTask)
14   .get(TaskController.getTasks);
15
16 //Route for user by ID
17 router.route(`${name}/:id`)
18   .get(TaskController.getTaskId)
19   .put(TaskController.updateTask);
20
21 router.route(`${nameComment}/:id`).post(TaskController.addComment);
22
23 router.route(`${nameFiles}/:id`)
24   .post(TaskController.addFile);
25
26 export default router;
```

Task_management

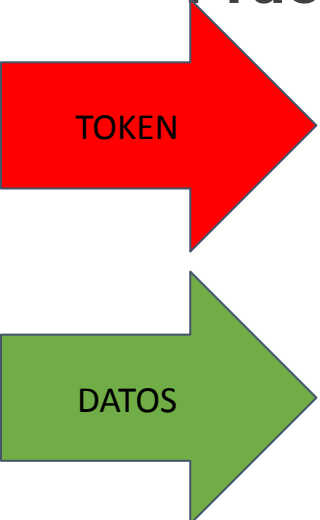
- > app
- > config
- > controllers
- > library
- > middleware
- > models
- > node_modules
- ▼ routes
 - JS auth.routes.js
 - JS email.routes.js
 - JS file.routes.js
 - JS task.routes.js**
 - JS user.routes.js
- > uploads
- > views
- gear .env
- package-lock.json
- package.json
- JS server.js

Node.js - Mongo.db



CRUD - Task

- Pruebas



POST

http://localhost:3000/api/task

Send

Params

Authorization

Headers (11)

Body

Scripts

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

{

2

"userId": "68759c73b17af6b273cd44",

3

"title": "Task Management",

4

"description": "Manage tasks effectively with features like notifications and tagging.",

5

"type": "Development",

6

"priority": "High",

7

"color": "#FF5733",

8

"dateStart": "2023-10-01T00:00:00Z",

9

"dateEnd": "2023-10-31T23:59:59Z",

10

"projectId": 0,

}

Body

Cookies (1)

Headers (7)

Test Results

201 Created

102 ms

955 B

Save Response

{}

JSON

Preview

Visualize

1

{

2

"data": {

3

"title": "Task Management",

TASK-MONGO-NODE

>

auth

>

user

>

task

GET

show

GET

showId

POST

add

POST

add Comments

POST

add Files

PUT

update

DEL

delete

>

email

>

files

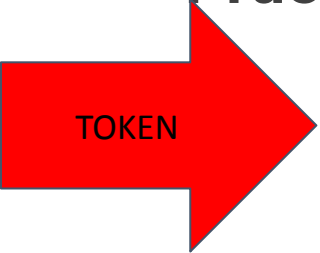
MÉTODO	RUTA	CONTROLADOR	MODELO
POST	http://localhost:3000/api/task	Task	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



GET {{baseUrl}} /api/task

Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

Auth Type Bearer Token

Token

200 OK • 77 ms • 6 KB • Save Response

JSON Preview Visualize

```
1 [
2   {
3     "_id": "687a7d6d0c058a7370b519d7",
4     "title": "Task Management",
5     "description": "Manage tasks effectively with features like notifications and tagging.",
6     "type": "Development",
7     "priority": "High",
8     "color": "#FF5733",
9     "startDate": "2023-10-01T00:00:00.000Z",
10    "endDate": "2023-10-31T23:59:59.000Z",
11    "status": "In Progress",
12    "notificationEmail": "diehercas@gmail.com",
13    "assignees": [
14      {
```

TASK-MONGO-NODE

- > auth
- > user
- > task
 - GET show
 - GET showId
 - POST add
 - POST add Comments
 - POST add Files
 - PUT update
 - DEL delete
- > email
- > files

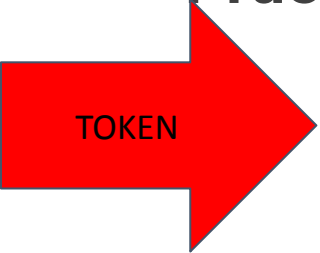
MÉTODO	RUTA	CONTROLADOR	MODELO
GET	http://localhost:3000/api/task	Task	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



GET Send

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

Auth Type: Bearer Token

Token:

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.

Body Cookies (1) Headers (7) Test Results 200 OK • 20 ms • 1.25 KB Save Response

```
{
  "_id": "687a7d6d0c058a7370b519d7",
  "title": "Task Management",
  "description": "Manage tasks effectively with features like notifications and tagging.",
  "type": "Development",
  "priority": "High",
  "color": "#FF5733",
  "startDate": "2023-10-01T00:00:00.000Z",
  "endDate": "2023-10-31T23:59:59.000Z",
  "status": "In Progress",
  "notificationEmail": "diehercas@gmail.com",
  "assignees": []
}
```

TASK-MONGO-NODE

- > auth
- > user
- task
 - GET show
 - GET showId**
 - POST add
 - POST add Comments
 - POST add Files
 - PUT update
 - DEL delete
- > email
- > files

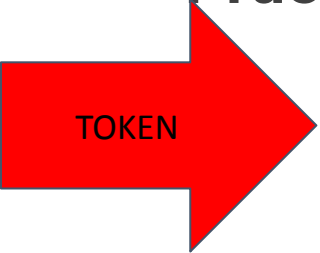
MÉTODO	RUTA	CONTROLADOR	MODELO
GET	http://localhost:3000/api/task:id	Task	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



PUT

{{baseURL}} /api/task/687a7d6d0c058a7370b519d7

Send

Params

Authorization

Headers (11)

Body

Scripts

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

{

2

"title": "Task Management",

3

"description": "Manage tasks effectively with features like notifications and tagging.",

4

"type": "Development",

5

"priority": "High",

6

"color": "#FF5733",

7

"createdBy": "6875cfc7fb0f3af1da9f08dd",

8

"notificationEmail": "diehercas@gmail.com",

9

"status": "In Progress",

10

"assignees": [

11

]

}

Body

Cookies (1)

Headers (7)

Test Results

200 OK

17 ms

931 B

Save Response

JSON

Preview

Visualize

16

"email": "diehercas@gmail.com",

17

"role": "developer",

18

"_id": "687a88e08082fd0fb1563cf3"

19

}

20

}

TASK-MONGO-NODE

>

auth

>

user

>

task

GET

show

GET

showId

POST

add

POST

add Comments

POST

add Files

PUT

update

DEL

delete

>

email

>

files

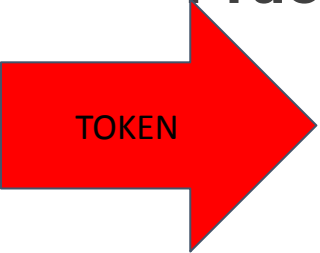
MÉTODO	RUTA	CONTROLADOR	MODELO
PUT	http://localhost:3000/api/task:id	Tesk	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



DELETE Send

Params **Authorization** Headers (8) Body Scripts Tests Settings

Query Params

	Key	Value	Description	
	Key	Value	Description	

TASK-MONGO-NODE

- > auth
- > user
- ▼ task
 - GET show
 - GET showId
 - POST add
 - POST add Comments
 - POST add Files
 - PUT update
 - DEL delete**
- > email
- > files

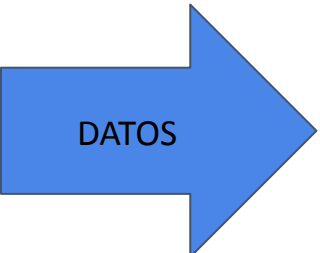
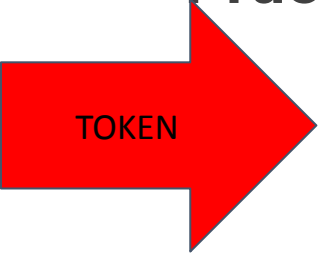
MÉTODO	RUTA	CONTROLADOR	MODELO
DELETE	http://localhost:3000/api/task:id	Tesk	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



POST

http://localhost:3000/api/task/comment/687a7eb40c058a7370b519da

Send

Params

Authorization

Headers (11)

Body

Scripts

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

```
1 {
2   "userId": "6875cfc7fb0f3af1da9f08dd",
3   "name": "Diego Casallas Vanegas",
4   "text": "comment 2"
5 }
6
```

Body

Cookies (1)

Headers (7)

Test Results

200 OK

19 ms

1.21 KB

Save Response

JSON

Preview

Visualize

```
18     "_id": "687a7eb40c058a7370b519db"
19   },
20 ],
21   "tags": [
22     "urgent",
23     "development"
24   ],
25   "projectId": null,
26   "createdBy": "6875cfc7fb0f3af1da9f08dd",
27   "creationDate": "2025-07-18T17:04:52.638Z",
28   "files": [],
```

TASK-MONGO-NODE

>

auth

>

user

>

task

GET

show

GET

showId

POST

add

POST

add Comments

POST

add Files

PUT

update

DEL

delete

>

email

>

files

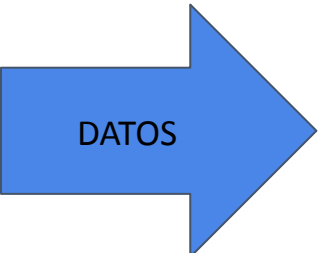
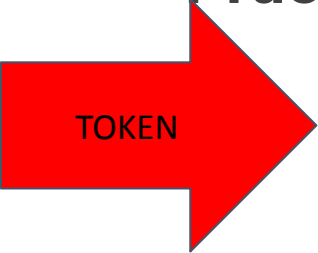
MÉTODO	RUTA	CONTROLADOR	MODELO
POST	http://localhost:3000/api/task/comment/:id	Task	Task

Node.js - Mongo.db



CRUD - Task

- Pruebas



POST Send

Params Authorization Headers (11) Body Scripts Tests Settings Cookies Beautify

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

```
1 {
2   "message": "File uploaded successfully",
3   "name": "1752862277583-413348011-LogoAprobadoaCina22023-01blanco-01.png",
4   "originalName": "Logo Aprobado aCina2 2023-01 blanco-01.png",
5   "size": 68435,
6   "type": "image",
7   "url": "C:\\xampp\\htdocs\\SENA\\SENA_ADS0\\BACK-END\\Node.
      js-Mongo\\Task_management\\uploads\\1752862277583-413348011-LogoAprobadoaCina22023-01blanco-01.png",
8   "extension": ".png",
9   "uploadedAt": "2025-07-18T18:11:17.592Z"
}
```

Body Cookies (1) Headers (7) Test Results 200 OK 84 ms 1.25 KB Save Response

{ } JSON Preview Visualize

```
31 "type": "image",
32 "extension": ".png",
33 "size": 68435,
```

TASK-MONGO-NODE

- > auth
- > user
- task
 - GET show
 - GET showId
 - POST add
 - POST add Comments
 - POST add Files
 - PUT update
 - DEL delete
- > email
- > files

MÉTODO	RUTA	CONTROLADOR	MODELO
POST	http://localhost:3000/api/task/files/:id	Task	Task



G R A C I A S

Línea de atención al ciudadano: 01 8000 910270
Línea de atención al empresario: 01 8000 910682



www.sena.edu.co