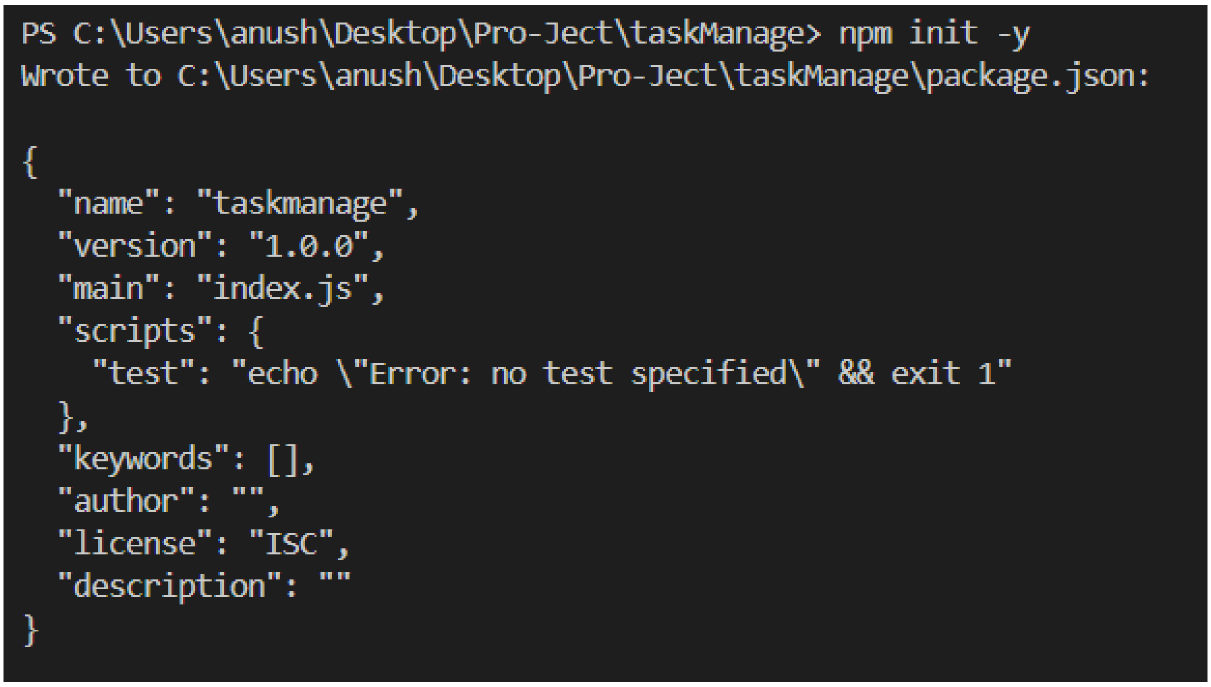
# Task Management System

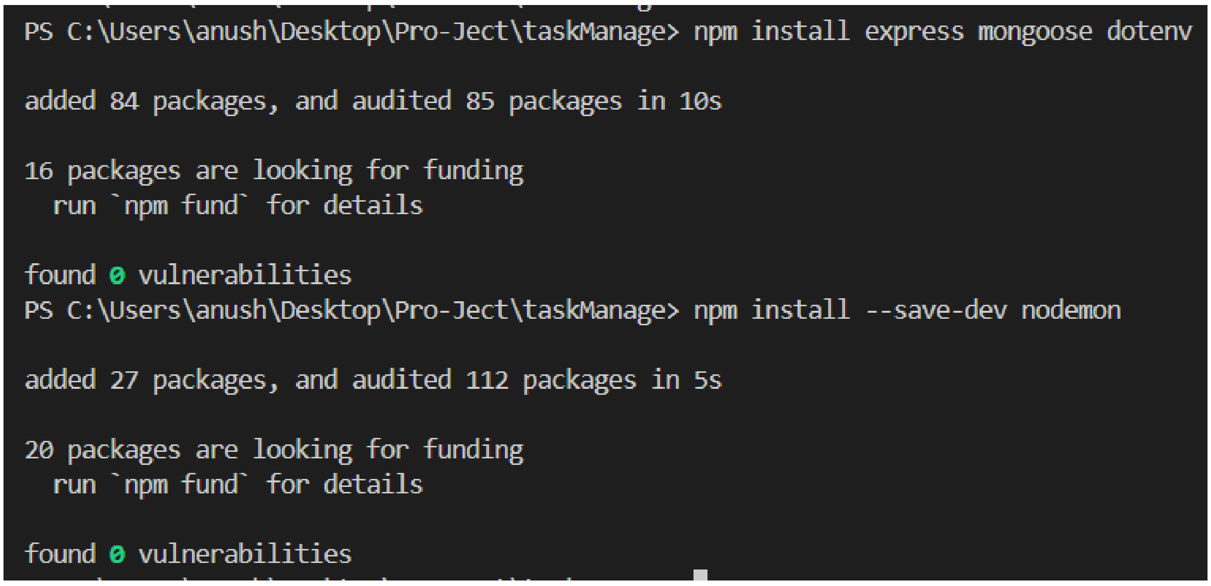
## In Project folder terminal

### Initialize the Node.js



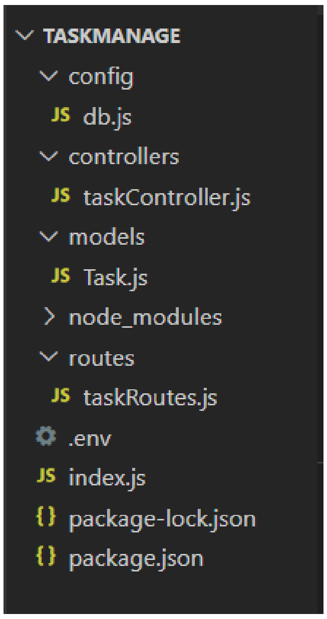
```
PS C:\Users\anush\Desktop\Pro-Ject\taskManage> npm init -y
Wrote to C:\Users\anush\Desktop\Pro-Ject\taskManage\package.json:

{
  "name": "taskmanage",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

### Install Dependencies



```
PS C:\Users\anush\Desktop\Pro-Ject\taskManage> npm install express mongoose dotenv

added 84 packages, and audited 85 packages in 10s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\anush\Desktop\Pro-Ject\taskManage> npm install --save-dev nodemon

added 27 packages, and audited 112 packages in 5s

20 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

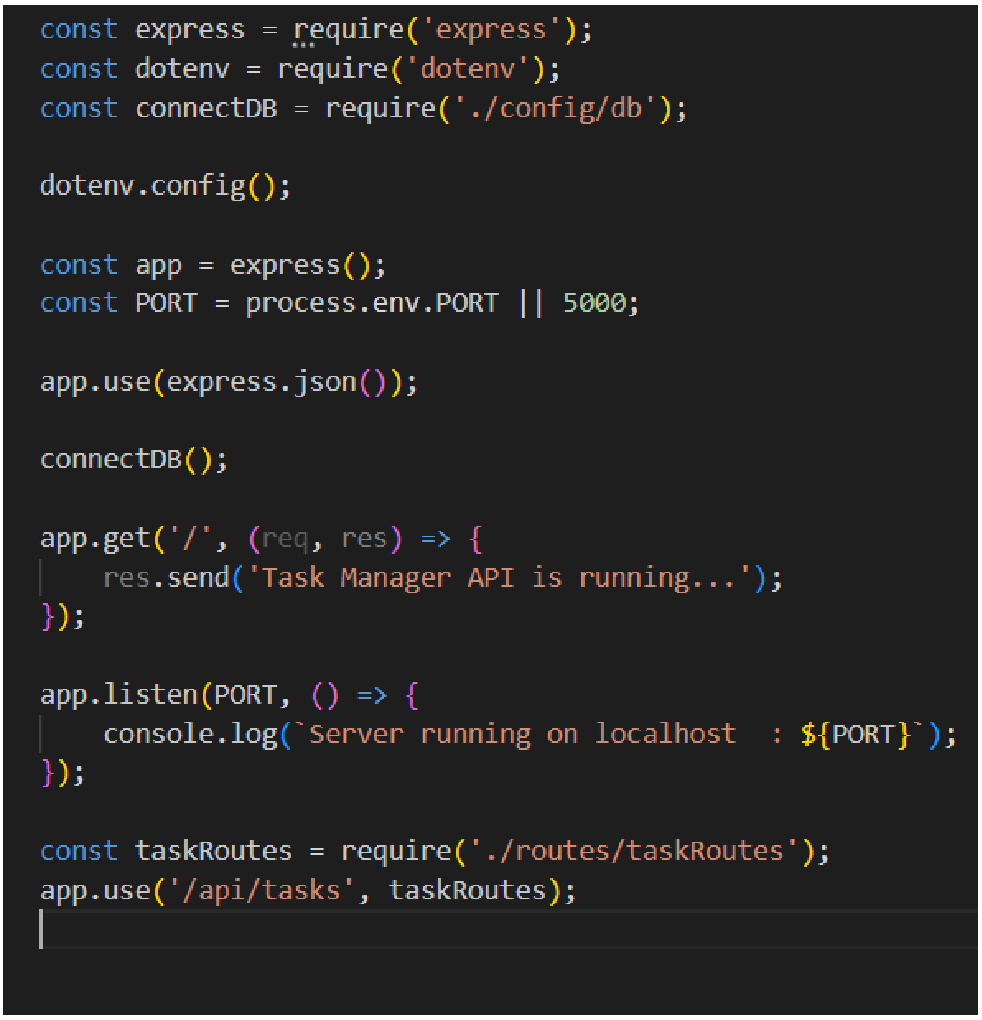
## Folder Structure



```
TASKMANAGE
  config
    db.js
  controllers
    taskController.js
  models
    Task.js
  node_modules
  routes
    taskRoutes.js
  .env
  index.js
  package-lock.json
  package.json
```

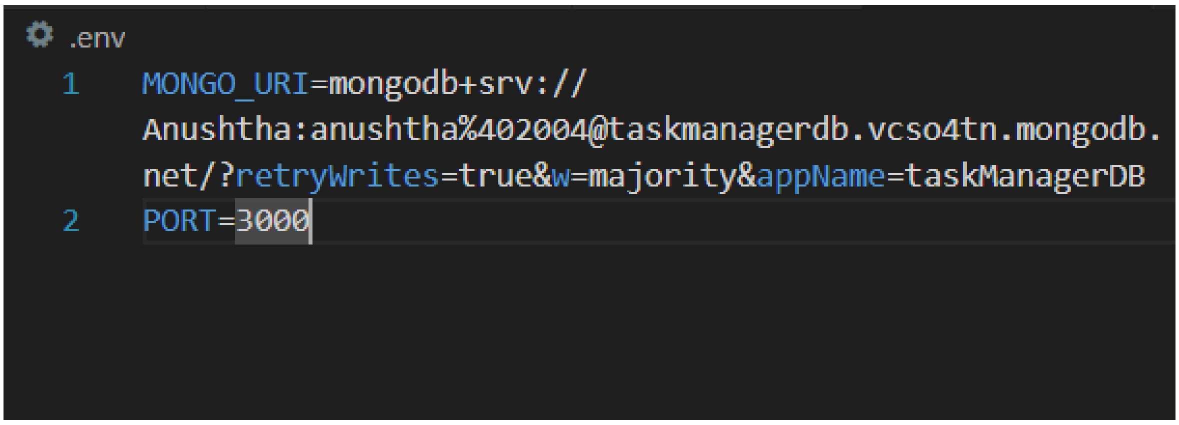
## Code

### Index.js



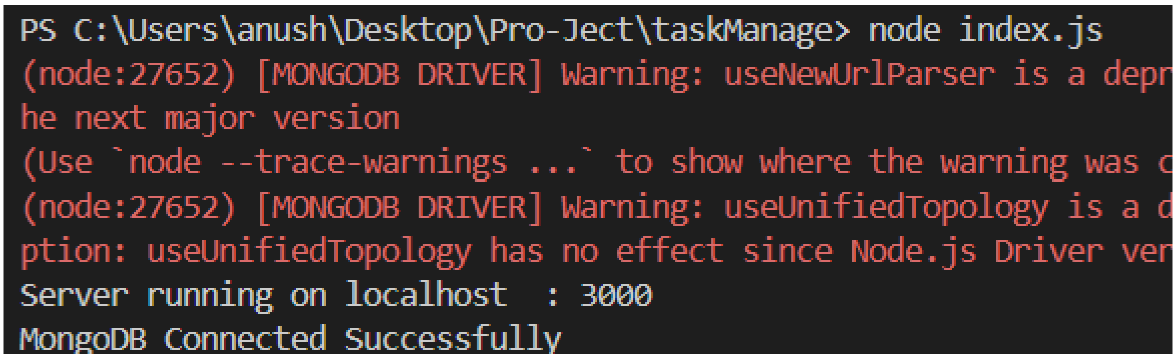
```
const express = require('express');
const dotenv = require('dotenv');
const connectDB = require('./config/db');

dotenv.config();

const app = express();
const PORT = process.env.PORT || 5000;

app.use(express.json());

connectDB();

app.get('/', (req, res) => {
    res.send('Task Manager API is running...');
});

app.listen(PORT, () => {
    console.log(`Server running on localhost  : ${PORT}`);
});

const taskRoutes = require('./routes/taskRoutes');
app.use('/api/tasks', taskRoutes);
```

**Connection to MongoDB**



```
.env
MONGO_URI=mongodb+srv://
Anushtha:anushtha%402004@taskmanagerdb.vcso4tn.mongodb.
net/?retryWrites=true&w=majority&appName=taskManagerDB
PORT=3000
```

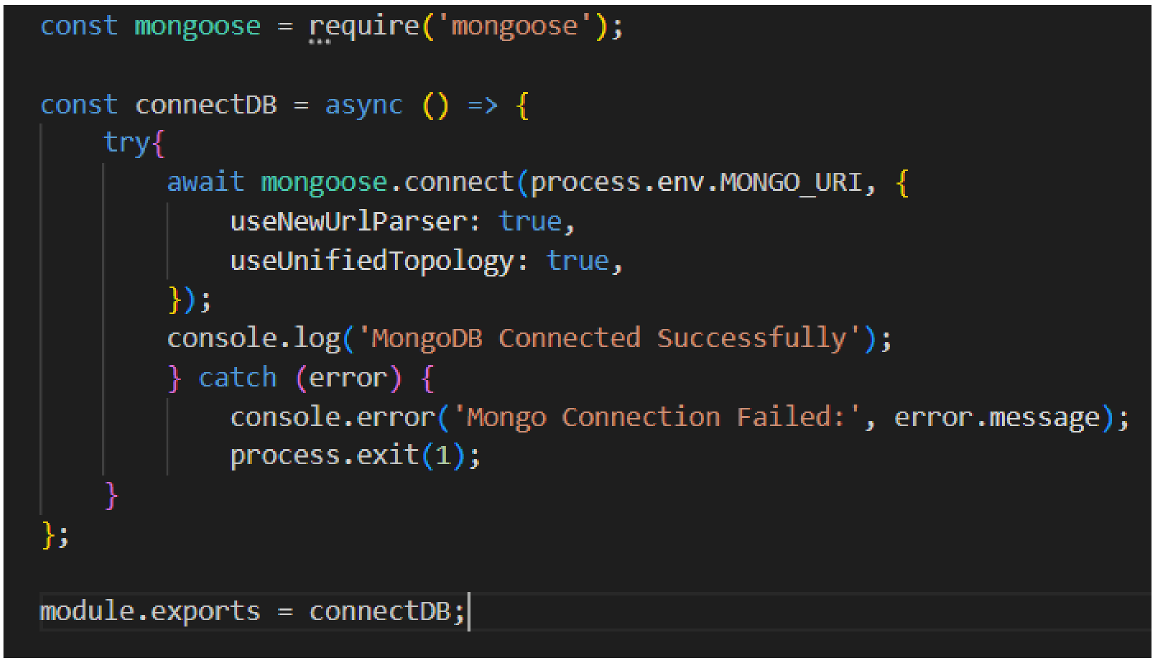
## Check The Connection



```
PS C:\Users\anush\Desktop\Pro-Ject\taskManage> node index.js
(node:27652) [MONGODB DRIVER] Warning: useNewUrlParser is a depr
he next major version
(Use `node --trace-warnings ...` to show where the warning was c
(node:27652) [MONGODB DRIVER] Warning: useUnifiedTopology is a d
ption: useUnifiedTopology has no effect since Node.js Driver ver
Server running on localhost  : 3000
MongoDB Connected Successfully
```

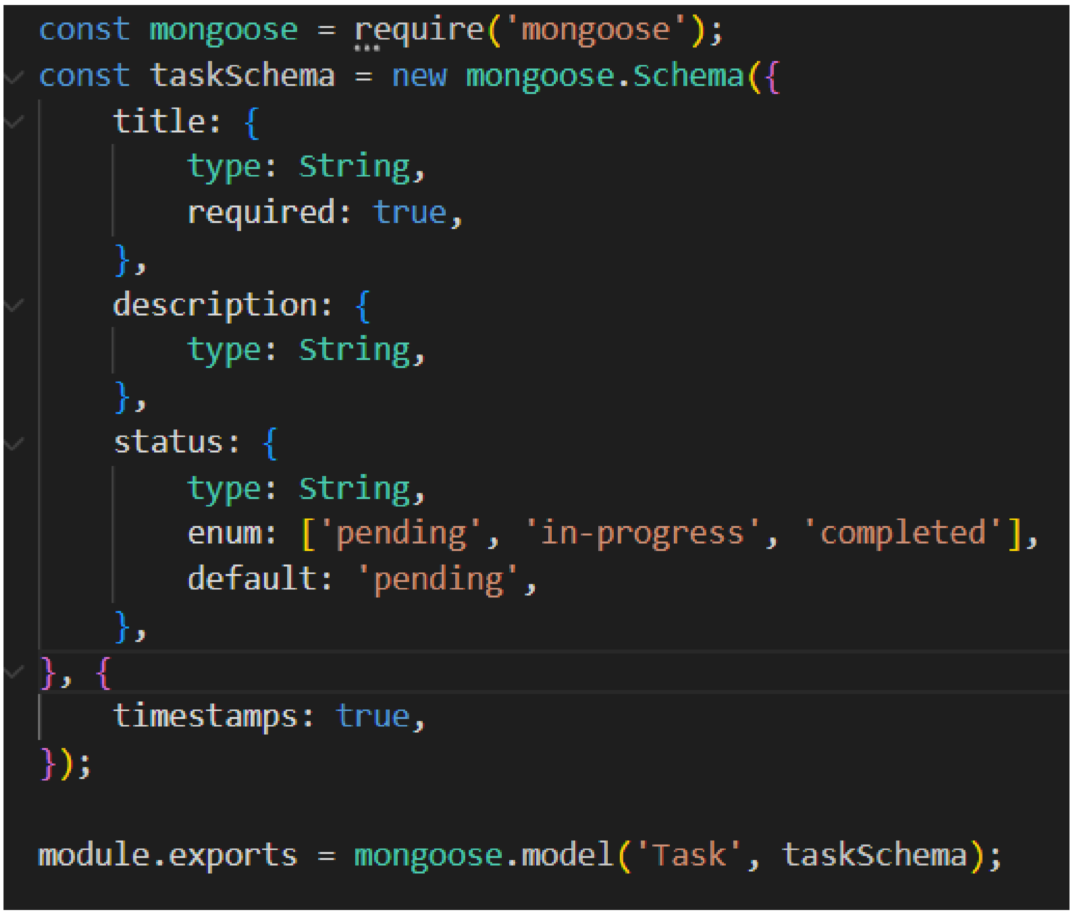
## Code

**db.js**



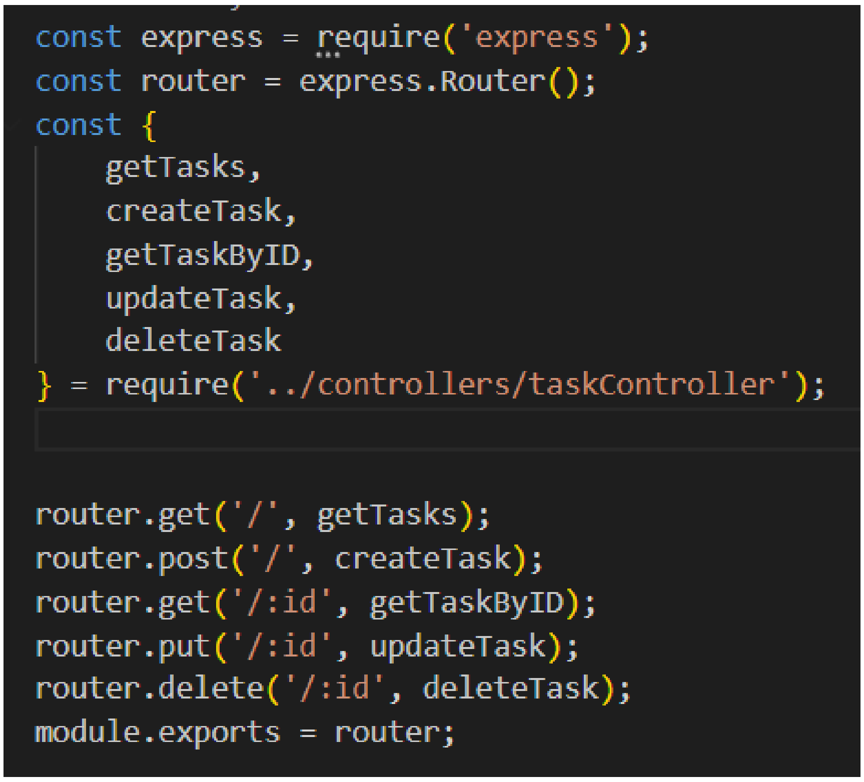
```
const mongoose = require('mongoose');

const connectDB = async () => {
    try{
        await mongoose.connect(process.env.MONGO_URI, {
            useNewUrlParser: true,
            useUnifiedTopology: true,
        });
        console.log('MongoDB Connected Successfully');
        } catch (error) {
            console.error('Mongo Connection Failed:', error.message);
            process.exit(1);
    }
};

module.exports = connectDB;
```

**Task.js**



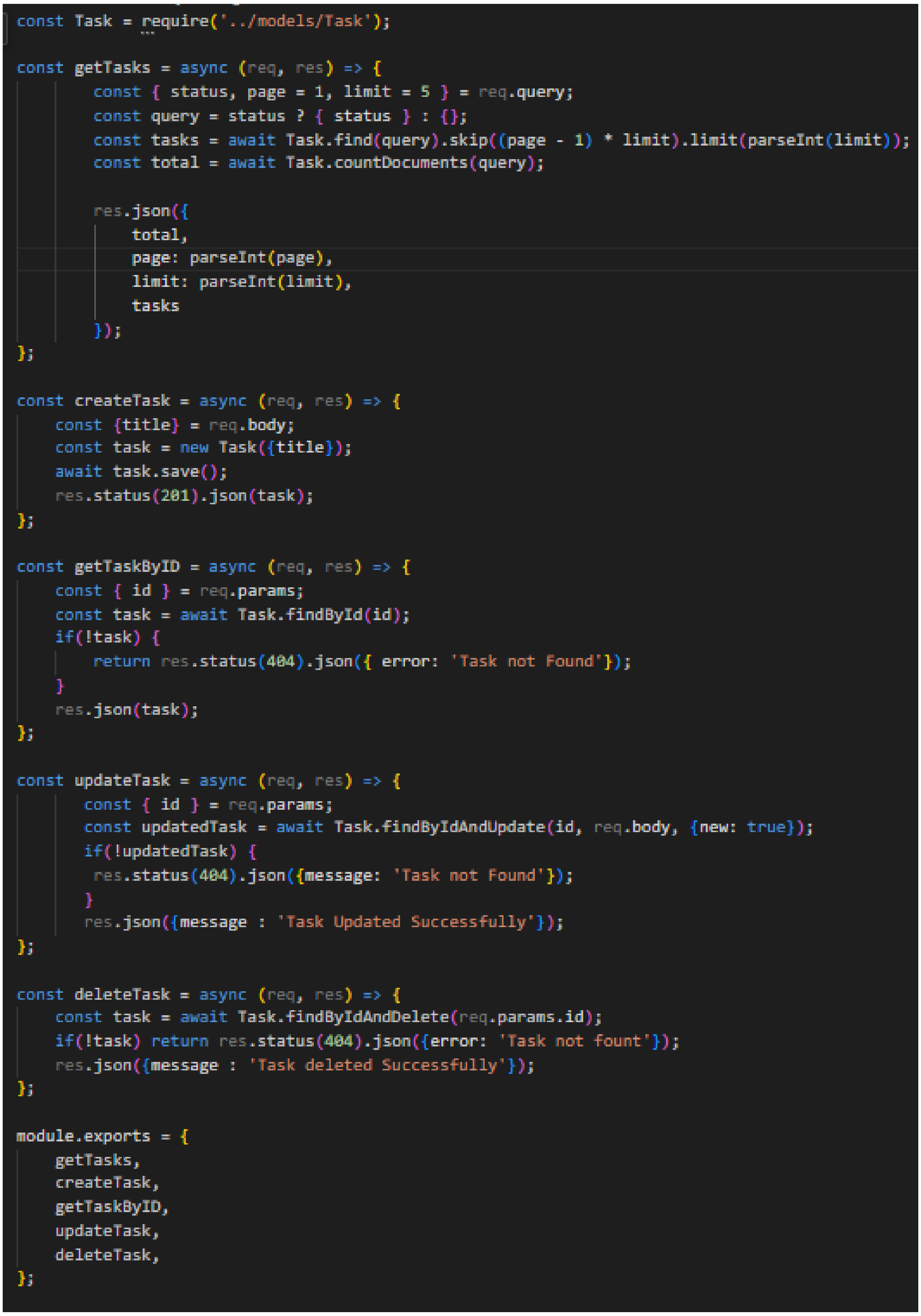
```javascript
const mongoose = require('mongoose');
const taskSchema = new mongoose.Schema({
    title: {
        type: String,
        required: true,
    },
    description: {
        type: String,
    },
    status: {
        type: String,
        enum: ['pending', 'in-progress', 'completed'],
        default: 'pending',
    },
}, {
    timestamps: true,
});

module.exports = mongoose.model('Task', taskSchema);
```

**taskRoutes.js**



```javascript
const express = require('express');
const router = express.Router();
const {
    getTasks,
    createTask,
    getTaskByID,
    updateTask,
    deleteTask
} = require('../controllers/taskController');


router.get('/', getTasks);
router.post('/', createTask);
router.get('/:id', getTaskByID);
router.put('/:id', updateTask);
router.delete('/:id', deleteTask);
module.exports = router;
```

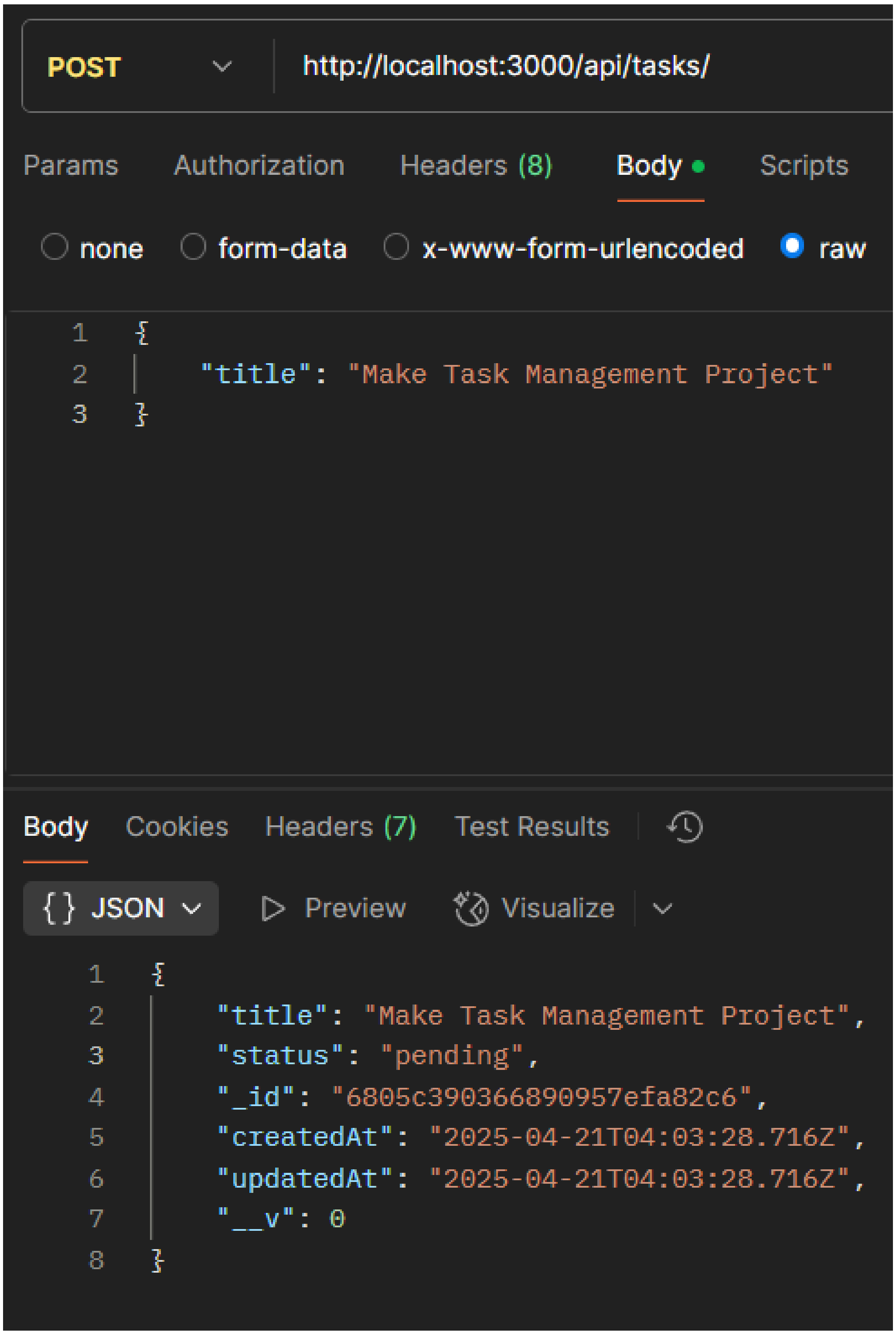
## taskController.js



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

const getTasks = async (req, res) => {
        const { status, page = 1, limit = 5 } = req.query;
        const query = status ? { status } : {};
        const tasks = await Task.find(query).skip((page - 1) * limit).limit(parseInt(limit));
        const total = await Task.countDocuments(query);

        res.json({
            total,
            page: parseInt(page),
            limit: parseInt(limit),
            tasks
        });
};

const createTask = async (req, res) => {
    const {title} = req.body;
    const task = new Task({title});
    await task.save();
    res.status(201).json(task);
};

const getTaskByID = async (req, res) => {
    const { id } = req.params;
    const task = await Task.findById(id);
    if(!task) {
        return res.status(404).json({ error: 'Task not Found'});
    }
    res.json(task);
};

const updateTask = async (req, res) => {
       const { id } = req.params;
       const updatedTask = await Task.findByIdAndUpdate(id, req.body, {new: true});
       if(!updatedTask) {
        res.status(404).json({message: 'Task not Found'});
       }
       res.json({message : 'Task Updated Successfully'});
};

const deleteTask = async (req, res) => {
    const task = await Task.findByIdAndDelete(req.params.id);
    if(!task) return res.status(404).json({error: 'Task not fount'});
    res.json({message : 'Task deleted Successfully'});
};

module.exports = {
    getTasks,
    createTask,
    getTaskByID,
    updateTask,
    deleteTask,
};
```

## Testing using Postman

**POST(Create Task)**



POST http://localhost:3000/api/tasks/

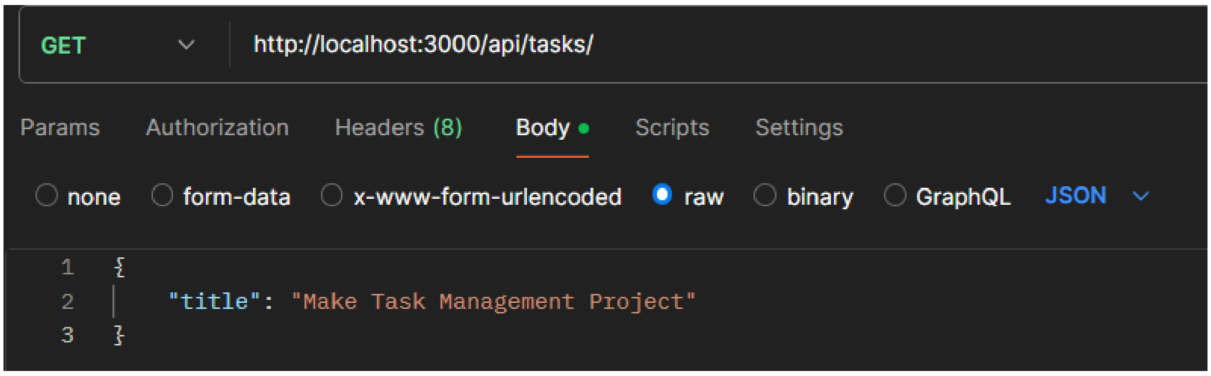
Body: [✓ raw] none form-data x-www-form-urlencoded

```
{
    "title": "Make Task Management Project"
}
```

Response Body (JSON):

```
{
    "title": "Make Task Management Project",
    "status": "pending",
    "_id": "6805c390366890957efa82c6",
    "createdAt": "2025-04-21T04:03:28.716Z",
    "updatedAt": "2025-04-21T04:03:28.716Z",
    "__v": 0
}
```

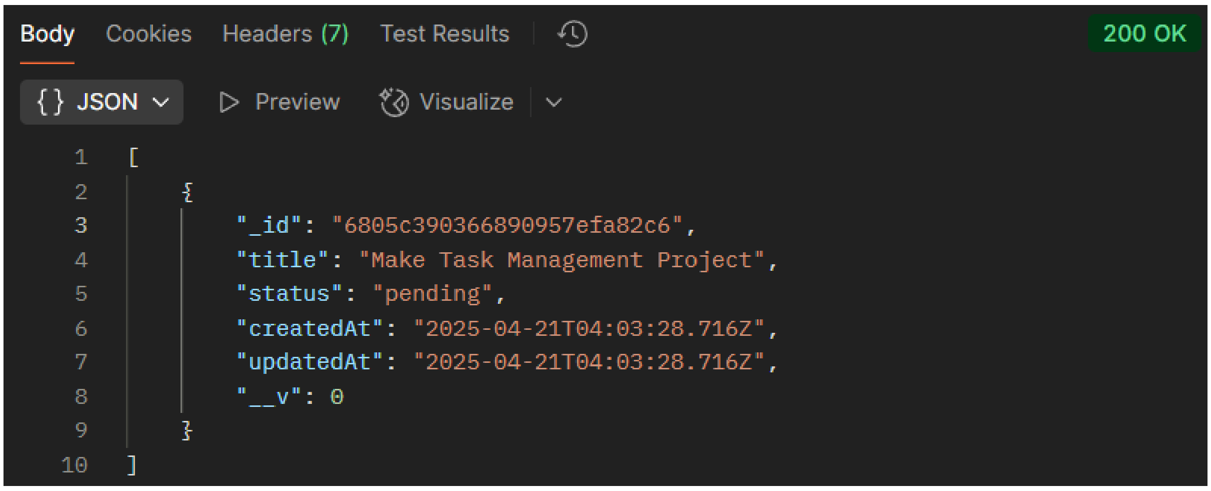
## GET(fetch Task)



http://localhost:3000/api/tasks/

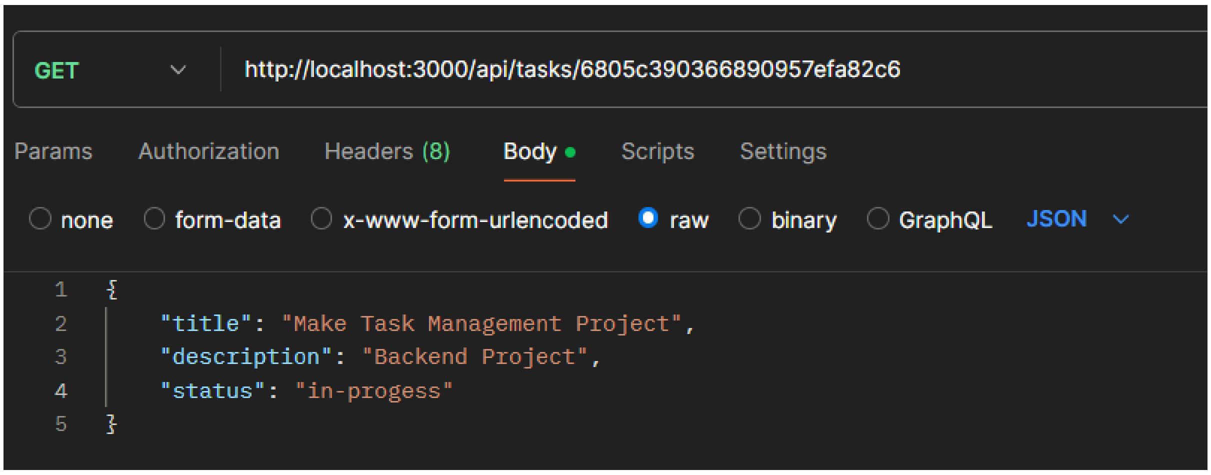
none ○ form-data ○ x-www-form-urlencoded [✓ raw] ○ binary ○ GraphQL JSON

```
{
    "title": "Make Task Management Project"
}
```



```
[
    {
        "_id": "6805c390366890957efa82c6",
        "title": "Make Task Management Project",
        "status": "pending",
        "createdAt": "2025-04-21T04:03:28.716Z",
        "updatedAt": "2025-04-21T04:03:28.716Z",
        "__v": 0
    }
]
```

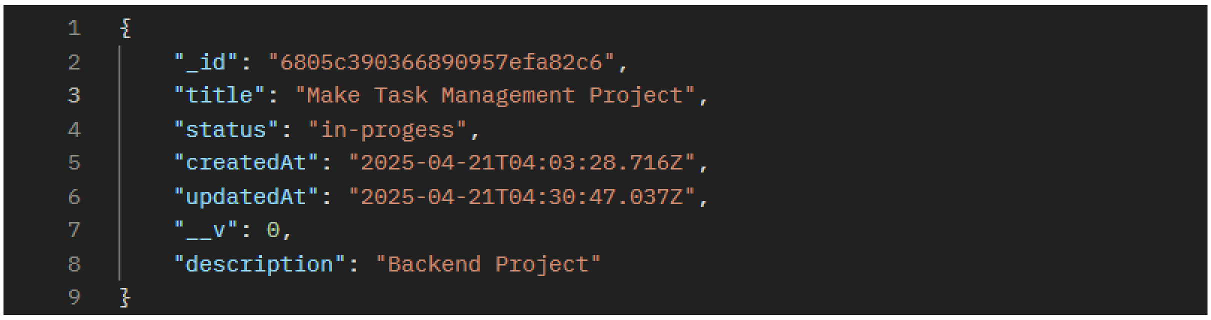
## Fetch by ID



http://localhost:3000/api/tasks/6805c390366890957efa82c6

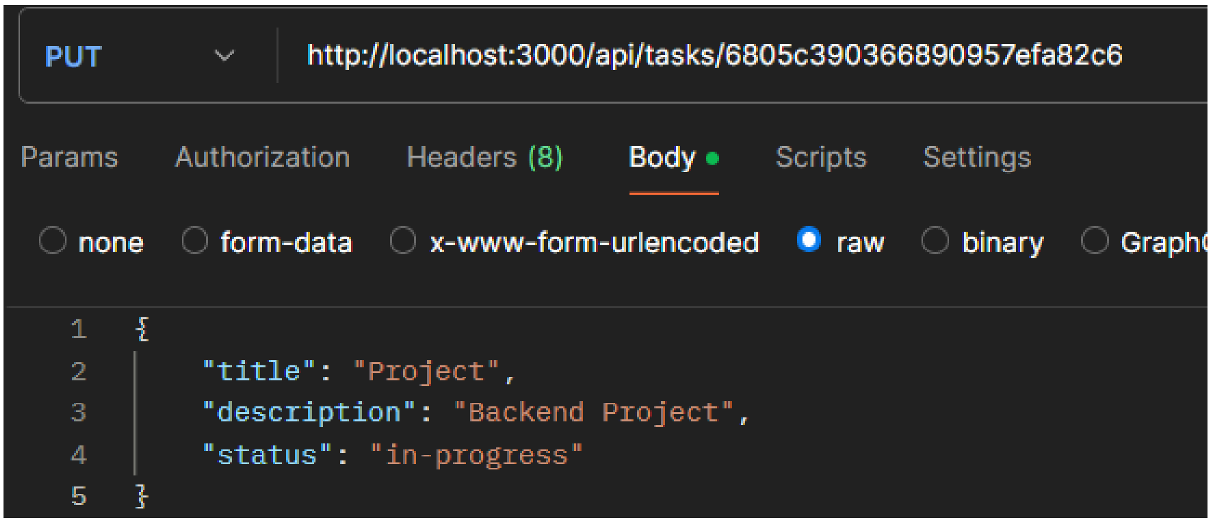
none ○ form-data ○ x-www-form-urlencoded [✓ raw] ○ binary ○ GraphQL JSON

```
{
    "title": "Make Task Management Project",
    "description": "Backend Project",
    "status": "in-progess"
}
```



```
{
    "_id": "6805c390366890957efa82c6",
    "title": "Make Task Management Project",
    "status": "in-progess",
    "createdAt": "2025-04-21T04:03:28.716Z",
    "updatedAt": "2025-04-21T04:30:47.037Z",
    "__v": 0,
    "description": "Backend Project"
}
```

## Update(Update Task Status)

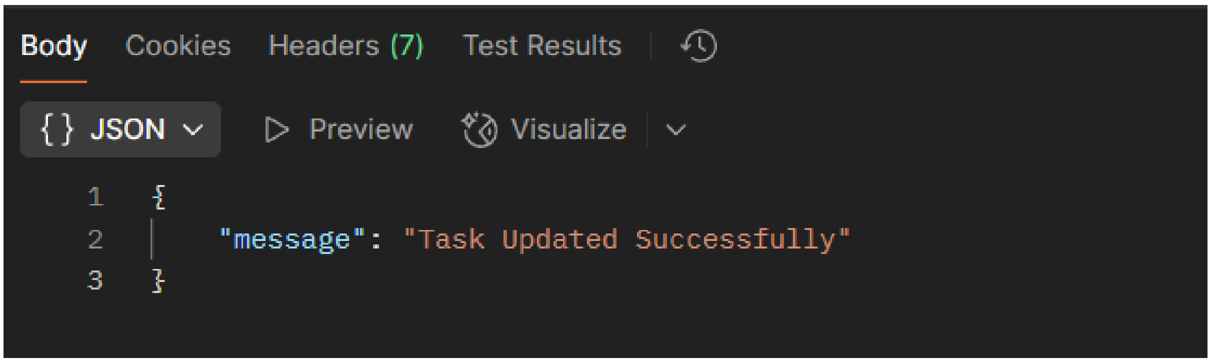


PUT http://localhost:3000/api/tasks/6805c390366890957efa82c6

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded [✓ raw] binary Graph

```
{
    "title": "Project",
    "description": "Backend Project",
    "status": "in-progress"
}
```

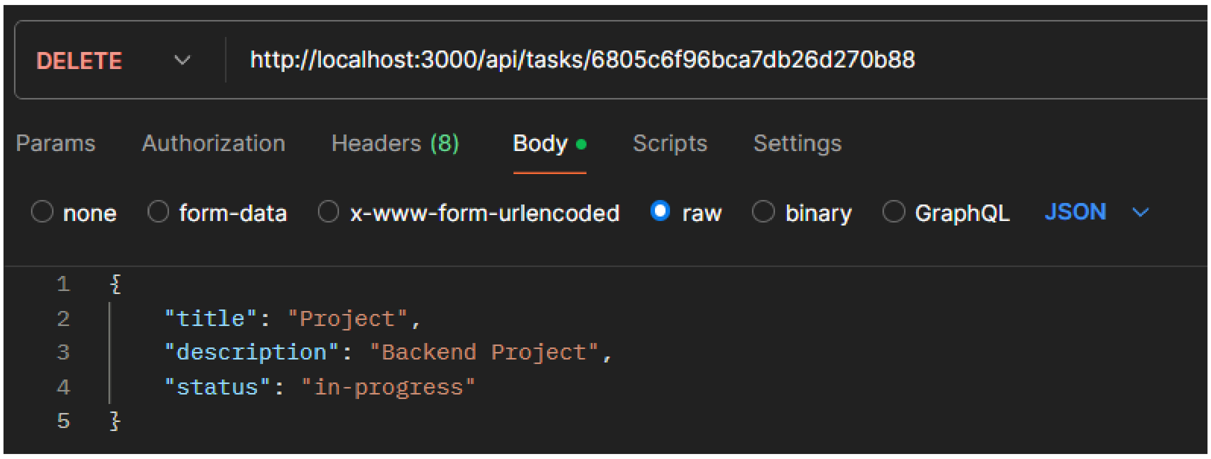


Body Cookies Headers (7) Test Results

{} JSON Preview Visualize

```
{
    "message": "Task Updated Successfully"
}
```

## Delete

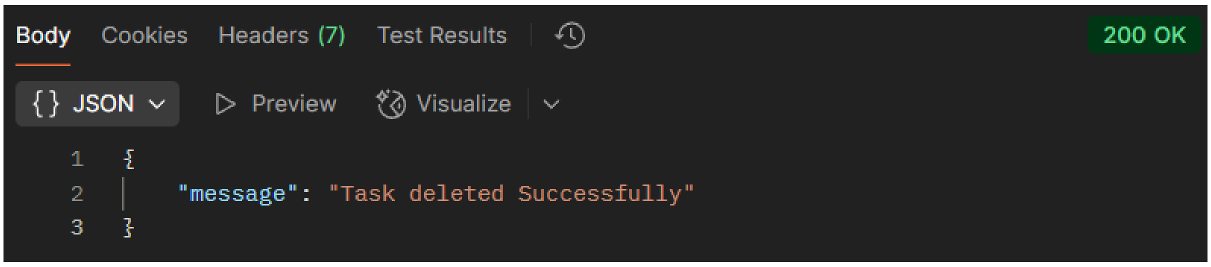


DELETE http://localhost:3000/api/tasks/6805c6f96bca7db26d270b88

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "title": "Project",
    "description": "Backend Project",
    "status": "in-progress"
}
```



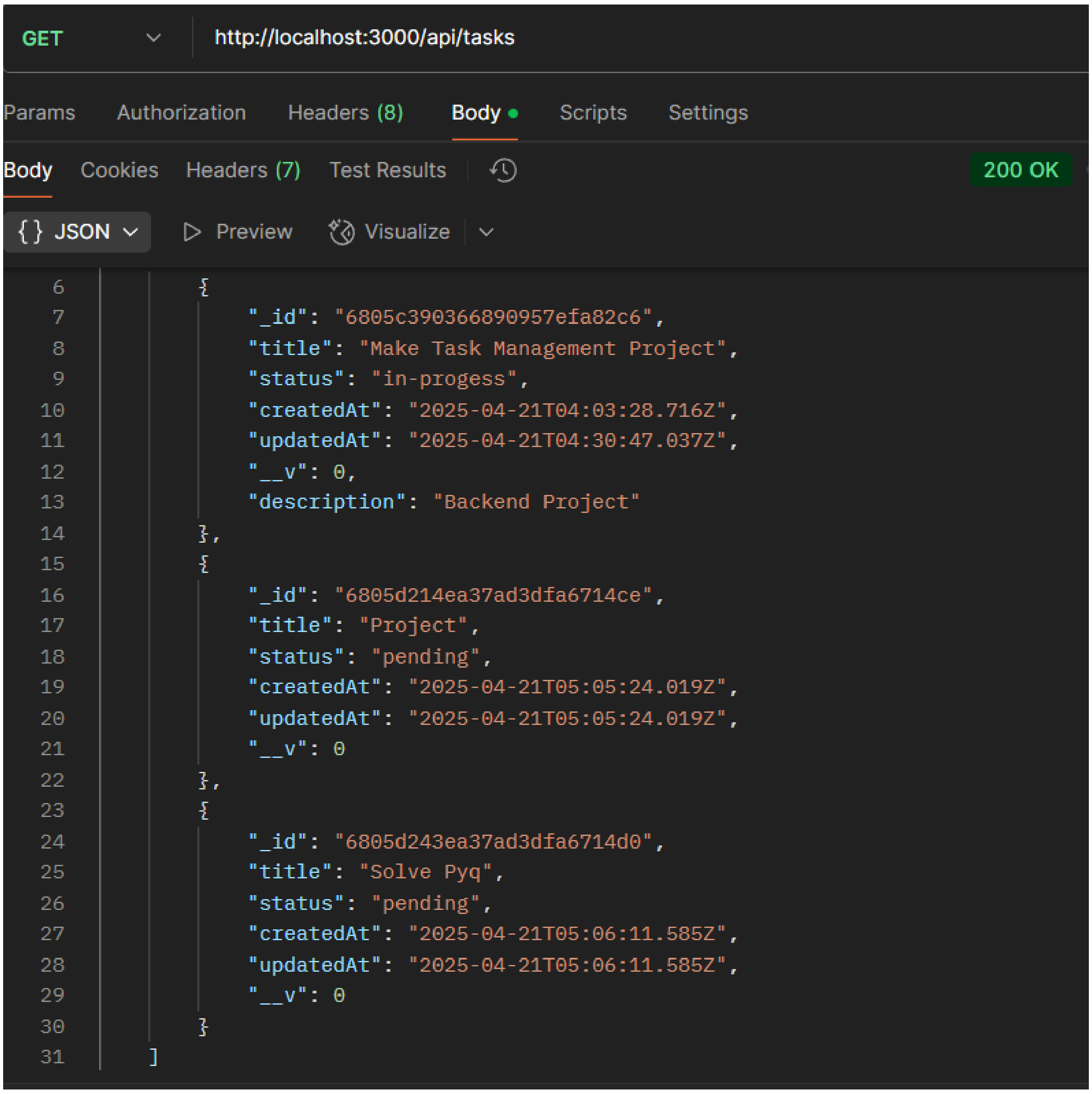
Body Cookies Headers (7) Test Results 200 OK

{} JSON Preview Visualize

```
{
    "message": "Task deleted Successfully"
}
```

## Filter by Status

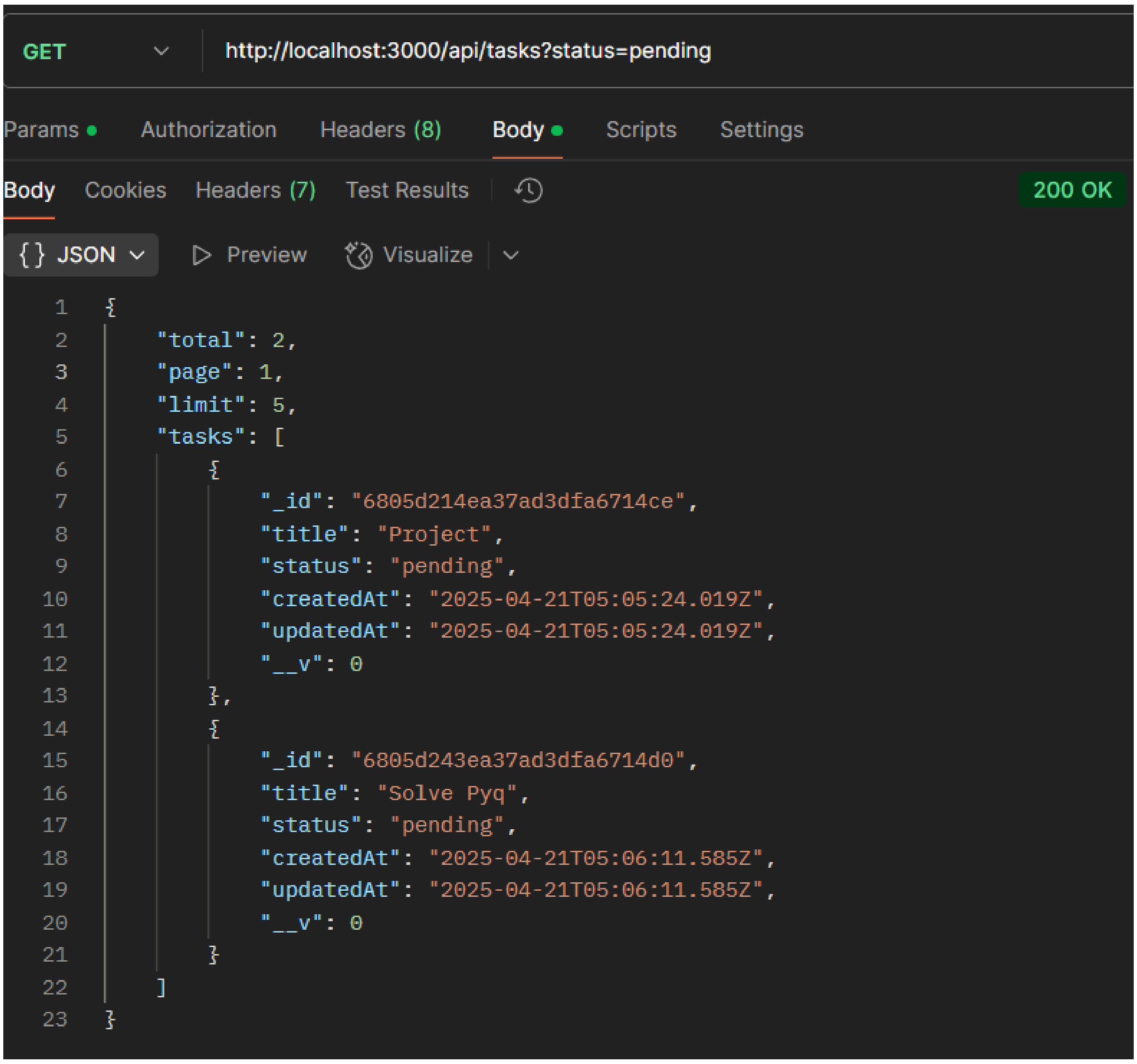
## All tasks



http://localhost:3000/api/tasks

```
        {
            "_id": "6805c390366890957efa82c6",
            "title": "Make Task Management Project",
            "status": "in-progess",
            "createdAt": "2025-04-21T04:03:28.716Z",
            "updatedAt": "2025-04-21T04:30:47.037Z",
            "__v": 0,
            "description": "Backend Project"
        },
        {
            "_id": "6805d214ea37ad3dfa6714ce",
            "title": "Project",
            "status": "pending",
            "createdAt": "2025-04-21T05:05:24.019Z",
            "updatedAt": "2025-04-21T05:05:24.019Z",
            "__v": 0
        },
        {
            "_id": "6805d243ea37ad3dfa6714d0",
            "title": "Solve Pyq",
            "status": "pending",
            "createdAt": "2025-04-21T05:06:11.585Z",
            "updatedAt": "2025-04-21T05:06:11.585Z",
            "__v": 0
        }
    ]
```

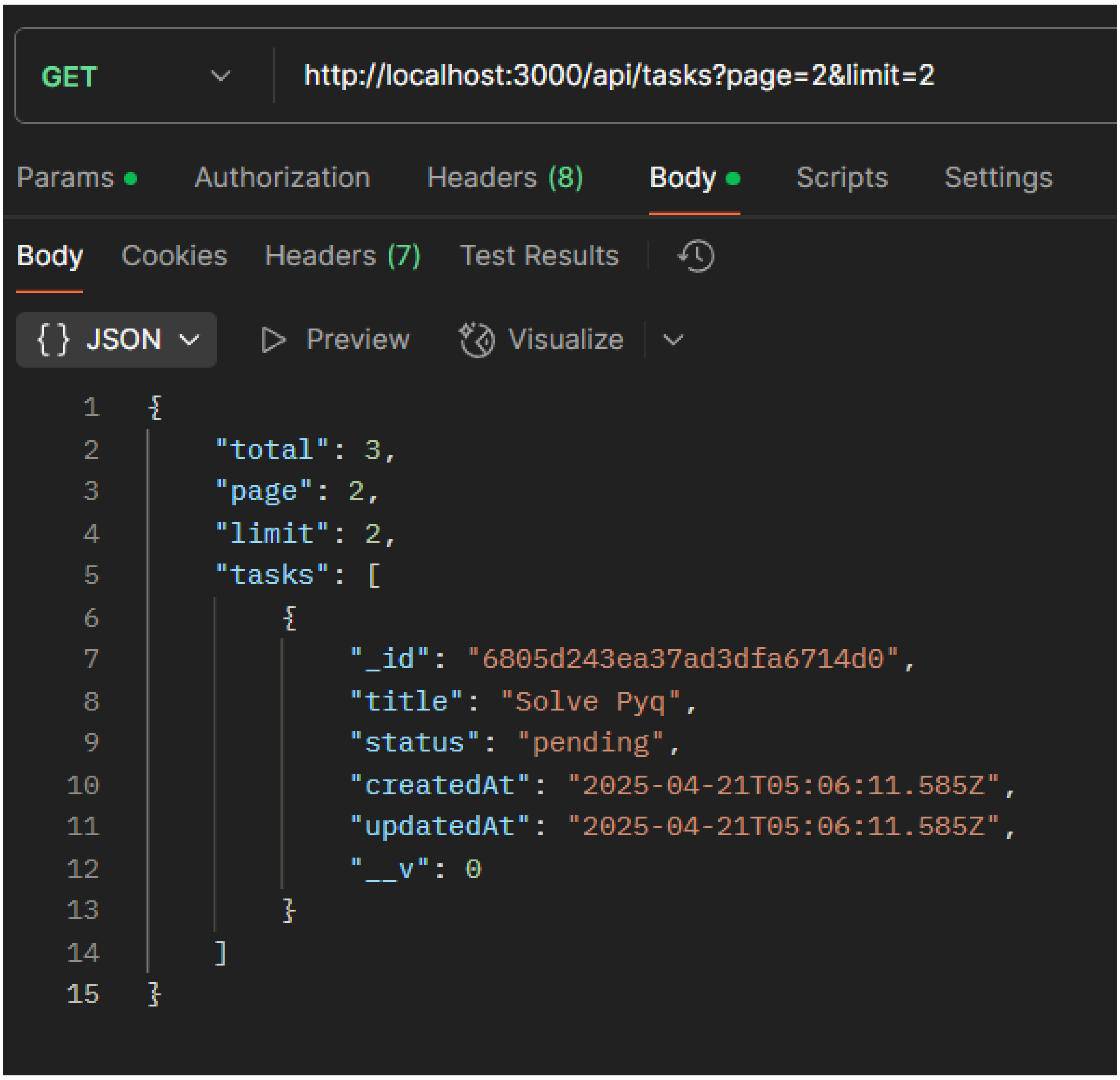
## Filter only Pending Task



```
http://localhost:3000/api/tasks?status=pending
```

```
{
    "total": 2,
    "page": 1,
    "limit": 5,
    "tasks": [
        {
            "_id": "6805d214ea37ad3dfa6714ce",
            "title": "Project",
            "status": "pending",
            "createdAt": "2025-04-21T05:05:24.019Z",
            "updatedAt": "2025-04-21T05:05:24.019Z",
            "__v": 0
        },
        {
            "_id": "6805d243ea37ad3dfa6714d0",
            "title": "Solve Pyq",
            "status": "pending",
            "createdAt": "2025-04-21T05:06:11.585Z",
            "updatedAt": "2025-04-21T05:06:11.585Z",
            "__v": 0
        }
    ]
}
```

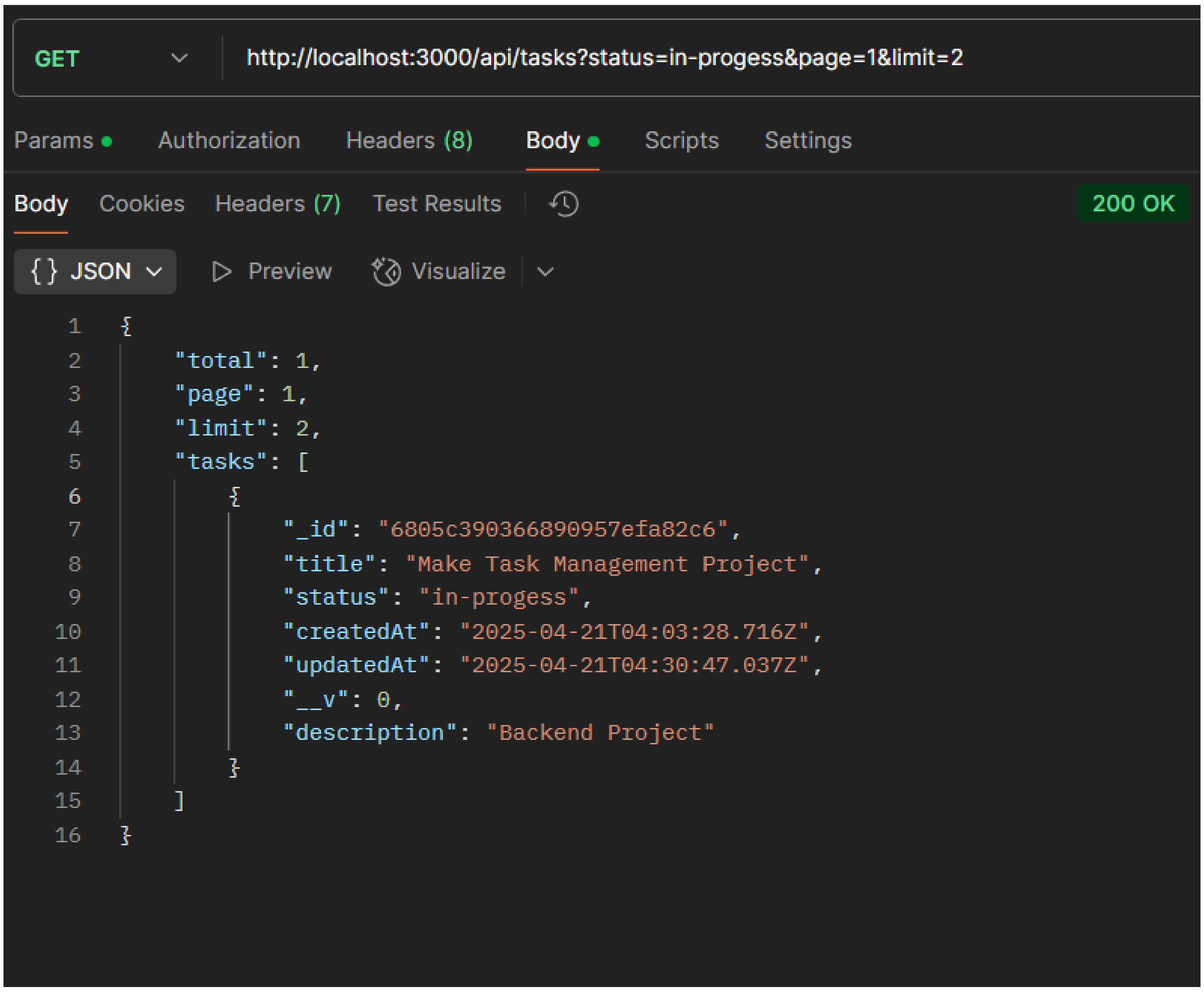
**Pagination**



```
http://localhost:3000/api/tasks?page=2&limit=2
```

```
{
    "total": 3,
    "page": 2,
    "limit": 2,
    "tasks": [
        {
            "_id": "6805d243ea37ad3dfa6714d0",
            "title": "Solve Pyq",
            "status": "pending",
            "createdAt": "2025-04-21T05:06:11.585Z",
            "updatedAt": "2025-04-21T05:06:11.585Z",
            "__v": 0
        }
    ]
}
```

## Both Pagination and Filtering Combined



```
http://localhost:3000/api/tasks?status=in-progess&page=1&limit=2
```

```
{
    "total": 1,
    "page": 1,
    "limit": 2,
    "tasks": [
        {
            "_id": "6805c390366890957efa82c6",
            "title": "Make Task Management Project",
            "status": "in-progess",
            "createdAt": "2025-04-21T04:03:28.716Z",
            "updatedAt": "2025-04-21T04:30:47.037Z",
            "__v": 0,
            "description": "Backend Project"
        }
    ]
}
```