

GET request

➔ localhost:3000/api/users

<https://www.prisma.io/docs/orm/overview/prisma-in-your-stack/rest>

REST API server example

GET

```
app.get('/feed', async (req, res) => {  
  const posts = await prisma.post.findMany({  
    where: { published: true },  
    include: { author: true },  
  })  
  res.json(posts)  
})
```

Note that the `feed` endpoint in this case returns a nested
Here's a sample response:

```
[  
  {  
    "id": "21",  
    "title": "Hello World",  
    "content": "null",  
    "published": "true",  
    "authorId": 42,  
    "author": {  
      "id": "42",  
      "name": "Alice",  
      "email": "alice@prisma.io"  
    }  
  }  
]
```

POST

```
app.post('/post', async (req, res) => {
  const { title, content, authorEmail } = req.body
  const result = await prisma.post.create({
    data: {
      title,
      content,
      published: false,
      author: { connect: { email: authorEmail } },
    },
  })
  res.json(result)
})
```

PUT

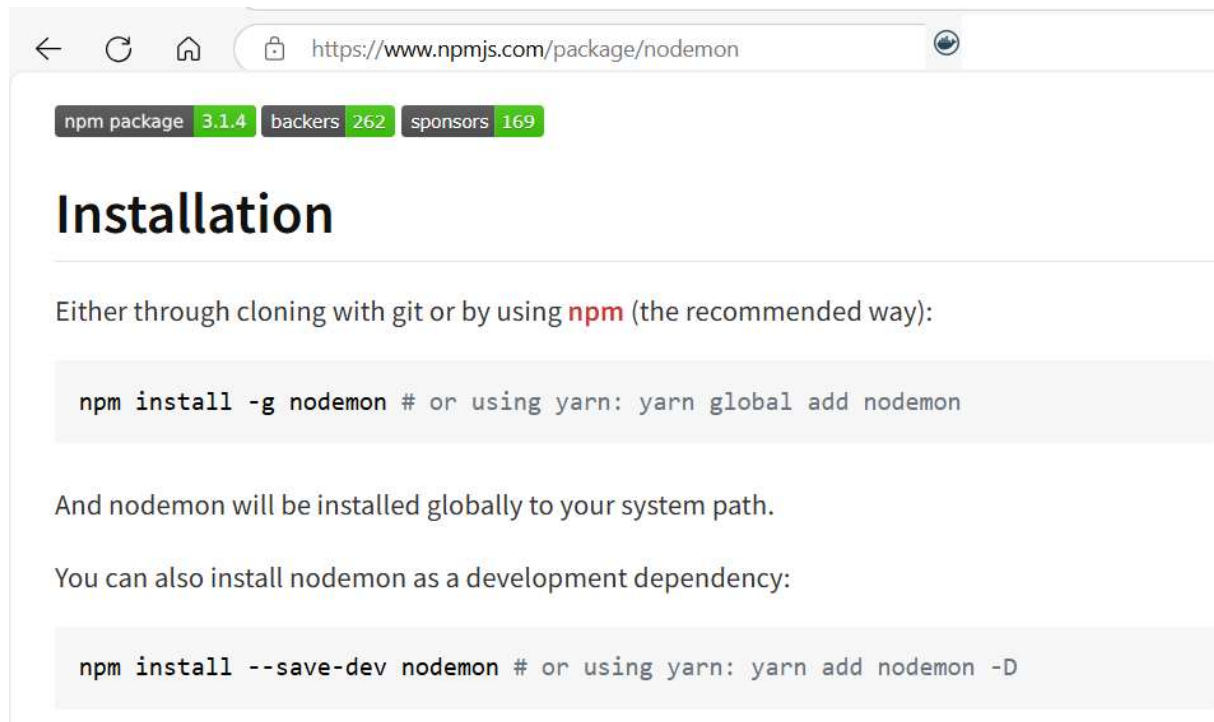
```
app.put('/publish/:id', async (req, res) => {
  const { id } = req.params
  const post = await prisma.post.update({
    where: { id: Number(id) },
    data: { published: true },
  })
  res.json(post)
})
```

DELETE

```
app.delete('/post/:id', async (req, res) => {
  const { id } = req.params
  const post = await prisma.post.delete({
    where: {
      id: Number(id),
    },
  })
  res.json(post)
})
```

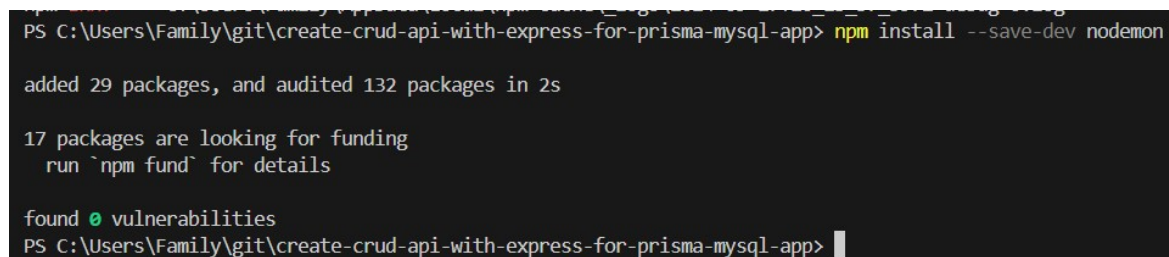
➤ Install nodemon dependency

<https://www.npmjs.com/package/nodemon>



The screenshot shows the npmjs.com page for the nodemon package. At the top, it displays 'npm package 3.1.4', 'backers 262', and 'sponsors 169'. The main heading is 'Installation'. Below it, the text says 'Either through cloning with git or by using **npm** (the recommended way):'. A code block shows the command: `npm install -g nodemon # or using yarn: yarn global add nodemon`. Below this, it says 'And nodemon will be installed globally to your system path.' and 'You can also install nodemon as a development dependency:'. Another code block shows the command: `npm install --save-dev nodemon # or using yarn: yarn add nodemon -D`.

C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app> **npm install --save-dev nodemon**



The screenshot shows the terminal output of the command `npm install --save-dev nodemon`. The output is as follows:

```
PS C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app> npm install --save-dev nodemon
added 29 packages, and audited 132 packages in 2s

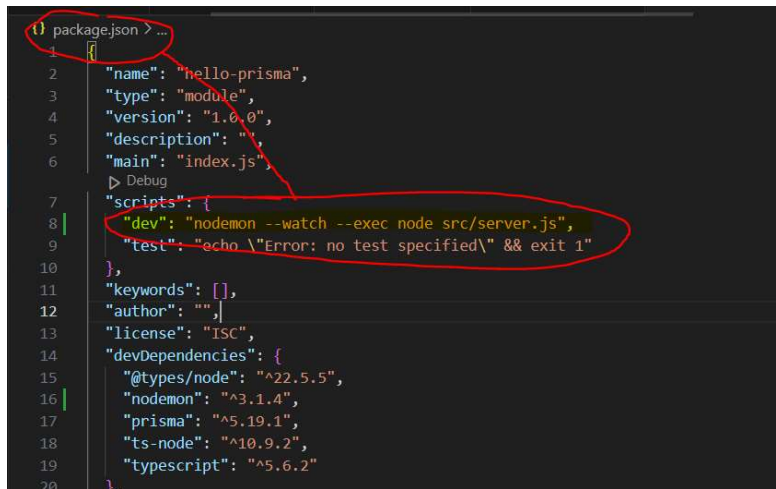
17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app> |
```

➔ Nodemon dependency was installed – OK

On package.json add:

"dev": "nodemon --watch --exec node src/server.js",



```
1 {} package.json > ...
2 {
3   "name": "hello-prisma",
4   "type": "module",
5   "version": "1.0.0",
6   "description": "",
7   "main": "index.js",
8   "scripts": {
9     "dev": "nodemon --watch --exec node src/server.js",
10    "test": "echo \\\"Error: no test specified\\\" && exit 1"
11  },
12  "keywords": [],
13  "author": "",
14  "license": "ISC",
15  "devDependencies": {
16    "@types/node": "^22.5.5",
17    "nodemon": "^3.1.4",
18    "prisma": "^5.19.1",
19    "ts-node": "^10.9.2",
20    "typescript": "^5.6.2"
21  }
22 }
```

package.json

```
{
  "name": "hello-prisma",
  "type": "module",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon --watch --exec node src/server.js",
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
}
```

Server.js

```
import { PrismaClient } from '@prisma/client';
import express from 'express';

const app = express();
const prisma = new PrismaClient();

// custom middleware
app.use((req, res, next) => {
  console.log(`${req.url} ${new Date()}`);
  next(); // call the next middleware in the stack
})

app.get('/', (req, res) => {
  res.send('HELLO WROLD')
})

// GET api/course
app.get('/api/users', async (req, res) => {
  // fetch all the users from db
  const allUsers = await prisma.user.findMany({
    include: {
      posts: true,
      profile: true,
    },
  })
  // get the prisma

  // get the course from the prisma object

  // call the many method

  // send back to user as a json
  res.status(200);
  res.json(allUsers);
});

app.listen(3000, () => {
  console.log('Server is running at port 3000');
})
```

➔ RE-RUN SERVER...

(As long as change something in the server.js. We need to re-run the server.)

➔ Ctrl + C ➔ Type ➔ J ➔ Stop the Server

```
tle":"Hello World","content":null,"published":true,"authorId":1}
Batchvorgang abbrechen (J/N)? j
PS C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app>
```

➔ RUN SERVER... | npm run dev

```
updated POST: {"id":1,"createdAt":"2024-09-15T21:49:30.671Z","updatedAt":"2024-09-18T11:50:34.805Z",
tle":"Hello World","content":null,"published":true,"authorId":1}
Batchvorgang abbrechen (J/N)? j
PS C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app> npm run dev

> hello-prisma@1.0.0 dev
> nodemon --watch --exec node src/server.js

[nodemon] 3.1.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): --exec
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node node src/server.js`
Server is running at port 3000
```

package.json

```
{
  "name": "hello-prisma",
  "type": "module",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon --watch --exec node src/server.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
}
```

```
PS C:\Users\Family\git\create-crud-api-with-express-for-prisma-mysql-app> npm run dev
```

```
> hello-prisma@1.0.0 dev
```

```
> nodemon --watch --exec node src/server.js
```

```
[nodemon] 3.1.4
```

```
[nodemon] to restart at any time, enter `rs`
```

```
[nodemon] to restart at any time, enter `rs`
```

```
[nodemon] watching path(s): --exec
```

```
[nodemon] watching extensions: js,mjs,cjs,json
```

```
[nodemon] watching extensions: js,mjs,cjs,json
```

```
[nodemon] starting `node node src/server.js`
```

```
Server is running at port 3000
```

```
/api/users Tue Sep 17 2024 12:43:03 GMT+0200 (Central European Summer Time)
```

```
□
```

Thunder Client | http Client



<https://www.prisma.io/docs/orm/overview/prisma-in-your-stack/rest>

REST API server example

GET

```
app.get('/feed', async (req, res) => {
  const posts = await prisma.post.findMany({
    where: { published: true },
    include: { author: true },
  })
  res.json(posts)
})
```

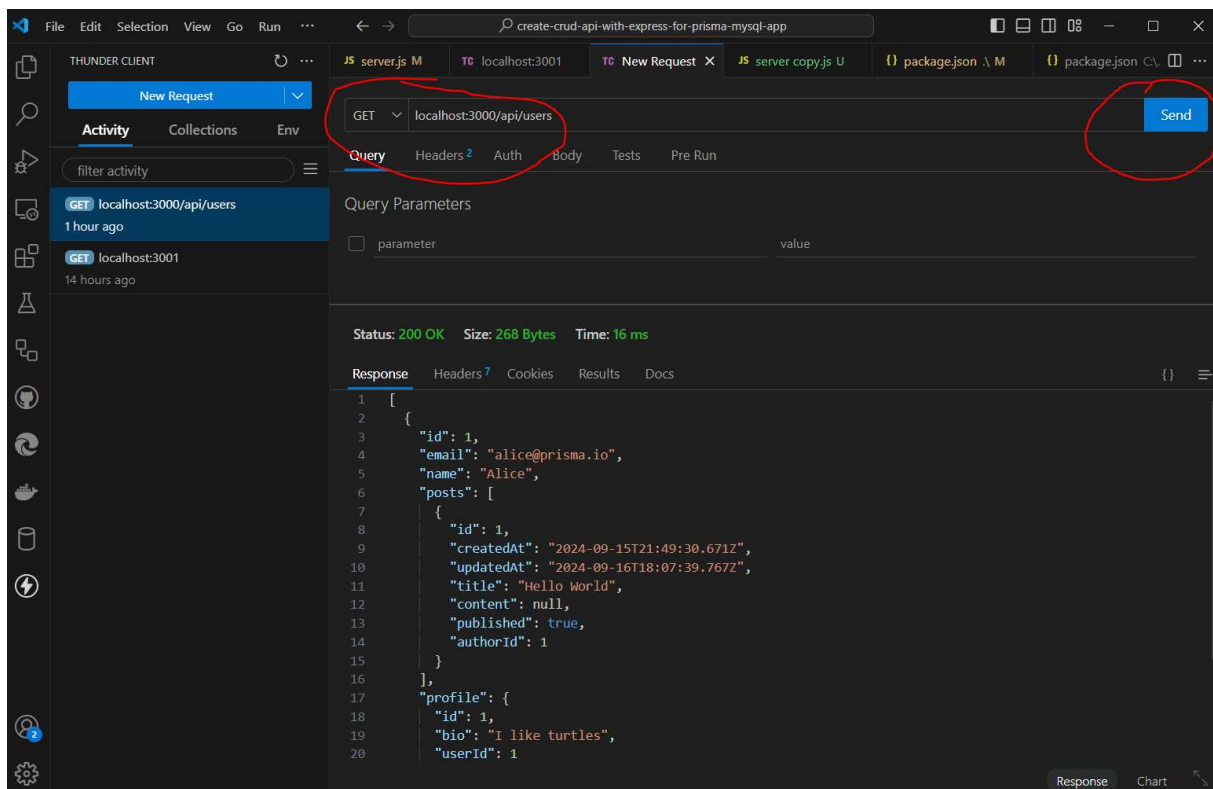
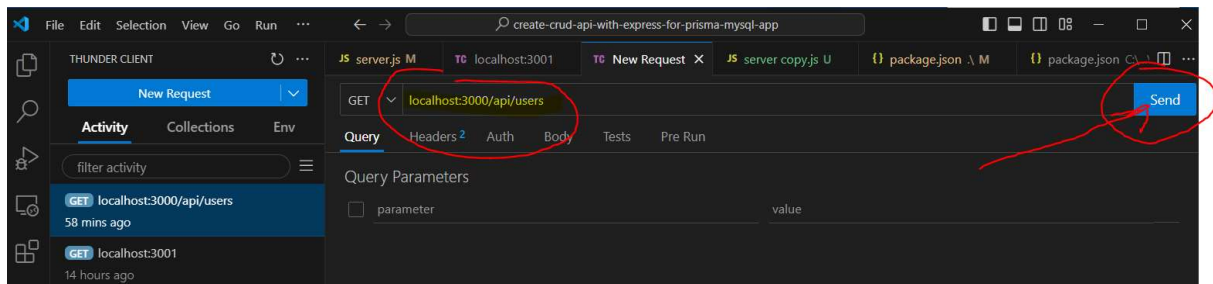
Note that the `feed` endpoint in this case returns a nested JSON response of `Post` objects that *include* an `author` object. Here's a sample response:

```
[
  {
    "id": "21",
    "title": "Hello World",
    "content": "null",
    "published": "true",
    "authorId": 42,
    "author": {
      "id": "42",
      "name": "Alice",
      "email": "alice@prisma.io"
    }
  }
]
```


Server.js

```
16
17 // GET api/course
18 app.get('/api/users', async (req,res) => {
19   // fetch all the users from db
20   const allUsers = await prisma.user.findMany({
21     include: {
22       posts: true,
23       profile: true,
24     },
25   })
26 }
```

GET: **localhost:3000/api/users** → Click **send** button



Status 200 OK → 😊 → Get response data as below.

```
Status: 200 OK   Size: 268 Bytes   Time: 16 ms

Response  Headers 7  Cookies  Results  Docs

1  [
2  {
3    "id": 1,
4    "email": "alice@prisma.io",
5    "name": "Alice",
6    "posts": [
7      {
8        "id": 1,
9        "createdAt": "2024-09-15T21:49:30.671Z",
10       "updatedAt": "2024-09-16T18:07:39.767Z",
11       "title": "Hello World",
12       "content": null,
13       "published": true,
14       "authorId": 1
15     }
16   ],
17   "profile": {
18     "id": 1,
19     "bio": "I like turtles",
20     "userId": 1
21   }
22 }
23 ]
```

```
[
  {
    "id": 1,
    "email": "alice@prisma.io",
    "name": "Alice",
    "posts": [
      {
        "id": 1,
        "createdAt": "2024-09-15T21:49:30.671Z",
        "updatedAt": "2024-09-16T18:07:39.767Z",
        "title": "Hello World",
        "content": null,
        "published": true,
        "authorId": 1
      }
    ],
    "profile": {
      "id": 1,
      "bio": "I like turtles",
      "userId": 1
    }
  }
]
```

GET: localhost:3000/api/users

GET Request OK 😊