Welcome to the GitHub repository for the interview task provided in this Notion link. To get started, follow these instructions:

Install Node.js

How you install Node.js varies depending on your operating system.

Operating System	Instructions
OS X	The easiest way to install Node.js on OS X is to use the official installer from nodejs.org . You can also use Homebrew if you prefer. To manage and switch between versions of Node.js on your machine, we recommend using nvm .
Windows	The easiest way to install Node.js on Windows is the official installer from nodejs.org . You can also use Chocolatey if you prefer. To manage and switch between versions of Node.js on your machine, we recommend using nvm-windows .
Linux	The Node.js installation method varies by distribution. To manage and switch between versions of Node.js on your machine, we recommend using nvm .

Install a text editor or IDE

Before we can start a Node.js project, we'll need a place to write our code.

If you already have a code-writing tool of choice, you can stick with it for developing your Node.js application. If you're looking for something new, we recommend trying out a few options:

- Visual Studio Code is currently the most popular Integrated Development Environment (IDE) used for JavaScript projects. It's a fast, free editor and debugger that runs on all platforms and comes with many helpful tools already installed.
- WebStorm is another extremely powerful IDE, built on the open-source IntelliJ Platform. It is free to try, but requires a paid license after 30 days.
- Node.js Tools for Visual Studio is a great option if you're already a Visual Studio user.

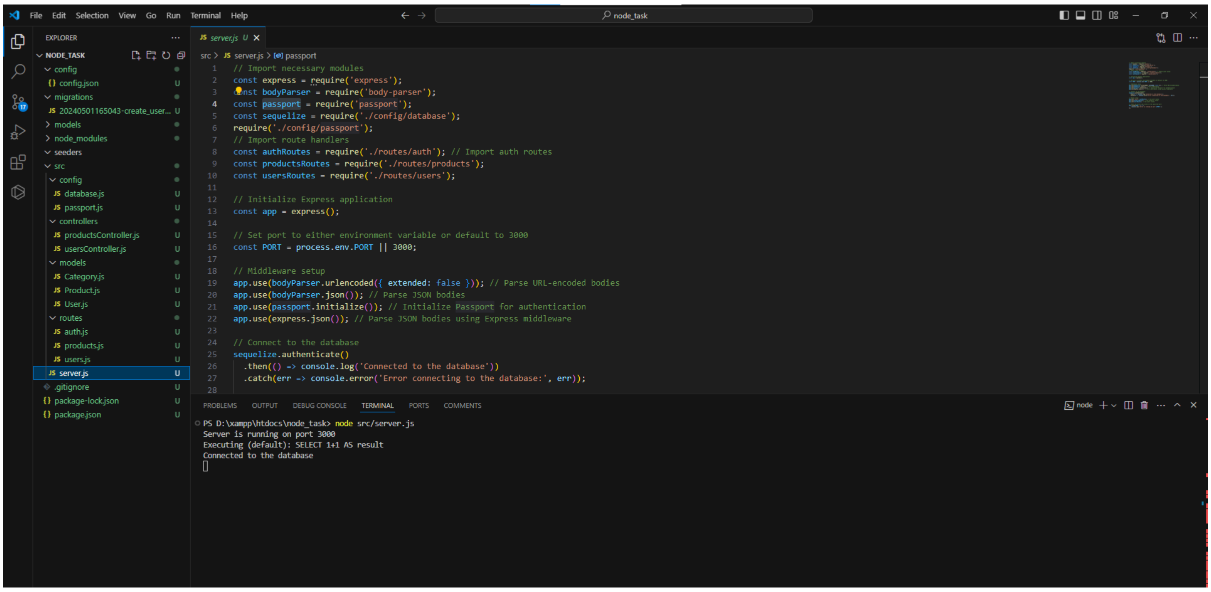
If you're new to programming, we highly recommend getting off to a good start with Visual Studio Code.

Clone this repository using the following command:

```
git clone https://github.com/ragubathi/node_task.git
```

1. Open your Visual Studio Code (VS Code) terminal and navigate to the cloned repository.
2. Run the following commands sequentially:
   - `npm install`
   - `sequelize db:migrate`
   - `node src/server.js`
3. Once the server is up and running, import the `Node tasks.postman_collection.json` file into your Postman collection.
4. You can now test the endpoints by hitting them through Postman.

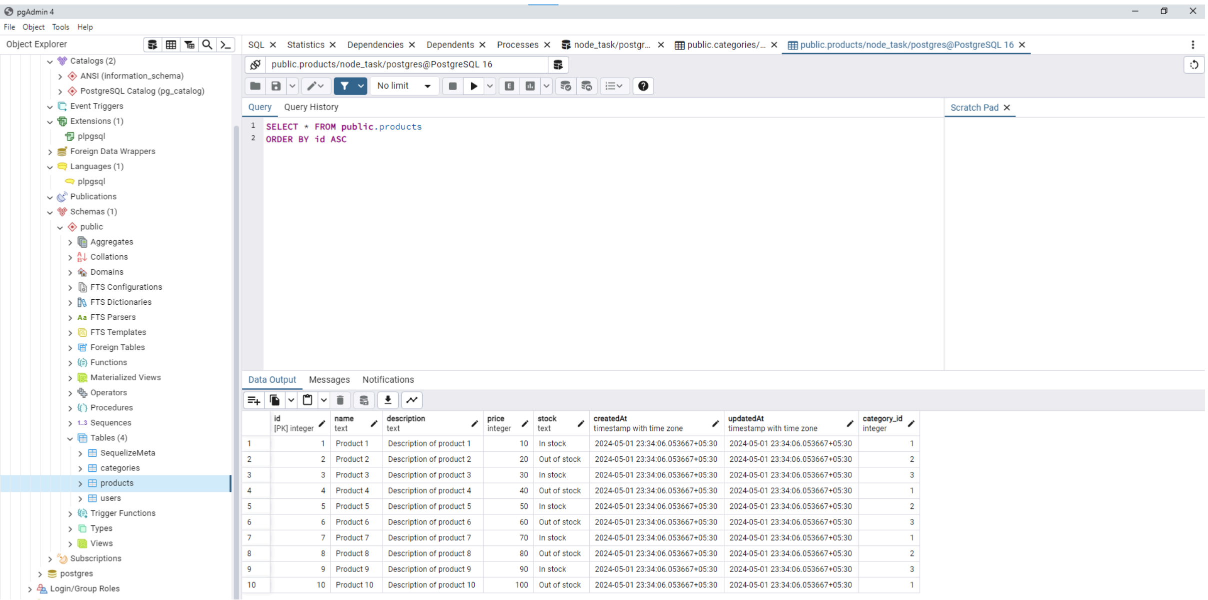
Feel free to explore and analyze the code. If you have any questions or feedback, please don't hesitate to reach out.



```
// Import necessary modules
const express = require('express');
const bodyParser = require('body-parser');
const passport = require('passport');
const sequelize = require('./config/database');
require('./config/passport');
// Import route handlers
const authRoutes = require('./routes/auth'); // Import auth routes
const productsRoutes = require('./routes/products');
const usersRoutes = require('./routes/users');

// Initialize Express application
const app = express();

// Set port to either environment variable or default to 3000
const PORT = process.env.PORT || 3000;

// Middleware setup
app.use(bodyParser.urlencoded({ extended: false })); // Parse URL-encoded bodies
app.use(bodyParser.json()); // Parse JSON bodies
app.use(passport.initialize()); // Initialize Passport for authentication
app.use(express.json()); // Parse JSON bodies using Express middleware

// Connect to the database
sequelize.authenticate()
  .then(() => console.log('Connected to the database'))
  .catch(err => console.error('Error connecting to the database:', err));
```

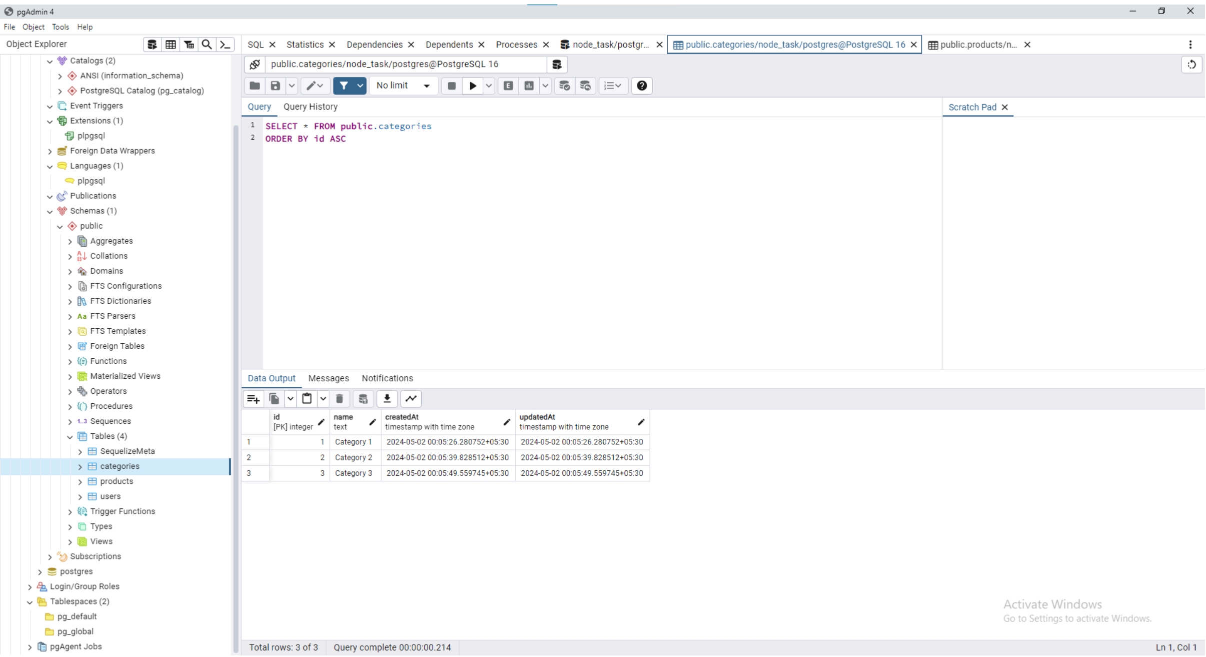
```
PS D:\xampp\htdocs\node_task> node src/server.js
Server is running on port 3000
Executing (default): SELECT 1+1 AS result
Connected to the database
```

For Question 1:



```sql
SELECT * FROM public.products
ORDER BY id ASC
```

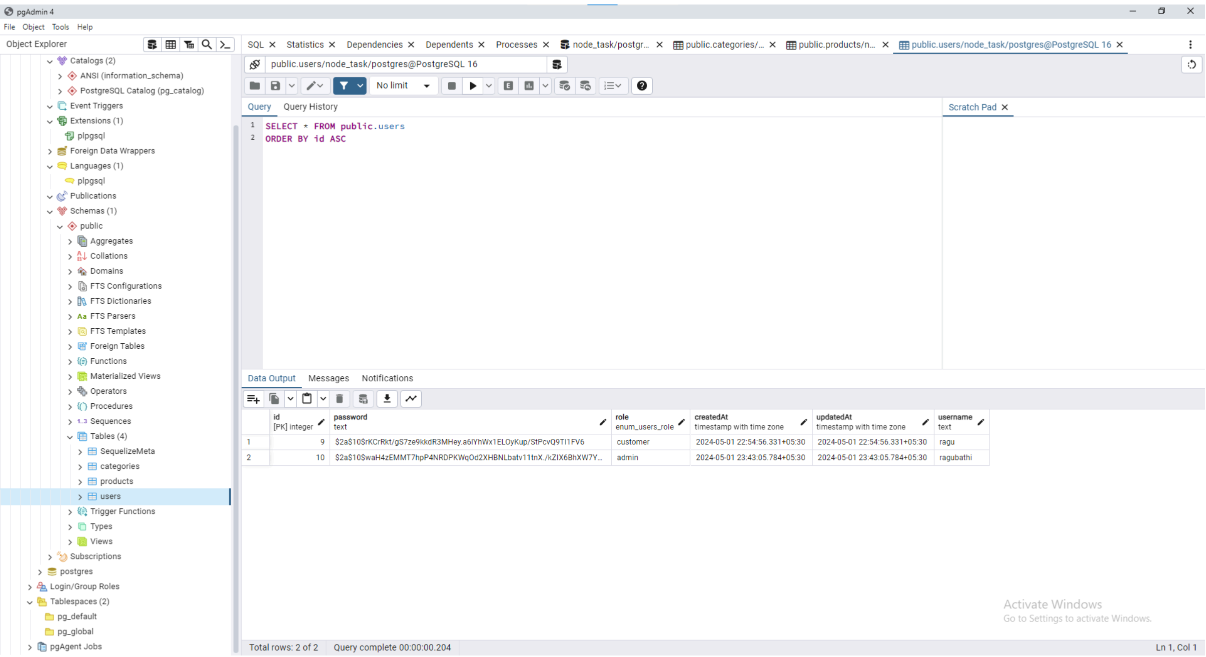
	id [PK] integer	name text	description text	price integer	stock text	createdAt timestamp with time zone	updatedAt timestamp with time zone	category_id integer
1	1	Product 1	Description of product 1	10	In stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	1
2	2	Product 2	Description of product 2	20	Out of stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	2
3	3	Product 3	Description of product 3	30	In stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	3
4	4	Product 4	Description of product 4	40	Out of stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	1
5	5	Product 5	Description of product 5	50	In stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	2
6	6	Product 6	Description of product 6	60	Out of stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	3
7	7	Product 7	Description of product 7	70	In stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	1
8	8	Product 8	Description of product 8	80	Out of stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	2
9	9	Product 9	Description of product 9	90	In stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	3
10	10	Product 10	Description of product 10	100	Out of stock	2024-05-01 23:34:06.053667+05:30	2024-05-01 23:34:06.053667+05:30	1



```sql
SELECT * FROM public.categories
ORDER BY id ASC
```

	id [PK] integer	name text	createdAt timestamp with time zone	updatedAt timestamp with time zone
1	1	Category 1	2024-05-02 00:05:26.280752+05:30	2024-05-02 00:05:26.280752+05:30
2	2	Category 2	2024-05-02 00:05:39.828512+05:30	2024-05-02 00:05:39.828512+05:30
3	3	Category 3	2024-05-02 00:05:49.559745+05:30	2024-05-02 00:05:49.559745+05:30

```
Total rows: 3 of 3    Query complete 00:00:00.214                                    Ln 1, Col 1
```



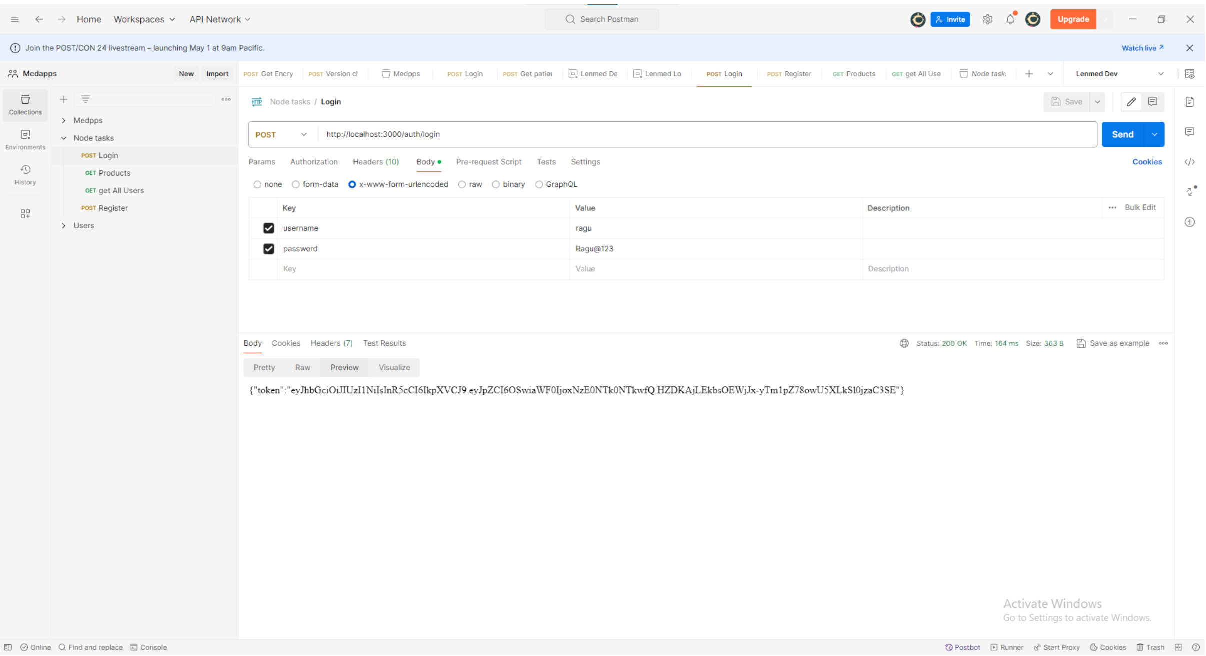
```
SELECT * FROM public.users
ORDER BY id ASC
```

	id [PK] integer	password text	role enum_users_role	createdAt timestamp with time zone	updatedAt timestamp with time zone	username text
1	9	$2a$10$rKCrRkt/gS7ze9kkdR3MHey.a6lYhWx1ELOyKup/StPcvQ9TI1FV6	customer	2024-05-01 22:54:56.331+05:30	2024-05-01 22:54:56.331+05:30	ragu
2	10	$2a$10$waH4zEMMT7hpP4NRDPKWqOd2XHBNLbatv11tnX./kZIX6BhXW7Y...	admin	2024-05-01 23:43:05.784+05:30	2024-05-01 23:43:05.784+05:30	ragubathi

Total rows: 2 of 2 Query complete 00:00:00.204

For Question 2:

Login endpoint to login as user / customer and get token to access /products endpoint



POST http://localhost:3000/auth/login

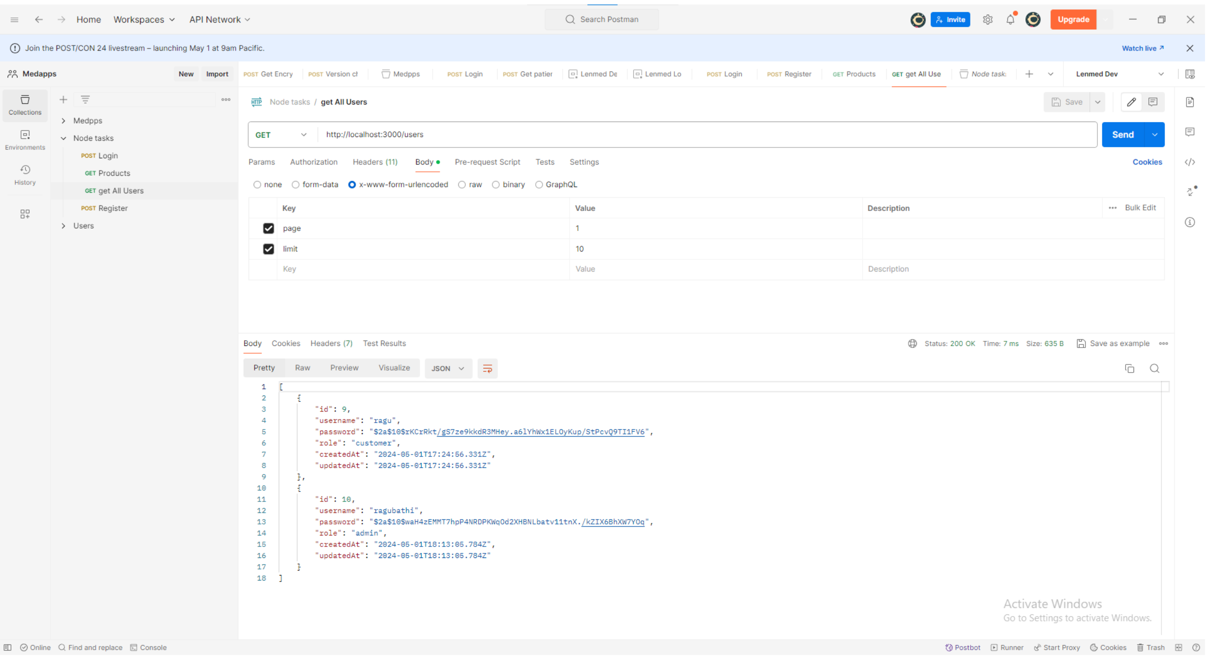
Body: ● x-www-form-urlencoded

	Key	Value	Description
☑	username	ragu	
☑	password	Ragu@123	

Status: 200 OK Time: 164 ms Size: 363 B

```
{"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6OSwiaWF0IjoxNzE0NTk0NTkwfQ.HZDKAjLEkbsOEWjJx-yTm1pZ78owU5XLkS10jzaC3SE"}
```

The /users endpoint can be accessed only by the admin.



Node tasks / get All Users

GET http://localhost:3000/users

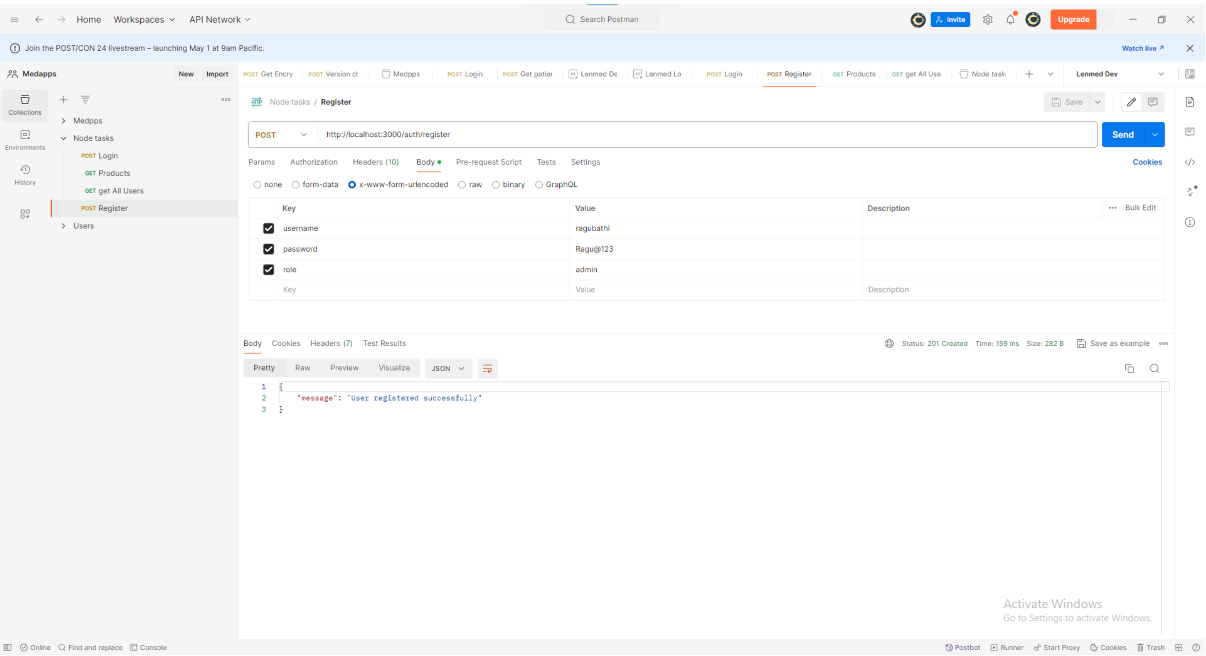
○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key	Value	Description
☑	page	1	
☑	limit	10	
	Key	Value	Description

Status: 200 OK Time: 7 ms Size: 635 B

```
[
    {
        "id": 9,
        "username": "ragu",
        "password": "$2a$10$rKCrRkt/gS7ze9kkdR3MHey.a6lYhWx1ELOyKup/StPcvQ9TI1FV6",
        "role": "customer",
        "createdAt": "2024-05-01T17:24:56.331Z",
        "updatedAt": "2024-05-01T17:24:56.331Z"
    },
    {
        "id": 10,
        "username": "ragubathi",
        "password": "$2a$10$waH4zEMMT7hpP4NRDPKWqOd2XHBNLbatv11tnX./kZIX6BhXW7YOq",
        "role": "admin",
        "createdAt": "2024-05-01T18:13:05.784Z",
        "updatedAt": "2024-05-01T18:13:05.784Z"
    }
]
```

To register new user.



Node tasks / Register

POST http://localhost:3000/auth/register

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key	Value	Description
☑	username	ragubathi	
☑	password	Ragu@123	
☑	role	admin	
	Key	Value	Description

Status: 201 Created Time: 159 ms Size: 282 B

```
{
    "message": "User registered successfully"
}
```