Here is a step-by-step guide on hosting your backend Express.js with PostgreSQL on DigitalOcean with .env config using Nginx:

1) Create a DigitalOcean account and create a new droplet.

2) Connect to your droplet via SSH and update the package index:

```
sudo apt-get update
```

3) Install Node.js and npm:

```
sudo apt-get install nodejs npm
```

4) Install PostgreSQL:

```
sudo apt-get install postgresql postgresql-contrib
```

5) Create a new PostgreSQL database and user:

```
sudo -u postgres psql

CREATE DATABASE mydatabase;

CREATE USER mydatabaseuser WITH PASSWORD 'mypassword';

GRANT ALL PRIVILEGES ON DATABASE mydatabase TO mydatabaseuser;

\q
```

6) Navigate to your Express.js project and create a .env file:

```
cd ~/yourproject

touch .env
```

7) Add your database credentials to the .env file:

```
DB_HOST=localhost

DB_USER=mydatabaseuser

DB_PASSWORD=mypassword

DB_NAME=mydatabase
```

8) Install the dotenv library:

```
npm install dotenv
```

9) Require the dotenv library in your Express.js server file and connect to the database:

```
const dotenv = require('dotenv');
dotenv.config();
javascript
Copy code
const { Client } = require('pg');
const client = new Client({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASSWORD,
  database: process.env.DB_NAME,
});
client.connect();
```

10) Install Nginx:

```
sudo apt-get install nginx
```

11) Create a new Nginx server block configuration file:

```
sudo nano /etc/nginx/sites-available/yourproject
```

12) Add the following configuration to the file:

```
server {
   listen 80; # Listen on port 80
   #listen [::]:80; Listen on port 80 for ipv6(Some error we can’t
   #debug don’t enable ipv6)
   server_name example.com;

   location / {
   proxy_pass http://localhost:3000;
   proxy_http_version 1.1;
   proxy_set_header Upgrade $http_upgrade;
   proxy_set_header Connection 'upgrade';
   proxy_set_header Host $host;
   proxy_cache_bypass $http_upgrade;
  }
}
```

13) Remove the default Nginx site.

```
sudo rm /etc/nginx/sites-enabled/default
```

14) Enable the Nginx server block:

```
sudo ln -s /etc/nginx/sites-available/yourproject /etc/nginx/sites-enabled/
```

15) Test the Nginx configuration:

```
sudo nginx -t
```

16) Restart Nginx:

```
sudo systemctl reload nginx
```

```
sudo systemctl restart nginx
```

17) Start your Express.js server:

```
npm start
```

18) Check that the Node.js application is running on port 3000. You can use the command `"sudo ss -tlnp"` to check which processes are listening on which ports. Look for a process with the name of your Node.js application or the command used to start it.
19) Configure your firewall rules to allow

    HTTP and HTTPS traffic:

```
sudo apt-get install ufw
sudo ufw allow ssh
sudo ufw allow 80
sudo ufw allow 443
sudo ufw enable
```

20) Check the status of the firewall:

```
sudo ufw status
```

21) To set up automatic start of your Express.js server on DigitalOcean using Nginx, you can use a process manager like PM2. Here are the steps:
    a) Install PM2 on your server:
    ```
    sudo npm install pm2 -g
    ```
    b) Start your Express.js server using PM2:
    ```
    cd /path/to/your/express/app
    pm2 start app.js
    pm2 save
    ```
22) Finally, access your Express.js app by visiting your server's IP address or domain name in your web browser.

Your backend Express.js with PostgreSQL should now be successfully hosted on DigitalOcean with Nginx and .env config.

Link to uninstall nginx:
https://www.cyberciti.biz/faq/remove-uninstall-nginx-from-ubuntu-debian-linux-apt-get-command/

Nginx (pronounced "engine-x") is a free, open-source, high-performance web server software widely used to serve dynamic web pages, reverse proxy, and load balancing. Nginx is known for its high performance and stability, making it an ideal choice for serving high-traffic websites. It can handle a large number of simultaneous connections and can be easily configured to serve static content, proxy requests, or handle SSL encryption. In addition to its performance and stability, Nginx is also known for its ease of use, with a simple configuration syntax and a modular design that allows for adding modules as needed. Nginx is used by many of the largest websites in the world and is widely considered one of the best web servers available.