Set up a Node.js Server on a Linux instance and make it accessible via the Internet.

This guide uses Debian 11 Through this guide you will be able to:

- Start a Node.js Server on your Linux machine (or instance).
- Configure nginx as a reverse proxy for it. After this, HTTP requests made to your machine (without specifying a port number) will be forwarded to your Node server.
- Set up a free and automatically renewing SSL certificate for your domain using certbot and Let's Encrypt.

Install NVM:

```
sudo apt install curl curl https://raw.githubusercontent.com/creationix/nvm/master/install.s
| bash source ~/.profile
```

Install Node

```
nvm install 16.10.0
```

or whatever your version is

Clone your Repo from Github

Optional step, you may get your code onto your system using any method you want.

Install GitHub CLI:

```
type -p curl >/dev/null || sudo apt install curl -y
curl -fsSL https://cli.github.com/packages/githubcli-archive-keyring.gpg | sudo dd of=/usr/s
&& sudo chmod go+r /usr/share/keyrings/githubcli-archive-keyring.gpg \
&& echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/githubcli-arch
&& sudo apt update \
&& sudo apt install gh -y
```

```
gh auth login
```

```
git clone YOUR_GITHUB_URL
```

cd into the working directory and run *npm i*.

Copy keys and shit

For files not in your source control, such as .env files and other keys, copy them onto your Linux machine using SCP or any other means.

Start Server locally

```
npm i -g pm2
pm2 server.js
```

replace server.js with whatever file name is the entry point to your application.

Test by running curl localhost:3000.

```
pm2 startup systemd
```

copy and paste the command shown, this will start pm2 on system reboot.

Nginx

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

At etc/nginx/sites-available: create file example.com.

Put contents as:

```
server{
server_name example.com www.example.com

    location / {
        proxy_pass http://127.0.0.1:3000;
        proxy_set_header Host $host;

# proxy_set_header X-Real-IP $remote-addr;

        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    listen 80;

}
```

Save. Delete default file.

```
sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/example.com
```

```
sudo nginx -t
```

The above command checks if the config file has syntax errors or not.

```
sudo sytemctl restart nginx
```

The above command will load the changes into nginx.

Change DNS settings in Domain Registrar Settings to point to the public IP of the Linux instance or machine being set up. This generally consists of creating A and AAAA DNS records.

Once this is done, going to your domain's URL in the browser will show the website created using your Node.js code.
To add HTTPS Support:

SSL with Certbot (Let's Encrypt)

```
sudo apt install certbot python3-certbot-nginx
sudo apt install ufw
sudo ufw status
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow ssh
sudo ufw enable
sudo ufw status verbose
sudo ufw allow 'Nginx Full'
sudo ufw delete allow 'Nginx HTTP'
sudo certbot --nginx -d example.com -d www.example.com
```

Verify auto-renewal:

```
sudo systemctl status certbot.timer
sudo certbot renew --dry-run
```