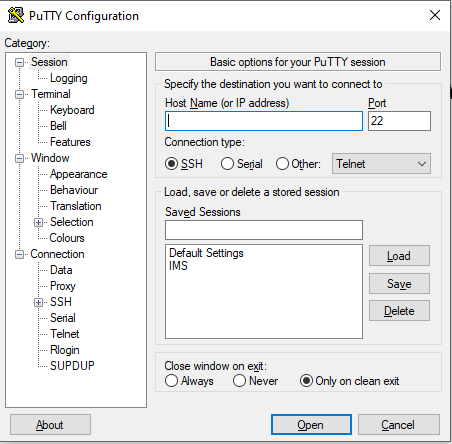
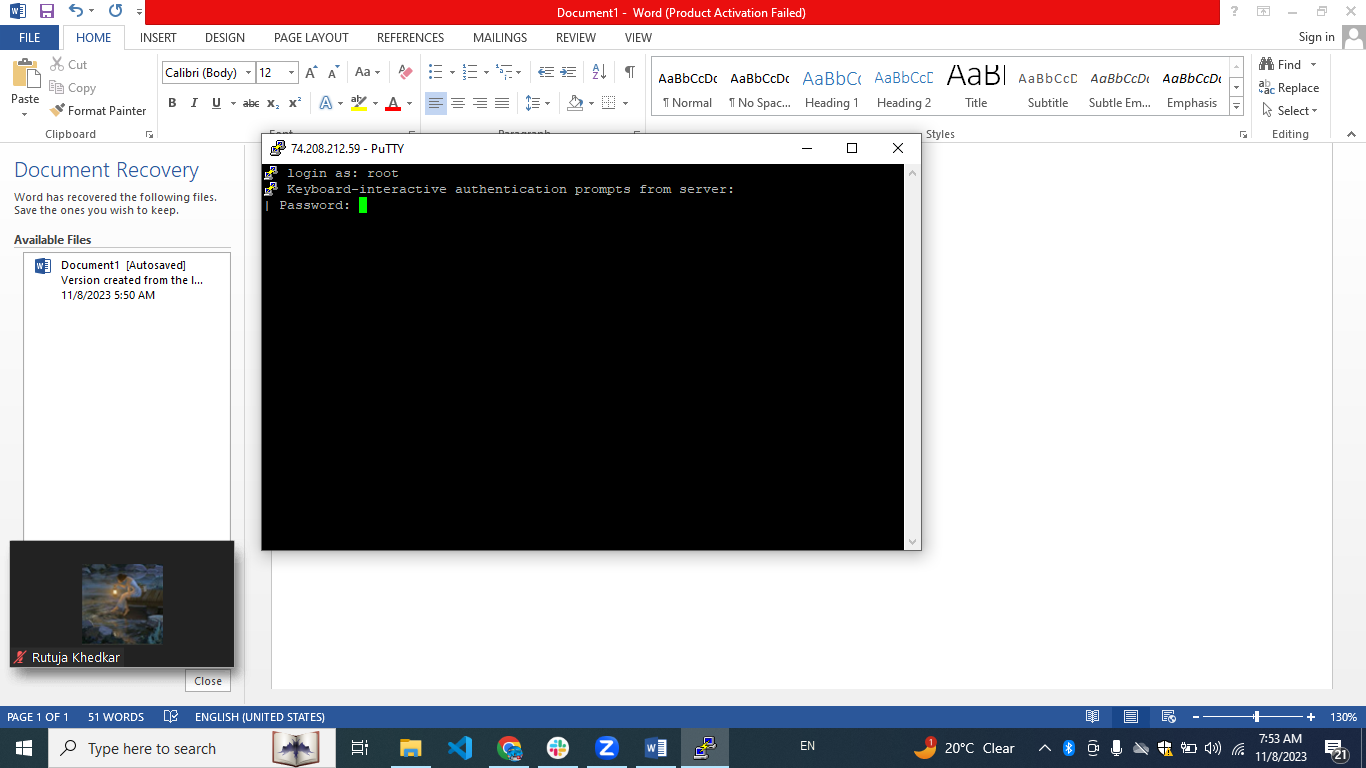
**Steps to deploy MERN Application on VPS Server using Putty**

**Step 1: Connect to Your Server Using PuTTY**

1. Download the putty by using the link:
2. Open the putty
3. You will see screen like this

****

1. Enter the IP address
2. Enter the port no.
3. Click on the Open button
4. You will see the screen like this

****

1. Write the **username** ‘root’. Then enter
2. Write the **password** (when you will enter the password it will not visible on prompt)
3. Write the **ls** command to see all project

**Step 2: Update and Upgrade Packages**

Run the following commands to update the package list and upgrade the installed packages on your server:

sudo apt-get update

sudo apt-get upgrade

```

**Step 3: Install Node.js and MongoDB**

Install Node.js and MongoDB on your server. You can follow the official documentation for your Linux distribution to do this.

**Step 4: Set Up Your MERN Application**

Clone your MERN application repository into your server using Git. Install dependencies and build your React app. Start your Node.js server.

git clone <repository\_url>

cd <your\_project\_folder>

npm install

npm run build

npm start

**Step 5: Install and Configure Nginx (if not)**

Install Nginx on your server:

sudo apt-get install nginx

Create a new Nginx server block configuration for your domain. Replace `example.com` with your actual domain name.

sudo nano /etc/nginx/sites-available/example.com

Add the following configuration (adjust the paths and ports according to your setup):

nginx

server {

listen 80;

server\_name example.com www.example.com;

location / {

proxy\_pass http://localhost:your\_node\_app\_port;

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;

}

location /api {

proxy\_pass http://localhost:your\_node\_api\_port;

# ... (add other proxy settings if needed)

}

# ... (add other server configurations if needed)

}

```

Create a symbolic link to enable the server block:

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

Restart Nginx to apply the changes:

sudo service nginx restart

**Step 6: Configure DNS Settings for Your Domain**

Go to your domain registrar's website and access the DNS settings for your domain. Create an A record pointing to your server's IP address.

**Step 7: Enable SSL/TLS (Optional, but highly recommended)**

Enabling SSL/TLS on your server using Let's Encrypt involves several steps. Here's a detailed guide to help you secure your MERN application with SSL/TLS using Let's Encrypt and Certbot:

**Prerequisites:**

1. A domain name: Make sure your domain is pointing to your server's IP address.

2. Nginx installed and configured: Follow the previous steps to set up your MERN app with Nginx.

3. Certbot: Certbot is a tool to obtain SSL/TLS certificates from Let's Encrypt. Install it on your server using the following command:

bash

sudo apt-get install certbot

```

**Step 1: Obtain SSL/TLS Certificate**

Run Certbot to obtain an SSL/TLS certificate for your domain. Replace `example.com` with your actual domain name.

bash

sudo certbot --nginx -d example.com -d www.example.com

Certbot will automatically configure Nginx to use the obtained certificates. Follow the prompts to provide an email address and agree to the terms of service. Certbot will then communicate with Let's Encrypt to generate and install the SSL/TLS certificates.

**Step 2: Configure SSL/TLS Redirect (Optional, but recommended)**

By default, Certbot will configure your Nginx server block to use SSL/TLS. However, it won't force HTTPS automatically. To redirect all HTTP traffic to HTTPS, edit your Nginx configuration file:

bash

sudo nano /etc/nginx/sites-available/example.com

Add or modify the following lines inside the server block:

nginx

server {

listen 80;

server\_name example.com www.example.com;

return 301 https://$host$request\_uri;

}

server {

listen 443 ssl;

server\_name example.com www.example.com;

**# SSL/TLS configuration**

ssl\_certificate /etc/letsencrypt/live/example.com/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/example.com/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

(other SSL/TLS settings if needed)

location / {

proxy\_pass http://localhost:your\_node\_app\_port;

(other proxy settings if needed)

}

(other server configurations if needed)

}

Save the file and exit the text editor.

**Step 3: Test Nginx Configuration**

Before restarting Nginx, it's a good practice to test the configuration for syntax errors:

sudo nginx -t

If the configuration test passes without any errors, you can proceed to restart Nginx:

sudo service nginx restart

**Step 4: Automatic Renewal (Important)**

Let's Encrypt certificates expire every 90 days. To automatically renew the certificates, Certbot will set up a cron job. You don't need to do anything for this; Certbot takes care of it during the installation process.

Your MERN application is now secured with SSL/TLS encryption, and all HTTP traffic will be automatically redirected to HTTPS.

**Step 8: Test Your Setup**

Open your web browser and navigate to your domain (https://example.com). Your MERN app should now be accessible through Nginx.

That's it! You've successfully set up and deployed your MERN application on Nginx, associated it with your domain name, and secured the connection with SSL/TLS.