Cloud-Based Server Setup Documentation

**Project Overview**

This project involves setting up a secure cloud-based server for a startup, ensuring the server has strong authentication measures, firewall configuration, Fail2Ban protection, and the Apache2 web server installed. The server is configured to allow users to log in using SSH keys, with password-based authentication disabled. The server is also protected by a firewall, allowing only necessary ports.

**Setting Up SSH Key Authentication**

1. Disable Password Authentication:

/etc/ssh/sshd\_config



1. Restart the SSH service to apply changes

**Create a New User and Add SSH Key**

1. Create a User:



1. SSH Key



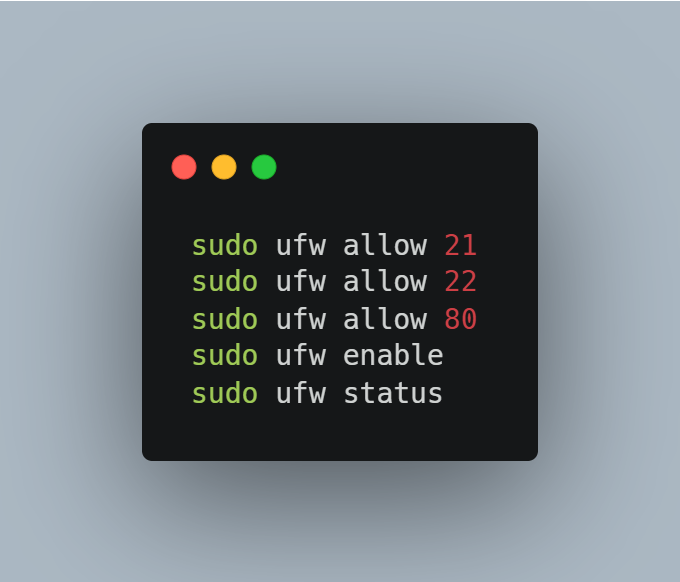
**Private key** (id\_rsa): This is stored on your machine and should be kept secret.

**Public key** (id\_rsa.pub): This is the key you can share with the remote server you want to access.

To access a server using your SSH key, you need to copy the public key (id\_rsa.pub) to the server's ~/.ssh/authorized\_keys file.

**Configuring the Firewall**

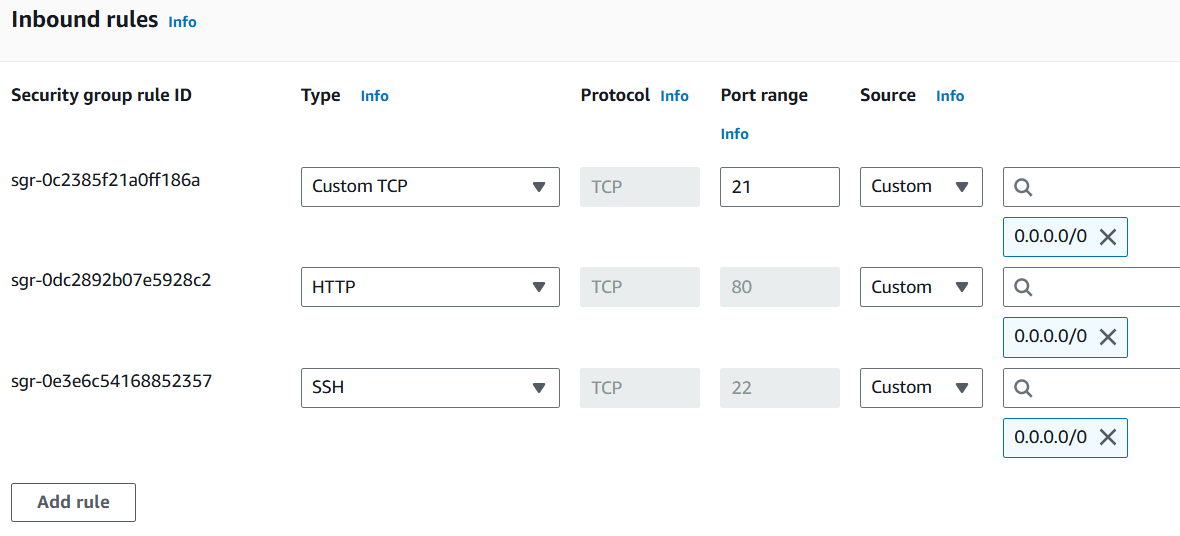
1. Install UFW
2. **Allow Only Necessary Ports**: Allow only ports 21, 22, and 80:



**Configure AWS Firewall Rules**

Set Up Security Group: In the AWS management console, create a security group that only allows:

* Port 21 (FTP)
* Port 22 (SSH)
* Port 80 (HTTP)



**Configuring Fail2Ban**

Fail2Ban is a tool that prevents brute-force login attacks by blocking IPs after a set number of failed login attempts.

* **Install Fail2Ban**

sudo apt install fail2ban

* **Configure SSH Protection: Open the Fail2Ban configuration file**

sudo nano /etc/fail2ban/jail.local

* **Set the Configuration: Add the following configuration:**



* **Restart Fail2Ban: Restart the Fail2Ban service**

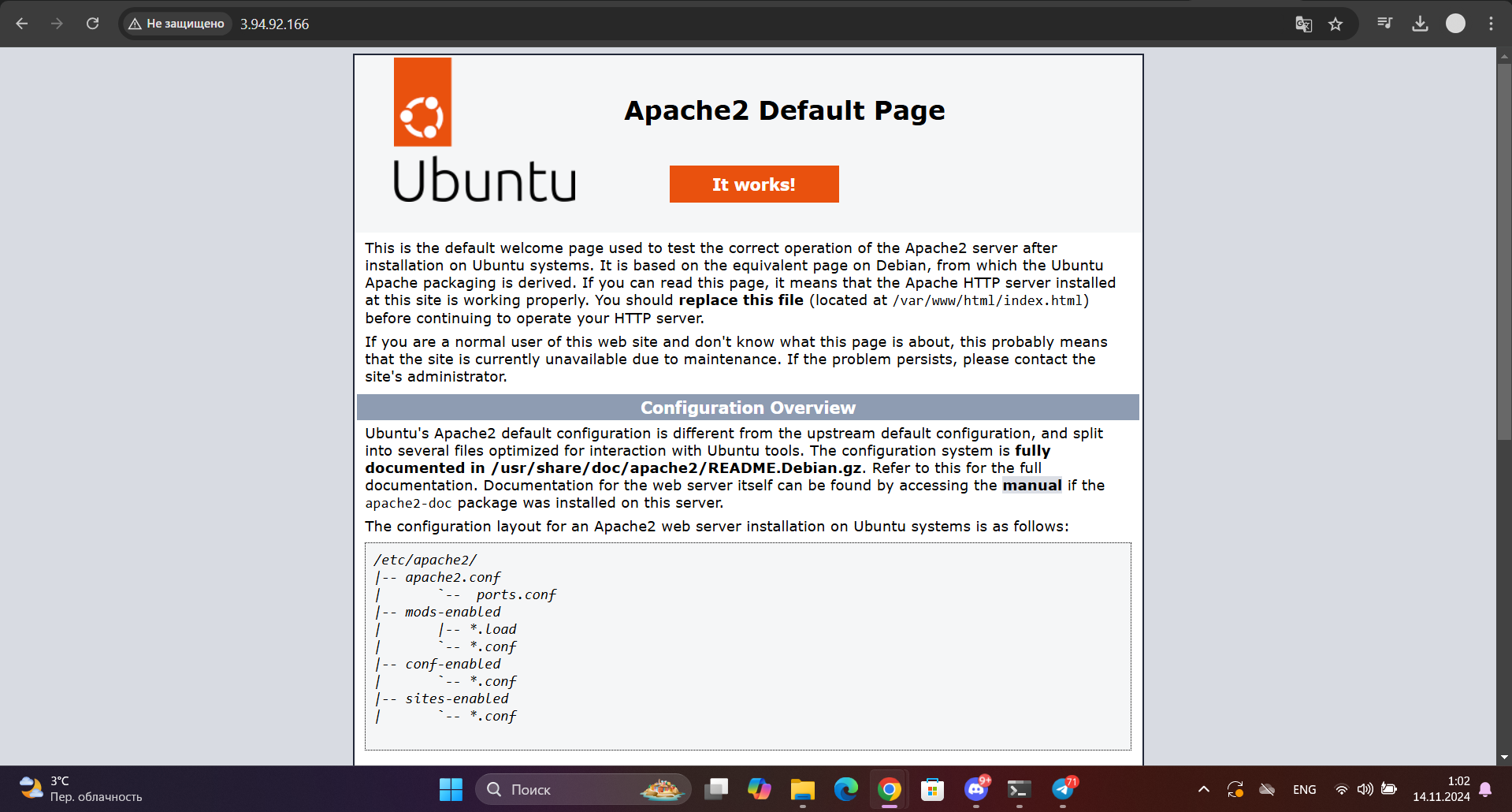
**Installing Apache2 Web Server**

* **Install Apache2**

sudo apt update

sudo apt install apache2

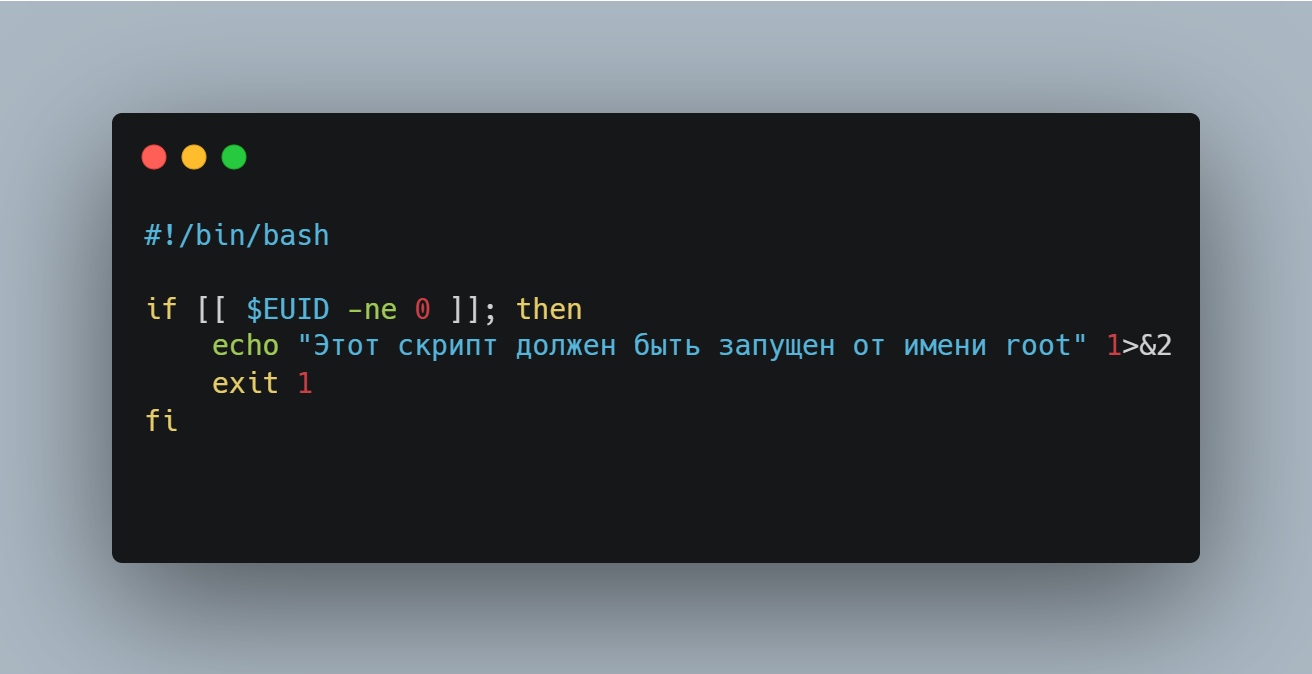
* **Verify Apache2 is Running: Open a browser and navigate to the server’s IP address to confirm Apache2 is serving the default page.**

****

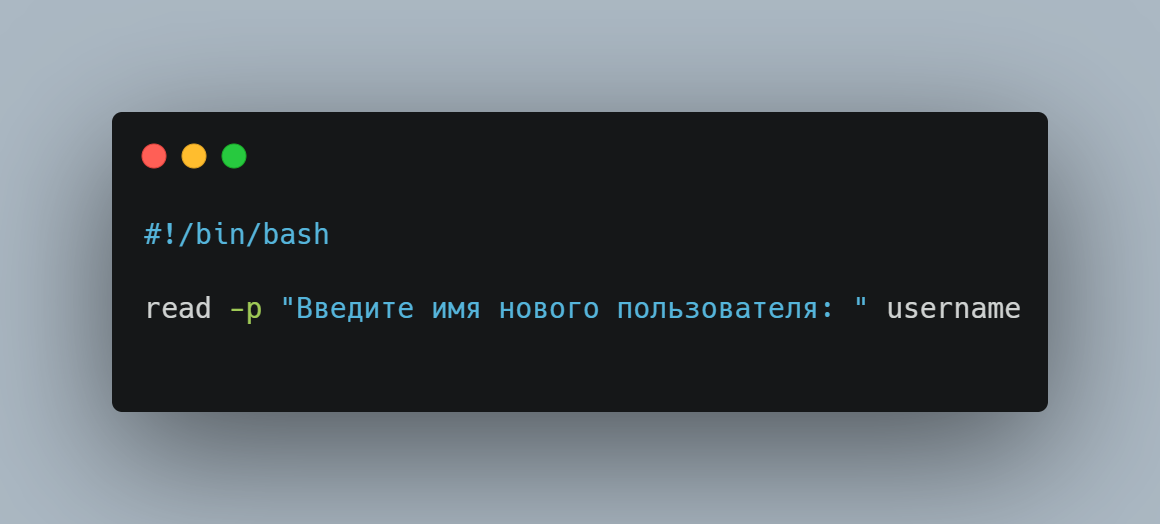
**Automated User Setup Script**

**Bash script that automatically adds a new user, generates an SSH key pair for them, and configures SSH access.**

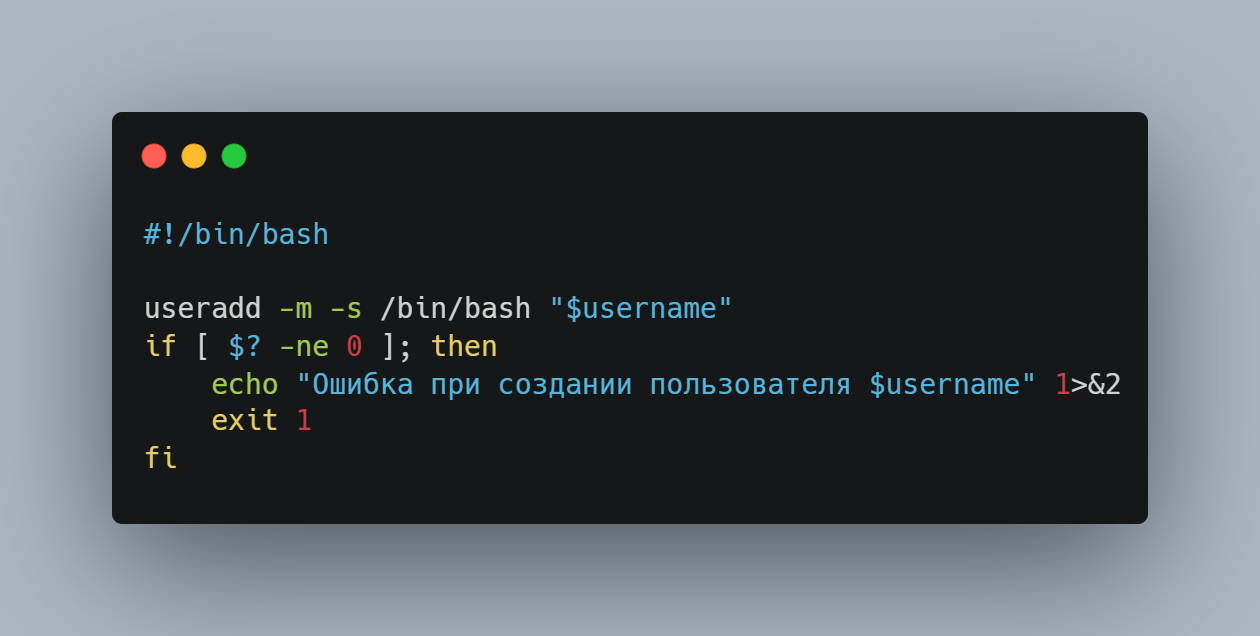
1. **We check if the user has root privileges or if it uses sudo.**

****

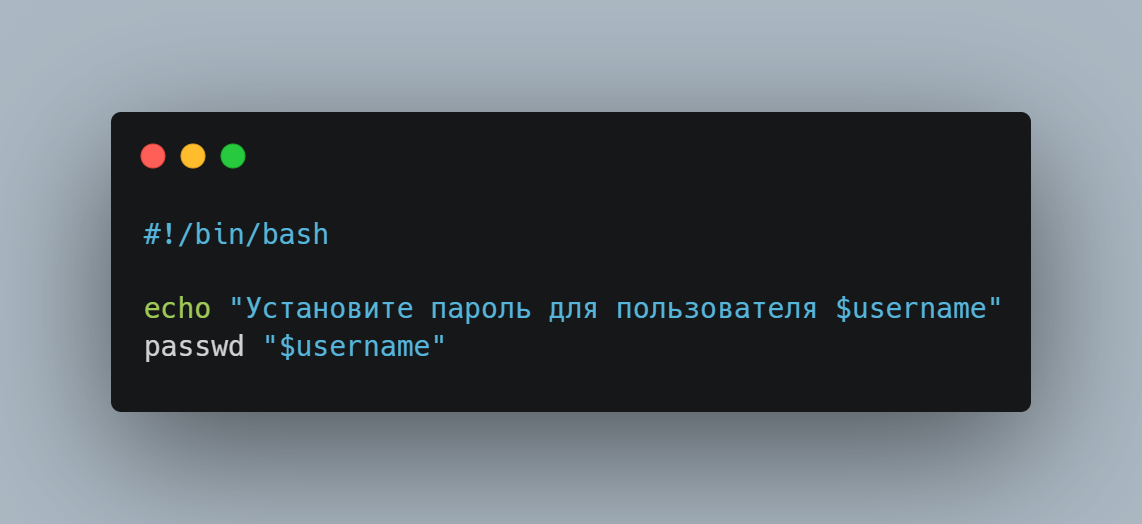
1. **The settings of the new user.**

****

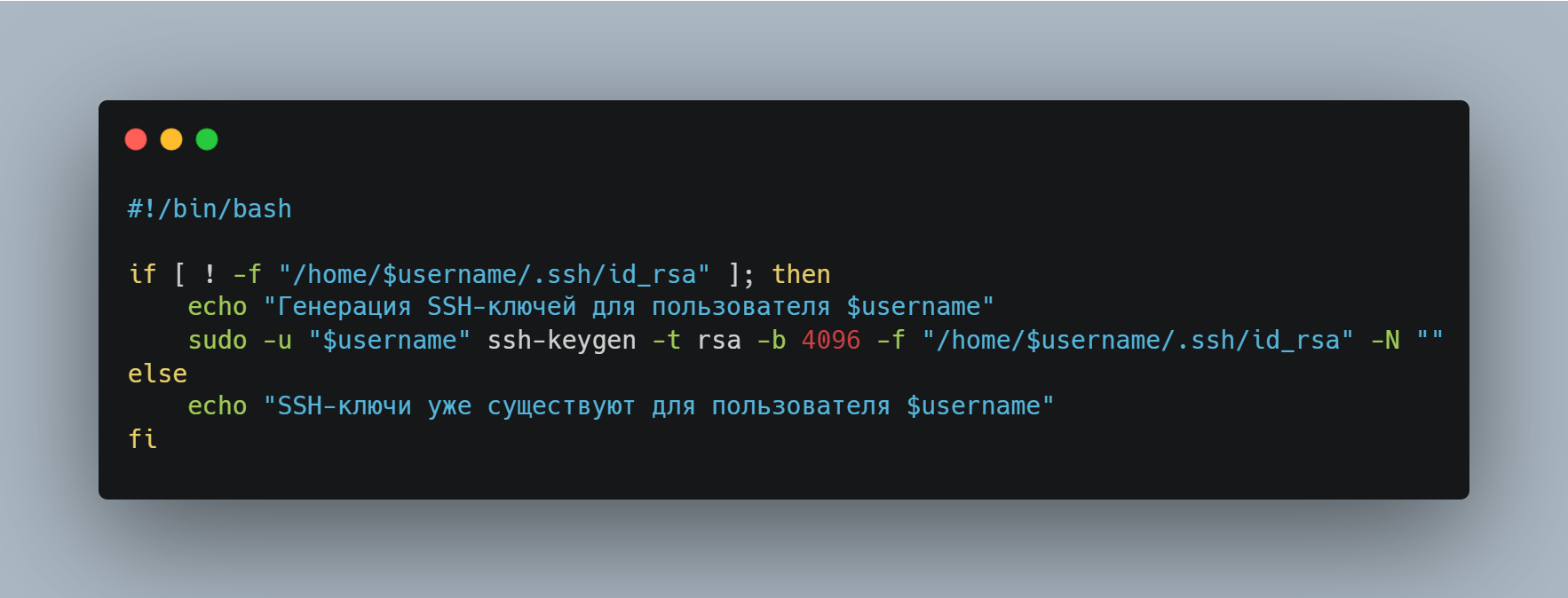
1. **Creation of the new user.**

****

1. **Setting the password.**

****

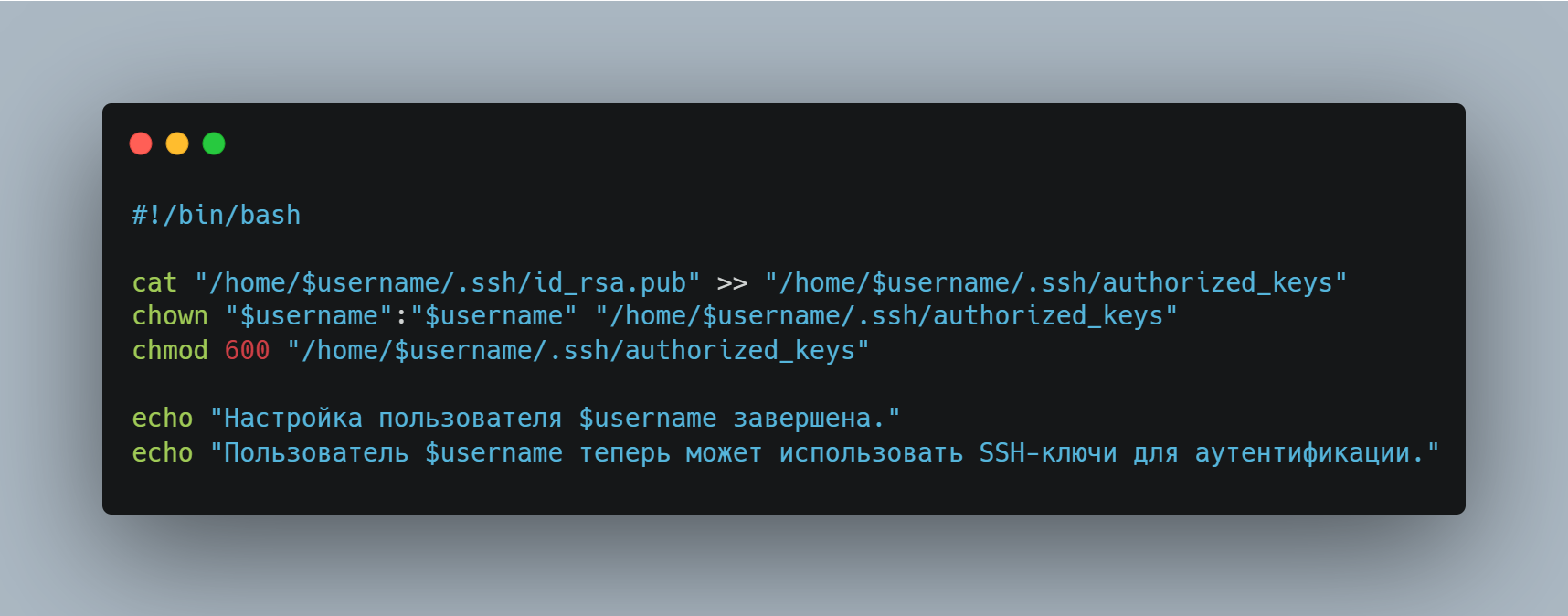
1. **Generating ssh key.**

****

1. **We check if the .ssh folder exists or not.**

****

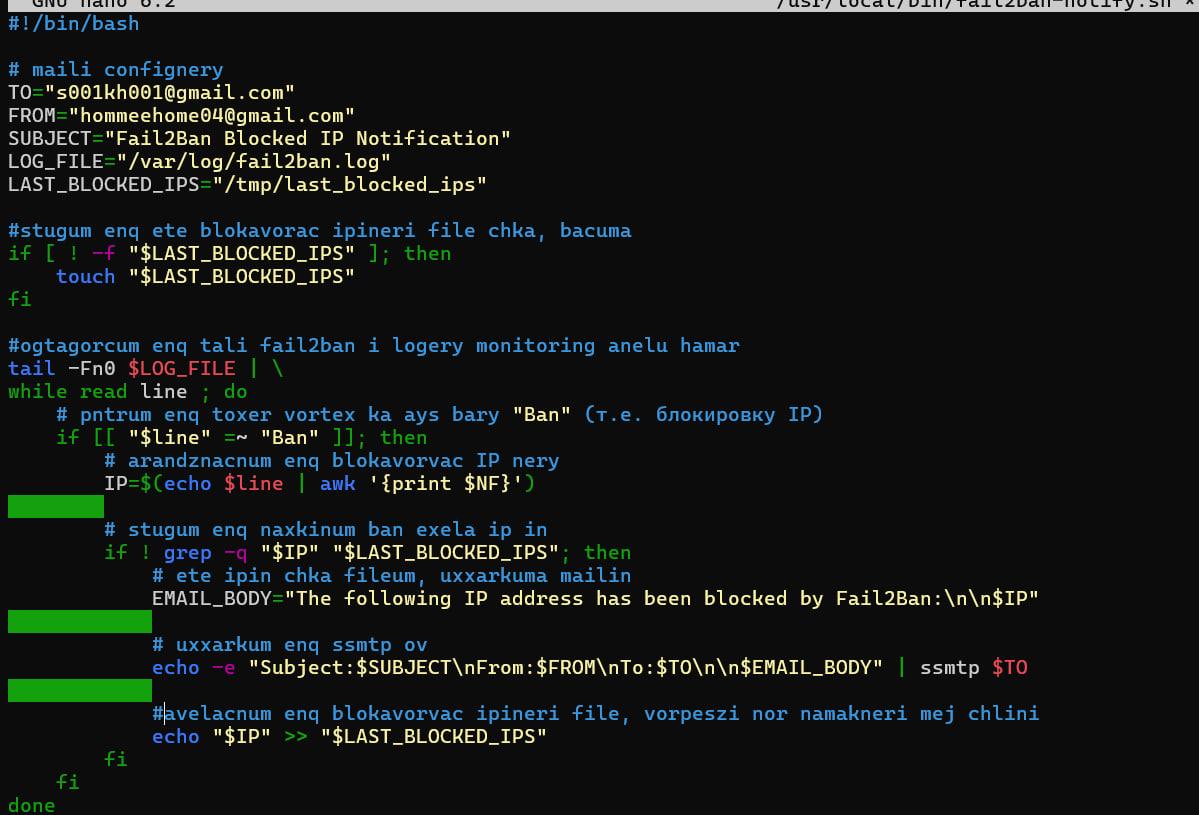
1. **We add the public key into authorized\_keys.**

****

**Security Notification Script**

**Bash script that monitors system logs for suspicious activity and sends an email alert in case of potential breaches or security incidents.**

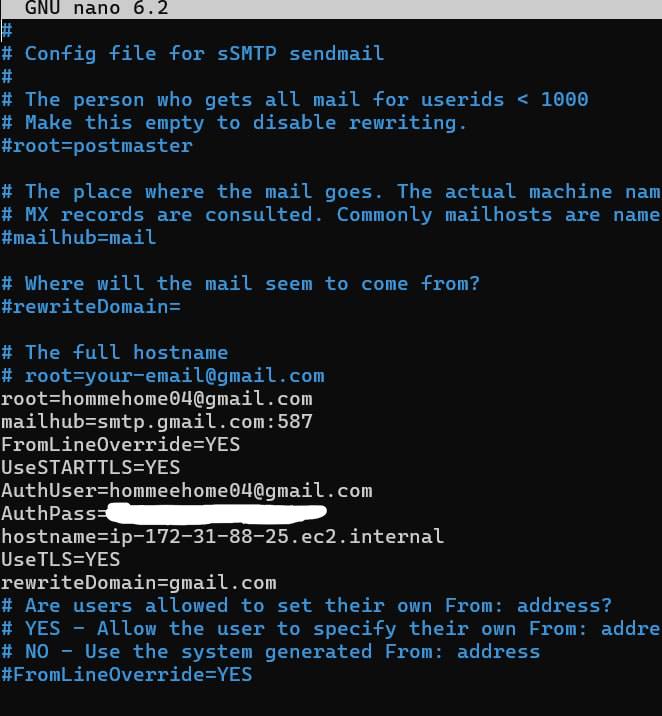
1. **Mail configurations**
2. **We check if there are no blocked IP files, create one**
3. **We use the logs of tali fail2ban for the monitoring**
4. **We look for lines where the word "Ban" is present**
5. **Separate the blocked IP's**
6. **We check if it was banned previously**
7. **We check if the IP is not in the file, we send it to the specified mail**
8. **We send it by ssmtp**
9. **We add a new file for blocked IPs in order to not have them in the list of new messages**

****

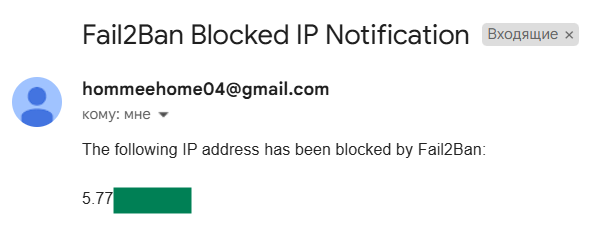
**Install SSMTP service**

* **sudo apt install ssmtp**
* **open this file for configurations for using Gmail**

**sudo nano /etc/ssmtp/ssmtp.conf**

****

**Lastly, we run the script, and after 5 wrong inputs we get an email**

****

**Docker + Apache2 Reverse Proxy**

**Research and implement a solution using Apache2 as a reverse proxy with Docker containers**

**Install Docker**

* sudo apt update
* sudo apt install docker-ce docker-ce-cli containerd.io

**Install Apache2**

* sudo apt install apache2
* sudo systemctl status apache2

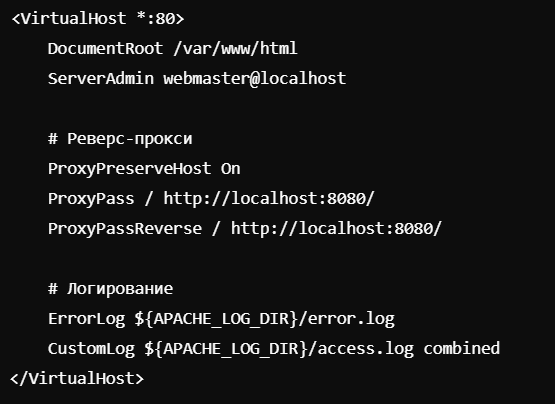
**Install Modules for Reverse Proxy**

* sudo a2enmod proxy
* sudo a2enmod proxy\_http
* sudo a2enmod rewrite
* sudo a2enmod headers

**Open the Docker**

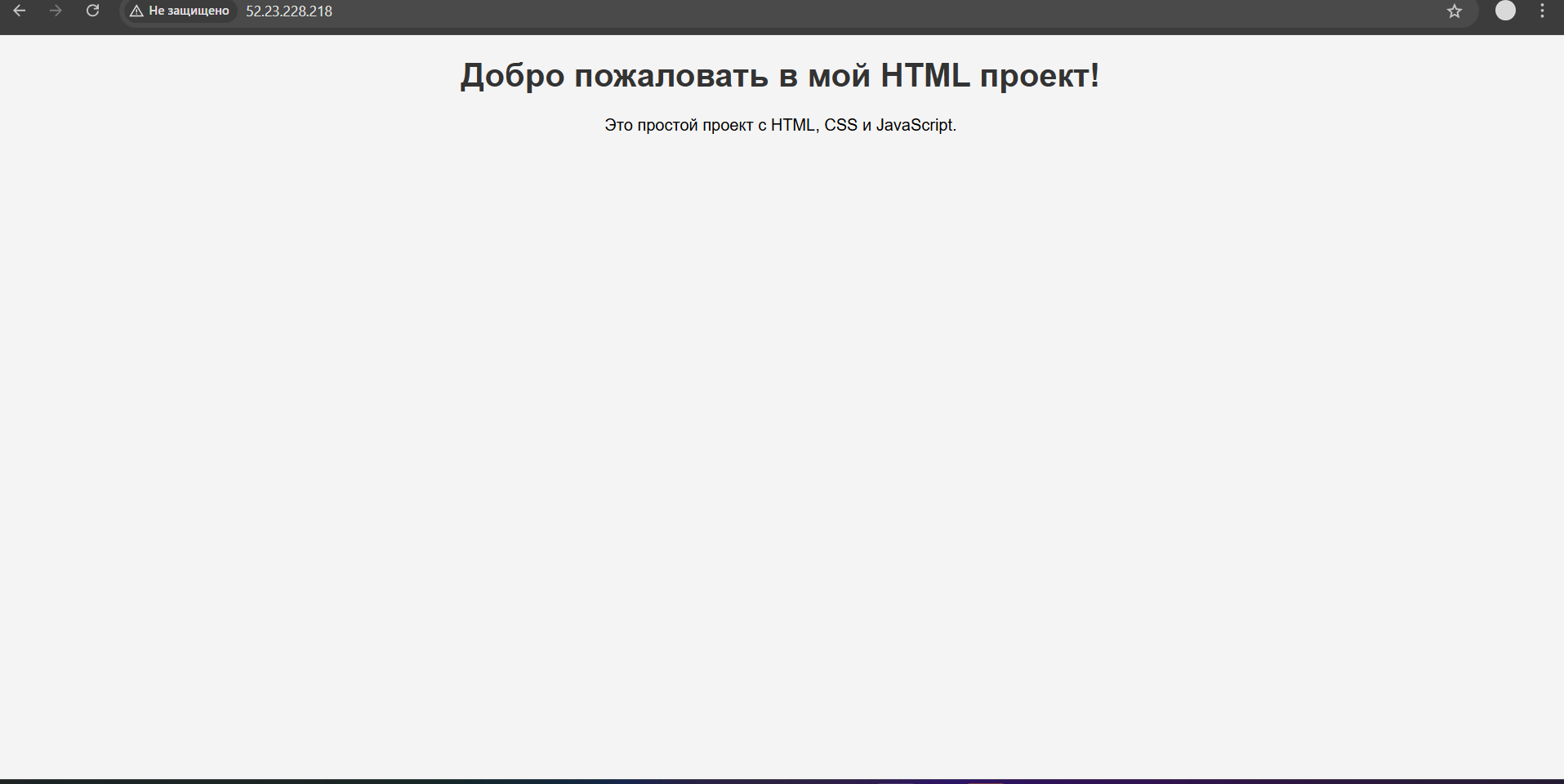
* sudo docker build -t my-html-project
* sudo docker run -d -p 8080:80 my-html-project
* Set the Apache for proxifying the request on your container. Open the configuration file of the virtual host Apache

sudo nano /etc/apache2/sites-available/000-default.conf



* sudo systemctl restart apache2

**Open browser**

****