

To deploy a **Flask application** on **RHEL 9** using **Apache (httpd) as a web server**, follow these steps:

Step 1: Install Required Packages

First, update the system and install necessary packages.

```
dnf update -y
dnf install -y python3 python3-pip httpd mod_ssl
dnf install -y mod_wsgi
```

Step 2: Create a Virtual Environment for Flask

It is best practice to use a virtual environment for Python applications.

```
git clone https://github.com/sunilkumar0633/flask\_app.git
mkdir -p /var/www/flaskapp
cp -rf flask_app/* /var/www/flaskapp
cd /var/www/flaskapp
python3 -m venv venv
source venv/bin/activate
```

Step 3: Install Flask

```
pip install flask
```

Step 4: Create a WSGI Entry Point

Create a WSGI file for the Flask app:

```
vim /var/www/flaskapp/flaskapp.wsgi
```

Add the following content:

```
import sys
import site

# Activate virtual environment
site.addsitedir('/var/www/flaskapp/venv/lib/python3.9/site-packages')
sys.path.insert(0, "/var/www/flaskapp")

from app import app as application
```

Save and exit.

Step 5: Configure Apache for Flask

Create a new Apache configuration file:

```
vim /etc/httpd/conf.d/flaskapp.conf
```

Add the following content:

```
<VirtualHost *:80>
    ServerName www.akaay.online #either use server public ip

    WSGIDaemonProcess flaskapp user=apache group=apache threads=5
    home=/var/www/flaskapp
    WSGIScriptAlias / /var/www/flaskapp/flaskapp.wsgi

    <Directory /var/www/flaskapp>
        Require all granted
    </Directory>

    Alias /static /var/www/flaskapp/static
    <Directory /var/www/flaskapp/static/>
```

```
        Require all granted
    </Directory>

    ErrorLog /var/log/httpd/flaskapp_error.log
    CustomLog /var/log/httpd/flaskapp_access.log combined
</VirtualHost>
```

Step 6: Adjust Permissions and Ownership

Set correct ownership and permissions:

```
chown -R apache:apache /var/www/flaskapp
chmod -R 755 /var/www/flaskapp
```

Step 8: Enable and Restart Apache

```
systemctl enable httpd
systemctl restart httpd
```

Step 9: Open Firewall Ports

Allow HTTP and HTTPS traffic:

```
firewall-cmd --add-service=http --permanent
firewall-cmd --add-service=https --permanent
firewall-cmd --reload
```

OR

If using aws ec2 instance then Add 80/tcp and 443/tcp port in security group.

Instances (1/4) [Info](#) Last updated less than a minute ago [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

Find Instance by attribute or tag (case-sensitive) [All states](#)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	worker node-1	i-03b430621040048e2	Stopped	t2.micro	-	View alarms
<input type="checkbox"/>	worker node-2	i-07b67d67793a0c957	Stopped	t2.micro	-	View alarms
<input checked="" type="checkbox"/>	flask	i-046762afd0eff5111	Running	t2.micro	2/2 checks passed	View alarms

i-046762afd0eff5111 (flask)

[Details](#) [Status and alarms](#) [Monitoring](#) [Security](#) [Networking](#) [Storage](#) [Tags](#)

▼ Security details

[Inbound rules](#) [Outbound rules](#) [Sharing - new](#) [VPC associations - new](#) [Tags](#)

Inbound rules (3) [Manage tags](#) [Edit inbound rules](#)

Search

	Name	Security group rule ID	IP version	Type	Protocol
<input type="checkbox"/>	-	sgr-09e65cf894b1edace	IPv4	SSH	TCP
<input type="checkbox"/>	-	sgr-0e3b39fefdaaa6c8a	IPv4	HTTPS	TCP
<input type="checkbox"/>	-	sgr-07348f68504a634e2	IPv4	HTTP	TCP

Edit inbound rules.

[EC2](#) > [Security Groups](#) > [sg-0509ae6d6eb8db325 - launch-wizard-14](#) > Edit inbound rules

Inbound rules [Info](#)

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-09e65cf894b1edace	SSH	TCP	22	C... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>	Delete
sgr-0e3b39fefdaaa6c8a	HTTPS	TCP	443	C... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>	Delete
sgr-07348f68504a634e2	HTTP	TCP	80	C... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>	Delete

[Add rule](#)

[Cancel](#) [Preview changes](#) [Save rules](#)

Step 10: Test the Deployment

Now, open your browser and visit:

http://your_server_ip/

To secure your **Flask application** running on **Apache** with an **SSL certificate** using **Certbot** on **RHEL 9**, follow these steps:

Step 1: Install Certbot and mod_ssl

First, install **Certbot** and **mod_ssl** (for HTTPS support in Apache).

```
dnf install -y certbot python3-certbot-apache mod_ssl
```

Step 2: Obtain an SSL Certificate

Run the following command to generate and install an SSL certificate for your domain (replace www.akaay.online with your actual domain):

```
certbot --apache -d www.akaay.online ( replace with your domain )
```

- It will **auto-detect your Apache configuration** and request a certificate from **Let's Encrypt**.
- During installation, Certbot will ask:
 - **"Would you like to redirect HTTP to HTTPS?"**
 - Choose **"2: Redirect"** to force all traffic to HTTPS.

Step 3: Verify the SSL Certificate

After installation, verify that your SSL certificate is active:

```
rm -f /etc/httpd/conf.d/ssl.conf  
certbot certificates
```

You should see details about your SSL certificate.

Step 4: Restart Apache

Now restart Apache to apply changes:

```
systemctl restart httpd
```

Map your ip address with domain on domain panel

Go to browser and search

<https://www.domain.com>