**Deploying a Python Django Project on AWS: Step-by-Step Guide with Commands**

[[Saurabh Chodvadiya](https://medium.com/@chodvadiyasaurabh?source=post_page-----84ca8a4f9d6f--------------------------------)](https://medium.com/@chodvadiyasaurabh?source=post_page-----84ca8a4f9d6f--------------------------------)

[Saurabh Chodvadiya](https://medium.com/@chodvadiyasaurabh?source=post_page-----84ca8a4f9d6f--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F4d5f686a0e1e&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40chodvadiyasaurabh%2Fdeploying-a-python-django-project-on-aws-step-by-step-guide-with-commands-84ca8a4f9d6f&user=Saurabh+Chodvadiya&userId=4d5f686a0e1e&source=post_page-4d5f686a0e1e----84ca8a4f9d6f---------------------post_header-----------)

3 min read

·

Jul 16, 2023

3

**Introduction:**

Deploying a Python Django project on Amazon Web Services (AWS) can be an exciting experience that brings your web application to a global audience. In this blog, we’ll guide you through the process of deploying a Django project on AWS using detailed commands. By leveraging AWS services like Amazon EC2, Amazon RDS, and Amazon S3, you can create a scalable and reliable environment for your Django application. Let’s dive into the step-by-step deployment process!

**Step 1: Prepare Your Django Project**

Ensure your Django project is fully functional and tested locally. Create a requirements.txt file containing all the necessary Python packages used in your project.

**Step 2: Set Up an AWS Account**

Sign up for an AWS account at aws.amazon.com and access the AWS Management Console to begin the deployment process.

**Step 3: Launch an Amazon EC2 Instance**

Launch an EC2 instance using the AWS Management Console or run the following command in your terminal:

aws ec2 run-instances --image-id ami-xxxxxxxxxxxxxxxxx --instance-type t2.micro --key-name your-key-pair-name --security-group-ids your-security-group-id --subnet-id your-subnet-id

**Step 4: Connect to Your EC2 Instance**

Using SSH, connect to your EC2 instance:

ssh -i /path/to/your-key.pem ec2-user@your-ec2-public-ip

**Step 5: Install Required Dependencies**

Install Python, pip, and virtualenv on your EC2 instance:

sudo yum update -y  
sudo yum install python3 python3-pip -y  
pip3 install virtualenv

**Step 6: Set Up a Virtual Environment**

Create a virtual environment for your Django project:

mkdir my-django-project  
cd my-django-project  
virtualenv venv  
source venv/bin/activate

**Step 7: Upload Your Django Project Files**

Upload your Django project files to the EC2 instance using SCP or any preferred method.

**Step 8: Install Required Packages**

Install the required Python packages specified in the requirements.txt file:

pip3 install -r requirements.txt

**Step 9: Configure Your Database**

Set up an Amazon RDS instance to host your Django project’s database. Create a database and update settings.py to connect to the RDS instance.

**Step 10: Set Up Amazon S3**

Create an Amazon S3 bucket to store media files (e.g., images, videos). Update settings.py to use the S3 bucket for media storage.

**Step 11: Run Migrations and Collect Static Files**

Run Django migrations to set up the database and collect static files:

python3 manage.py makemigrations  
python3 manage.py migrate  
python3 manage.py collectstatic

**Step 12: Install and Configure a Web Server**

Install and configure Nginx as a reverse proxy server:

sudo amazon-linux-extras install nginx1.12 -y  
sudo systemctl start nginx  
sudo systemctl enable nginx

**Step 13: Deploy Your Django Project**

Run your Django application using Gunicorn:

pip3 install gunicorn  
gunicorn myproject.wsgi:application

**Step 14: Secure Your Deployment**

Enable HTTPS by obtaining and configuring an SSL certificate using ACM or a third-party provider.

**Step 15: Associate a Custom Domain Name (Optional)**

Associate a custom domain name with your Django project using Route 53 or a domain registrar’s settings.

**Conclusion:**

Congratulations! You’ve successfully deployed your Python Django project on AWS using detailed commands. By following this step-by-step guide, you’ve created a scalable and secure environment for your application. AWS services like EC2, RDS, and S3 have enabled you to bring your web application to a global audience. As your project grows, consider exploring additional AWS services to enhance performance and scalability. Happy coding and enjoy the benefits of running your Django project on AWS!