EC2 Django Deployment – Step-by-Step Checklist

This document is a **clean, repeatable deployment guide** based on your proven setup. Use it **next time without re-thinking**.

Step 1: Launch EC2 Instance

1.1 EC2 Configuration

- **AMI**: Ubuntu 22.04 LTS
- **Instance Type**: t2.micro (1GB RAM)
- **Storage**: 30GB gp2

1.2 Security Group

Allow: - SSH (22) → *your IP only* - HTTP (80) - HTTPS (443) - Custom TCP (8000) – testing only

1.3 Connect to Server

```
chmod 400 emart.pem
ssh -i emart.pem ubuntu@<PUBLIC_IP>
```

1.4 Initial System Setup

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y git curl wget build-essential
```

Step 2: Setup Swap Memory (CRITICAL)

```
sudo fallocate -l 2G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
free -h
```

Step 3: Security Basics

3.1 Firewall

```
sudo ufw allow ssh
sudo ufw allow http
sudo ufw allow https
sudo ufw enable
```

3.2 (Optional) Change SSH Port

```
sudo nano /etc/ssh/sshd_config
# Port 2222
sudo systemctl restart ssh
```

Step 4: Install Python 3.12

Option A: Deadsnakes (Recommended)

```
sudo add-apt-repository ppa:deadsnakes/ppa -y
sudo apt update
sudo apt install -y python3.12 python3.12-venv python3.12-dev python3-pip
curl -sS https://bootstrap.pypa.io/get-pip.py | python3.12
python3.12 --version
```

Step 5: Install PostgreSQL 16

```
sudo sh -c 'echo "deb https://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt update
sudo apt install -y postgresql-16 postgresql-contrib-16
sudo systemctl enable postgresql
sudo systemctl start postgresql
```

5.1 Optimize PostgreSQL (Low RAM)

```
sudo nano /etc/postgresql/16/main/postgresql.conf
```

Set:

```
shared_buffers = 128MB
effective_cache_size = 256MB
work_mem = 4MB
maintenance_work_mem = 64MB
```

```
sudo systemctl restart postgresql
```

5.2 Create DB & User

```
sudo -i -u postgres
createuser shornamart --pwprompt
createdb shornamartdb --owner=shornamart
exit
```

Edit `pg_hba.conf`:

```
sudo nano /etc/postgresql/16/main/pg_hba.conf
```

Add:

```
host shornamartdb shornamart 127.0.0.1/32 scram-sha-256
```

Step 6: SSH Key for GitHub

```
ssh-keygen -t ed25519 -C "your-email@example.com"
cat ~/.ssh/id_ed25519.pub
```

Add key to GitHub.

Step 7: Clone Project

```
sudo mkdir -p /opt
git clone --branch release --single-branch git@github.com:cseshahriar/e-mart.git /opt/emart
sudo chown -R ubuntu:ubuntu /opt/emart
cd /opt/emart
```

Step 8: Virtual Environment & Django

```
python3.12 -m venv venv
source venv/bin/activate
pip install --upgrade pip
pip install django gunicorn psycopg2-binary whitenoise[brotli] python-dotenv
python manage.py check
```

Step 9: Gunicorn Service

```
sudo nano /etc/systemd/system/gunicorn.service
```

```
[Unit]
Description=Gunicorn for E-Mart
After=postgresql.service

[Service]
User=ubuntu
Group=www-data
WorkingDirectory=/opt/emart
ExecStart=/opt/emart/venv/bin/gunicorn
  --workers 2
  --bind unix:/opt/emart/emart.sock
  config.wsgi:application
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable gunicorn
sudo systemctl start gunicorn
sudo systemctl status gunicorn
```

Step 10: Manual Gunicorn Test

```
source venv/bin/activate
gunicorn --bind 0.0.0.0:8000 config.wsgi:application
```

Step 11: Install & Configure Nginx

```
sudo apt install -y nginx
sudo rm -f /etc/nginx/sites-enabled/default
sudo nano /etc/nginx/sites-available/emart
```

```
upstream emart_server {
    server unix:/opt/emart/emart.sock;
}

server {
    listen 80;
    server_name _;
    client_max_body_size 25M;

    location /static/ {
        alias /opt/emart/static/;
    }

    location /media/ {
        alias /opt/emart/media/;
    }

    location / {
        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_pass http://emart_server;
    }
}
```

```
sudo ln -s /etc/nginx/sites-available/emart /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl reload nginx
```

Step 12: 502 Error Checklist

```
sudo systemctl status gunicorn
ls -la /opt/emart/emart.sock
sudo journalctl -u gunicorn -n 30
sudo tail -f /var/log/nginx/error.log
```

Fix permissions if needed:

```
sudo chmod 755 /opt/emart
sudo chmod 755 /home/ubuntu
```

DONE ✅

You now have a **production-ready Django deployment on EC2**.

Next upgrade: HTTPS (Let’s Encrypt) + GitHub Actions deploy.