Deploy a project on **AWS EC2** without directly exposing application ports. We’ll achieve this using **Nginx as a reverse proxy** to forward traffic from **port 80 (HTTP)** to your application running on an internal port (like **3000**).

# 🚀 Step 1: Launch an EC2 Instance

1. **Go to AWS EC2 Console** and click **Launch Instance**.
2. Choose an AMI (e.g., **Amazon Linux 2**).
3. Select an instance type (e.g., **t2.micro** for the free tier).

## Configure Instance Details:

### Network: Default VPC

* + Subnet: Select a public subnet
  + Auto-assign Public IP: **Enable**

1. **Add Storage:** Keep the default (8 GiB).

## Configure Security Group:

* + Allow **HTTP (port 80)** and **HTTPS (port 443)** for public access.
  + Allow **SSH (port 22)** for remote login.

1. **Launch the instance** and download the **Key Pair (.pem)** file.

# ☎ Step 2: Connect to Your EC2 Instance

### SSH into the instance using your key pair:

ssh -i "mykey.pem" ec2-user@<EC2-Public-IP>

# 🛠️ Step 3: Update and Install Dependencies

### Update packages and install **Nginx** and **Node.js** (or any other backend framework):

sudo yum update -y

sudo amazon-linux-extras install nginx1 -y sudo yum install git -y

curl -sL https://rpm.nodesource.com/setup\_18.x | sudo bash - sudo yum install nodejs -y

# 📂 Step 4: Clone Your Application

### Clone your project from GitHub (or transfer files manually):

git clone https://github.com/username/myapp.git cd myapp

npm install # Or pip install -r requirements.txt for Python apps

# 🚀 Step 5: Run Your Application Locally

### Start your app on an internal port (e.g., 3000):

node app.js # or npm start

### Verify it works locally by visiting:

curl [http://localhost:3000](http://localhost:3000/)

# 🌐 Step 6: Configure Nginx as a Reverse Proxy

### Edit the default Nginx configuration file:

sudo nano /etc/nginx/nginx.conf

Replace the server block with:

server {

listen 80; server\_name \_;

location / {

proxy\_pass [http://localhost:3000;](http://localhost:3000/) 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;

}

}

### Test and restart Nginx:

sudo nginx -t

sudo systemctl restart nginx

# 🔧 Step 7: Configure Firewall and Security Groups

1. **Open HTTP and HTTPS Ports (80, 443) in the Security Group** of your EC2 instance.
2. Check your **EC2 instance’s public IP** or **Elastic IP**:
3. curl http://<EC2-Public-IP>

You should see your application’s response.

# 🔁 Step 8: Configure Your Application to Run in the Background

### Use **PM2** to keep the app running:

sudo npm install -g pm2 pm2 start app.js

pm2 startup pm2 save

# A Step 9: (Optional) Set Up HTTPS with SSL/TLS

### To enable HTTPS, use **Certbot** for SSL certificates:

1. Install Certbot:

sudo yum install certbot python3-certbot-nginx -y

### Obtain a certificate:

sudo certbot --nginx -d yourdomain.com -d [www.yourdomain.com](http://www.yourdomain.com/)

### Automatic renewal:

sudo crontab -e

### Add the following line:

0 0 \* \* \* /usr/bin/certbot renew --quiet

# 🌐 Step 10: Access Your Application

### Visit your application via the public IP or domain:

http://<EC2-Public-IP> https://yourdomain.com

# 📝 Step 11: Monitor Your Application

Use **PM2** and **Nginx logs** to monitor:

pm2 logs

sudo tail -f /var/log/nginx/access.log sudo tail -f /var/log/nginx/error.log

# ✅ Step 12: Troubleshooting Tips

* If you face errors, check the following:

## Nginx configuration:

* + sudo nginx -t

## Service status:

* + sudo systemctl status nginx
  + pm2 status
  + **Security Group settings:** Ensure HTTP (80) and HTTPS (443) are open.