

OffBeatEscape

Deployment Strategy



University
of Windsor



Table of Contents

OVERVIEW OF THE DEPLOYMENT STRATEGY	4
STEPS FOR DEPLOYMENT.....	4
• CREATE A NEW UBUNTU SERVER ON AWS EC2	4
• CONNECT TO UBUNTU EC2 INSTANCE VIA SSH	4
• SETUP WEB SERVER WITH NODE.JS, MONGODB + NGINX	4
• DEPLOY NODE.JS AND MONGODB BACK-END API	4
• BUILD AND DEPLOY THE ANGULAR FRONT-END APP	4
• CONFIGURE NGINX TO SERVE THE NODE.JS API AND ANGULAR FRONT-END	4
• TEST YOUR APPLICATION USING PUBLIC URL	4
PROCEDURE FOR DEPLOYMENT	5
• CREATE A NEW UBUNTU SERVER ON AWS EC2	5
• CONNECT TO UBUNTU EC2 INSTANCE VIA SSH	5
• SETUP WEB SERVER WITH NODE.JS, MONGODB AND NGINX.....	5
• DEPLOY NODE.JS AND MONGODB BACK-END API.....	6
• BUILD AND DEPLOY THE ANGULAR FRONT-END APP.....	6
• CONFIGURE NGINX TO SERVE THE NODE.JS API AND ANGULAR FRONT-END	6
1. DELETE THE DEFAULT NGINX SITE CONFIG FILE WITH THE COMMAND: SUDO RM /ETC/NGINX/SITES-AVAILABLE/DEFAULT. 6	
5. RESTART NGINX SERVER.	7
6. THIS MARKS THE END OF DEPLOYMENT.	7
• TEST YOUR APPLICATION USING PUBLIC URL.....	7



Objective

This document serves as a deployment strategy for the Offbeat Escape Application.

Author – Sidhartha Verma

Recipient - Dr. Aznam Yacoub

Written On – 13-04-2021



Overview of the deployment strategy

The application has been deployed using Amazon's AWS platform. A server running linux operating system is hosted in AWS. All necessary libraries and applications are installed on this system. The application is deployed and the hosted server is then accessed by a public URL.

Steps for deployment

- Create a new Ubuntu Server on AWS EC2
- Connect to Ubuntu EC2 Instance via SSH
- Setup Web Server with Node.js, MongoDB + NGINX
- Deploy Node.js and MongoDB Back-end API
- Build and Deploy the Angular Front-end app
- Configure NGINX to serve the Node.js API and Angular front-end
- Test your application using public URL.



Procedure for deployment

- **Create a new Ubuntu Server on AWS EC2**

1. Sign into the AWS Management Console at <https://aws.amazon.com/console/>.
2. Go to the EC2 Service section.
3. Click the "Launch Instance" button.
4. **Choose AMI** - Check the "Free tier only" checkbox, enter "Ubuntu" in search box and press enter, then select the "Ubuntu Server 18.04" Amazon Machine Image (AMI).
5. **Choose Instance Type** - Select the "t2.micro" (Free tier eligible) instance type and click "Configure Security Group" in the top menu.
6. **Configure Security Group** - Add a new rule to allow HTTP traffic then click "Review and Launch".
7. **Review** - Click Launch
8. Select "Create a new key pair", enter a name for the key pair (e.g. "my-aws-key") and click "Download Key Pair" to download the private key, you will use this to connect to the server via SSH.
9. Click "Launch Instances", then scroll to the bottom of the page and click "View Instances" to see details of the new Ubuntu EC2 instance that is launching.

- **Connect to Ubuntu EC2 Instance via SSH**

1. Connect to the EC2 instance using via ssh.
2. Use your private key if you are using terminal or other ssh clients to connect to the EC2 instance.
3. Once connected proceed with the next step.

- **Setup Web Server with Node.js, MongoDB and NGINX**

1. Install node js on the server using the command
2. Install npm on the server using the appropriate command
3. Install pm2 library.
4. Install mongoDB database.
5. Install the NGINX web-server.



- **Deploy Node.js and MongoDB Back-end API**

1. Clone the application from: <https://github.com/verma81/OffBeatEscape.git>.
2. Start the pm2 server using `sudo pm2 start server.js`
3. The above command runs node everytime on the server machine once the EC2 instance is started.
4. The API is now running on Node.js under the PM2 process manager and listening on port 4000. Only port 80 (HTTP) is publicly accessible on the server so we can't hit the API yet, this will be possible after we've configured NGINX as a reverse proxy to pass through HTTP traffic to the api.

- **Build and Deploy the Angular Front-end app**

1. Clone the application from: <https://github.com/verma81/OffBeatEscape.git>.
2. Move inside "client" folder. This is the front-end of the application.
3. Run the command `ng build --prod` inside the "client" folder.
4. This will generate the production build for the front-end application.
5. Copy the items of the "dist" folder.
6. Connect to the EC2 instance using FTP client. For example "Cyberduck" for MAC and "WinSCP" for Windows.
7. Create a folder in the ubuntu machine and paste the contents of the "dist" folder inside it.
8. This deploys the front-end of the application.

- **Configure NGINX to serve the Node.js API and Angular front-end**

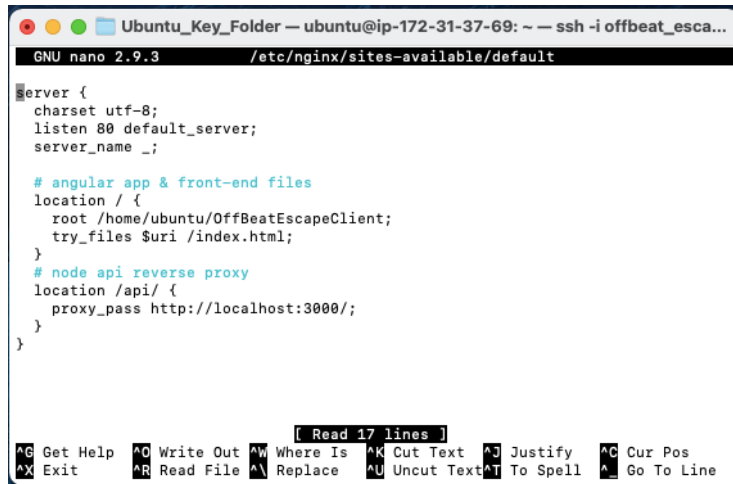
1. Delete the default NGINX site config file with the command: `sudo rm /etc/nginx/sites-available/default`.
2. Launch the nano text editor to create an new default site config file with `sudo nano /etc/nginx/sites-available/default`.
3. Configure the file with the following settings.



Offbeat Escape Deployment Strategy

Sidhartha Verma

Version 1.0



```
GNU nano 2.9.3 /etc/nginx/sites-available/default

server {
    charset utf-8;
    listen 80 default_server;
    server_name _;

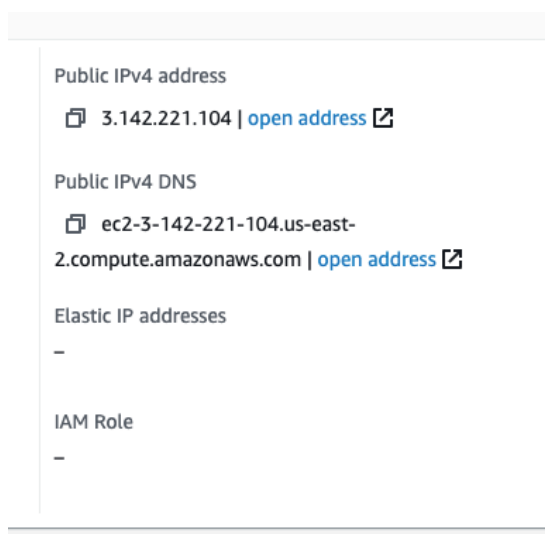
    # angular app & front-end files
    location / {
        root /home/ubuntu/OffBeatEscapeClient;
        try_files $uri /index.html;
    }

    # node api reverse proxy
    location /api/ {
        proxy_pass http://localhost:3000/;
    }
}
```

4. Save the file.
5. Restart NGINX server.
6. This marks the end of deployment.

- **Test your application using public URL.**

1. Open your AWS console. See the public IP of your EC2 instance.



2. Open up the IP and see that your application is running.