# Description:

Build and deploy an angular app on openshift using docker strategy and official nginx image.

***Note****: Remember to update the “****api\_base\_url****” key in* ***environment.prod.ts*** *file same as the* ***backend api route url*** *created , so that the front end can communicate with the backend api, else nothing will work.*

*Public repo url for code base for reference : https://github.com/escortnotice/pizza-order-delivery-ui*

# Create a new app in the current project with the below cli.

*Currently the project is uploaded in ibm git hub hence need to use the token for authentication else for public repos it’s not needed.*

## Method 1:

CLI Command: > **oc new-app https://shuroych:<token>@github.ibm.com/shuroych/pizza-order-delivery-ui.git --strategy=docker**

Ensure that the “**Dockerfile**” file is present in the project folder for docker strategy build to be used, else it will throw error.

Then run the below openshift config files:

1. Route.yaml

Other services are created on openshift cluster with the above command which are (image stream, build config, deployment config, service).

\*\* It is recommended to not use the above method to create the app, if it is uploaded to a private repo. We need to pass the token in the git url which gets stored in the build-config file. This is a breach of security as the password or token should be stored as secrets and should not get directly exposed to any file.

For public repos it is fine to use Method 1 as we don’t need to provide password or token

## Method 2:

Create a new app by running the openshift configuration file in the below order:

1. github-secret.yaml : This is needed if you are pulling the source code from a private repository. In our case we are pulling from “ibm.github” which is a private repository. Not needed for public repository.
2. Image-stream.yaml
3. Build-config.yaml
4. Service.yaml
5. Route.yaml
6. Deployment-config.yaml

# **Dockerfile** entries to be used.

|  |
| --- |
| FROM nginx:1.17.1-alpine  #install node and npm  RUN apk add --update nodejs nodejs-npm  # to fix the issue on openshift build where it was not able to install angular cli.  #this only occurred on the aarch64 variant of the image. The only thing different is the architecture.  # ref url: https://github.com/oznu/docker-homebridge/issues/93  RUN npm config set unsafe-perm true  #install angular cli  RUN npm install -g @angular/cli  #install bash  RUN apk update  RUN apk upgrade  RUN apk add bash  # make this the working directory, hereon all commands will be run here by default  WORKDIR /usr/src/app  # Copy all the contents of current folder(the project files) to /usr/src/app folder  COPY . .  #to read package.json and install dependencies  RUN npm install --silent  # build the app using environment.prod.ts file entries  RUN ng build --prod  #copy the build from dist folder in to nginx folder to get served.  RUN cp -r dist/pizza-order-delivery-ui/. /usr/share/nginx/html  ##The official NGINX docker container published on docker hub does not run on Openshift,  ## because of OpenShift security constraints. Hence it needs the below entries in docker file to run on openshift  ## also due to security restrictions, it cannot bind on port 80, so use some other port (like:8081 here)  # support running as arbitrary user which belogs to the root group  RUN chmod g+rwx /var/cache/nginx /var/run /var/log/nginx  # comment user directive as master process is run as user in OpenShift anyhow  RUN sed -i.bak 's/^user/#user/' /etc/nginx/nginx.conf  # override the default port 80 on nginx and make the app available on port 8081  RUN sed -i.bak 's/listen\(.\*\)80;/listen 8081;/' /etc/nginx/conf.d/default.conf  EXPOSE 8081 |

# Create a secret to access and pull the source code from a private registry using token.

## Method 1 : From UI

Secrets -> Source Secret

Secret Name: github-secret

Authentication Type : Basic

Username : shuroych

Password : <Github generated Token or Password >

## Method 2: By running a configuration file (yaml)

|  |
| --- |
| kind: Secret  apiVersion: v1  type: kubernetes.io/basic-auth  metadata:  name: github-secret  namespace: oc-in-action  stringData:  password: "<Github generated password or token>"  username: "shuroych" |

# Create an image-stream.

Without creating this image stream, you will not be able to trigger a successful build.

|  |
| --- |
| kind: ImageStream  apiVersion: image.openshift.io/v1  metadata:  name: pizza-order-delivery-ui |

# Create a “Build Config” for triggering builds in openshift using Docker strategy and pulling code from ibm github.

(you might need to remove the secret section in case of using public repo)

Click on “Add” button and click on “import yaml” and paste the below config file.

Once Buildconfig is created, click on the build config, then Actions-> start build

Once build is finished successfully a new image is created with latest tag and you can find it in “image streams” section.

|  |
| --- |
| apiVersion: build.openshift.io/v1  kind: BuildConfig  metadata:  name: pizza-order-delivery-ui  spec:  output:  to:  kind: ImageStreamTag  name: "pizza-order-delivery-ui:latest"  runPolicy: Serial  source:  git:  ref: master  uri: "https://github.ibm.com/shuroych/pizza-order-delivery-ui.git"  sourceSecret:  name: github-secret  type: Git  strategy:  dockerStrategy: ~  type: Docker  triggers:  -  github:  secret: OSn21\_\_9zXSDzhd82-T9  type: GitHub  -  generic:  secret: kA2ETpXZ1AWmBm65BAQs  type: Generic  -  type: ConfigChange  -  imageChange:  lastTriggeredImageID: "nginx@sha256:20b62c392073deac500292d6b37c851bb4d00986edb3d73d08c0f0e65019ce6c"  type: ImageChange  status:  lastVersion: 1 |

# Create a service (ClusterIP) service to proxy requests to the pods.

|  |
| --- |
| kind: Service  apiVersion: v1  metadata:  name: pizza-order-delivery-ui  spec:  ports:  - name: http  protocol: TCP  port: 8081  targetPort: 8081  selector:  app: pizza-order-delivery-ui  type: ClusterIP |

# Create a route to serve user requests.

You can use the below configuration or also can create it from UI.

|  |
| --- |
| apiVersion: route.openshift.io/v1  kind: Route  metadata:  name: pizza-order-delivery-ui  spec:  port:  targetPort: 8081  to:  kind: Service  name: pizza-order-delivery-ui |

Or use cli command:

oc expose service <service name> -n <project name>

oc expose svc/<servicename>

Example: > oc expose service pizza-order-delivery-ui -n pizzadeliveryproject

* oc expose svc/pizza-order-delivery-ui

# Now create a deployment config for deployment. Use the below yaml configuration.

Click on Add -> import yaml

Once created, click on the deployment config and then actions -> start rollout.

|  |
| --- |
| kind: DeploymentConfig  apiVersion: apps.openshift.io/v1  metadata:  name: pizza-order-delivery-ui  spec:  strategy:  type: Rolling  triggers:  - type: ConfigChange  - type: ImageChange  imageChangeParams:  automatic: true  containerNames:  - pizza-order-delivery-ui  from:  kind: ImageStreamTag  name: 'pizza-order-delivery-ui:latest'  replicas: 1  selector:  app: pizza-order-delivery-ui  deploymentconfig: pizza-order-delivery-ui  template:  metadata:  labels:  app: pizza-order-delivery-ui  deploymentconfig: pizza-order-delivery-ui  spec:  containers:  - name: pizza-order-delivery-ui  image: 'pizza-order-delivery-ui:latest'  ports:  - containerPort: 8081  protocol: TCP |

# To check logs for troubleshooting

* oc logs -f <podname>