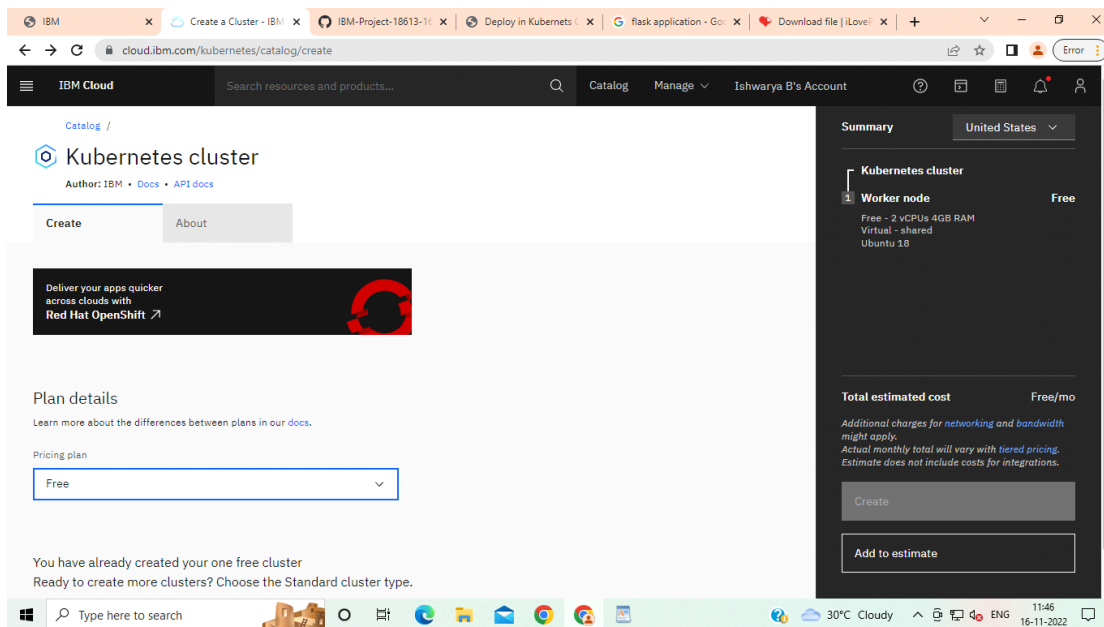
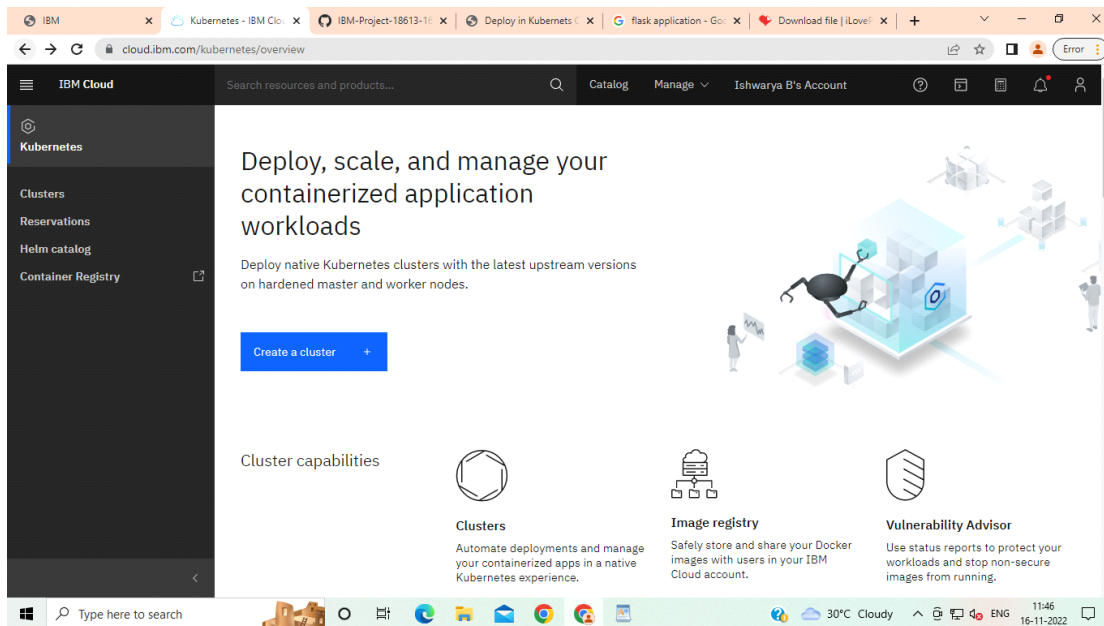


Upload Image To IBM Container Registry

Team ID	PNT2022TMID37215
Project Name	Project - SMART FASHION RECOMMENDER APPLICATION

Creation of kubernetes cluster:



The image displays two screenshots of the IBM Cloud Kubernetes dashboard. The top screenshot shows the 'Overview' page for a cluster named 'mycluster-free'. It displays status cards for Node status (1 of 1 Pending), Add-on status (Error), Master status (Unknown), and Ingress status (Pending). Below these are details for the cluster ID, version (1.24.7_1542), infrastructure (Classic), zones (Milan 01), creation time (11/16/2022, 11:33 AM), resource group, and image security enforcement (Enable button). A node health section shows 1 total nodes with a 100% pending status. The bottom screenshot shows the 'Worker nodes' page for the same cluster. It features a table with columns: Name, Status, Worker pool, Zone, Private IP, Public IP, and Version. One node is listed with ID 00000086, status 'Provisioning - Infrastructure operation...', worker pool 'default', zone 'Milan 01', private IP 10.144.182.12, public IP 159.122.179.200, and version 1.24.7_1543. The page also includes filters, search, and pagination controls.

Dockerfile:

```
FROM python:2.7
LABEL maintainer="AishwaryaB, 310119205009@smartinternz.com"
RUN apt-get update
RUN mkdir /app
WORKDIR /app
COPY . /app
RUN pip install -r requirements.txt
EXPOSE 5000
```

```
ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

Build an image from the Dockerfile:

```
Sending build context to Docker daemon 348.2kB
Step 1/8 : FROM python:2.7
--> 6c78e3b67cfe
Step 2/8 : LABEL maintainer="Kunal Mishra, kunal.mishra@ibm.com"
--> Using cache
--> a8b574d1201c
Step 3/8 : RUN apt-get update
--> Using cache
--> 630d334e9e
Step 4/8 : COPY ./app
--> #0772770094
Step 5/8 : WORKDIR /app
Removing intermediate container f9208b59d2fe
--> 86c0a70b6c
Step 6/8 : RUN pip install -r requirements.txt
--> Running in 8153a4d0d07
Collecting click==6.7 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/34/c3/8b6f99713dd85c38ec95b7f9807132ef8d750f1fab070075077/click-6.7-py2.py3-none-any.whl (71kB)
Collecting Flask==1.0.2 (from -r requirements.txt (line 2))
  Downloading https://files.pythonhosted.org/packages/77/4c/8859774a0451b617475143c949638834487f04a090780728b0b4b9f/Flask-1.0.2-py2.py3-none-any.whl (91kB)
Collecting itsdangerous==0.24 (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/d6/34/6bb3cd395c8b166880817515d8373b2773c1e1dd0271183194059/itsdangerous-0.24.tar.gz (44kB)
Collecting Jinja2==2.10 (from -r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/c5/77/46167b20c9067bba4477a79c076158832ba73173e2e72.10-py2.py3-none-any.whl (106kB)
Collecting MarkupSafe==0.8 (from -r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/46/ae/s3041d1d16d31d7a88073d57801e17a44795cd999a4b1c0d4173b/MarkupSafe-0.8.tar.gz
Collecting Werkzeug==0.14.1 (from -r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/0b/c4/17d5e6d75d5375a03c47bdc7d1b183ef6687e18d00004f1d35345/Werkzeug-0.14.1-py2.py3-none-any.whl (327kB)
Building wheels for collected packages: itsdangerous, MarkupSafe
  Running setup.py bdist_wheel for itsdangerous: started
  Running setup.py bdist_wheel for itsdangerous: finished with status 'done'
  Stored in directory: /root/.cache/pip/wheels/52/46/46/5599d31c155478dc29b04b0c707517918374c0a87f1e5
  Running setup.py bdist_wheel for MarkupSafe: started
  Running setup.py bdist_wheel for MarkupSafe: finished with status 'done'
  Stored in directory: /root/.cache/pip/wheels/33/90/20/bef8dc0277f6c5e63214b0d090f6e647678001ee4e6
Successfully built itsdangerous MarkupSafe
Installing collected packages: click, itsdangerous, MarkupSafe, Jinja2, Werkzeug, Flask
Successfully installed Flask-1.0.2 Jinja2-2.10 MarkupSafe-0.8 Werkzeug-0.14.1 click-6.7 itsdangerous-0.24
Removing intermediate container 8153a4d0d07
--> 66033a37c6
Step 7/8 : ENTRYPOINT [ "python" ]
--> Running in bdc1c3815d
Removing intermediate container bdc1c3815d
--> 73ef03bdc6
Step 8/8 : CMD [ "app.py" ]
--> Running in a76403b0d7
Removing intermediate container a76403b0d7
--> a8a0d37cd5
Successfully built a8a0d37cd5
Successfully tagged app:latest
kunal@kunal-mbp:~$ docker images
```

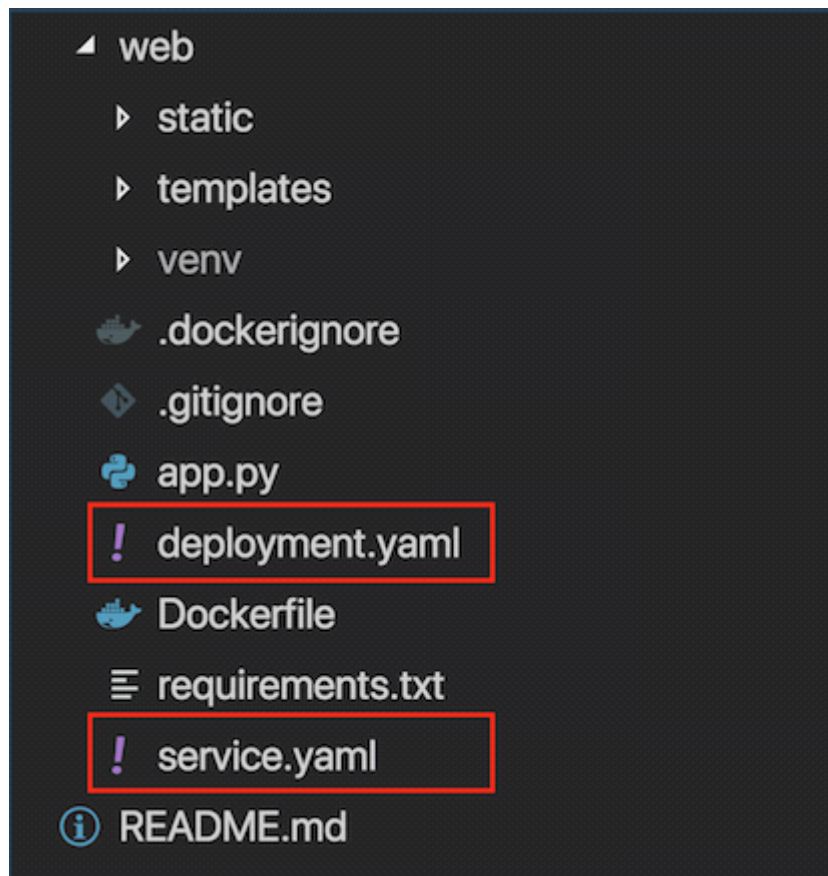
Push the image to the IBM Cloud Registry:

- 1) go to IBM Cloud Kubernetes Service.
- 2) select Private Repositories.
- 3) ibmcloud plugin install container-registry -r "IBM Cloud"
- 4) ibmcloud login -a <cloud_foundry_end_point_for_the_region>
- 5) ibmcloud cr namespace-add <namespace>
- 6) ibmcloud cr login
- 7) docker tag <image_name> <region_url>/<namespace>/<image_name>:<tag>
- 8) docker push <region_url>/<namespace>/<image_name>:<tag>
- 9) ibmcloud cr image-list

```
kunal@kunal-mbp:~$ ibmcloud cr image-list
Listing images...

REPOSITORY                                TAG      DIGEST      NAMESPACE  CREATED    SIZE    SECURITY STATUS
registry.ng.bluemix.net/flask-node/app    latest   b721d6708fe0 flask-node  1 day ago  366 MB  3 Issues
OK
kunal@kunal-mbp:~$
```

Create configuration files for Kubernetes:



deployment.yaml :

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: flask-node-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: flasknode
  template:
    metadata:
      labels:
        app: flasknode
    spec:
      containers:
      - name: flasknode
        image: registry.ng.bluemix.net/flask-node/app
        imagePullPolicy: Always
      ports:
```

- containerPort: 5000

service.yaml:

```
apiVersion: v1
kind: Service
metadata:
  name: flask-node-deployment
spec:
  ports:
    - port: 5000
      targetPort: 5000
  selector:
    app: flasknode
```

Deploy your application to Kubernetes:

- 1) ibmcloud cs region-set us-south
- 2.a) ibmcloud cs cluster-config cluster_kunal
- 2.b) > export KUBECONFIG=/Users/\$USER/.bluemix/plugins/container-service/clusters/<cluster_name>/<cluster_configuration_file.yaml>
- 3) kubectl get nodes
- 4) kubectl create -f deployment.yaml
- 5) kubectl create -f service.yaml
- 6) Look at the Kubernetes dashboard from the IBM Kubernetes Service overview page.

The screenshot shows the IBM Kubernetes Service (K8S) dashboard. The left sidebar contains a navigation menu with options like Cluster, Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, Namespace (default), Overview, Workloads (Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets), Discovery and Load Balancing (Ingresses, Services), and Config and Storage (Config Maps, Persistent Volume Claims). The main content area is titled 'Overview' and displays several tables:

- Deployments:** A table with columns Name, Labels, Pods, Age, and Images. It shows one deployment: 'flask-node-deployment' with 1/1 pods, 5 minutes age, and image 'registry.ng.bluemix.net/flask-node/app'.
- Pods:** A table with columns Name, Node, Status, Restarts, Age, CPU (cores), and Memory (bytes). It shows one pod: 'flask-node-deployment-5cd9c6f6bc-d6n6x' on node '10.47.79.201', status 'Running', 0 restarts, 5 minutes age, 0 CPU cores, and 19.352 Mi memory.
- Replica Sets:** A table with columns Name, Labels, Pods, Age, and Images. It shows one replica set: 'flask-node-deployment-5cd9c6f6bc' with 1/1 pods, 5 minutes age, and image 'registry.ng.bluemix.net/flask-node/app'.
- Services:** A table with columns Name, Labels, Cluster IP, Internal endpoints, External endpoints, and Age. It shows two services: 'kubernetes' (Cluster IP: 172.21.0.1, Internal endpoints: kubernetes:443 TCP, kubernetes:0 TCP, Age: a minute) and 'flask-node-deployment' (Cluster IP: 172.21.104.14, Internal endpoints: flask-node-deployment:5000 TCP, flask-node-deployment:0 TCP, Age: a minute).

- 7) Finally, go to your browser and ping the Public IP of your worker node.

