

Deploy, Update, and Scale Microservices using Serverless



Estimated Time: 30 minutes

In this lab, you will learn how to create a UK University information application composed of two microservices on Code Engine, and how to scale and update each microservice independently of the other. IBM Cloud Code Engine has been made available to you through this lab environment.

This application uses a JSON file which is obtained from [this link](#), filtering out on universities in the UK.

Learning Objectives:

After completing this lab you will be able to:

1. Start Code Engine service to create an application
2. Use the code engine service to deploy two microservices that are part of an application.
3. Scale one of the microservices to have more than one instance.
4. Update one of the microservices without having to stop and redeploy it.

Deploying the microservices

1. Go to the Code engine CLI terminal. If you don't have one, click below to set it up.

► Click here to see how to set up one

2. In the Code Engine CLI, clone the repository https://github.com/ibm-developer-skills-network/gentu-microservices_practiceProj.git by running the following command.

1. 1
1. `git clone https://github.com/ibm-developer-skills-network/gentu-microservices_practiceProj.git`

Copied! Executed!

3. Change to the `universities` directory in the project that you just cloned.

1. 1
1. `cd gentu-microservices_practiceProj/universities`

Copied! Executed!

4. List to see the directories inside it each of which have the microservices that we will deploy on Code Engine.

1. 1
1. `ls`

Copied! Executed!

You will see two directories listed.

- `listing`
- `websites`

The `listing` directory contains the microservice that will list all universities or a list of universities based on keyword search.

The `websites` directory contains the microservice that will provide the list of websites given a college name. You will later update the application to return websites for all colleges that match a keyword search.

5. Run the following command to deploy the `listing` as a microservice.

1. 1
1. `ibmcloud ce app create --name listing --image us.icr.io/${SN_ICR_NAMESPACE}/listing --registry-secret icr-secret --port 5000 --build-context-dir listing --build-source .`

Copied! Executed!

This takes a while.

```
ibmcloud ce app create --name listing --image us.icr.io/${SN_ICR_NAMESPACE}/listing --re
gistry-secret icr-secret --port 5000 --build-context-dir listing --build-source .
Creating application 'listing'...
Packaging files to upload from source path '.'...
Submitting build run 'listing-run-230217-01302675'...
Creating image 'us.icr.io/sn-labs-lavanyas/listing:230217-0630-nw5sj'...
Waiting for build run to complete...
Build run status: 'Running'
Build run completed successfully.
Run 'ibmcloud ce buildrun get -n listing-run-230217-01302675' to check the build run sta
tus.
Waiting for application 'listing' to become ready.
Configuration 'listing' is waiting for a Revision to become ready.
Ingress has not yet been reconciled.
Waiting for load balancer to be ready.
Run 'ibmcloud ce application get -n listing' to check the application status.
OK
```

6. Now run the following command to get the details about the app. This will also list URL that you can access the endpoints through. Copy the URL.

1. 1
1. `ibmcloud ce app get -n listing`

Copied! Executed!

```
theia@theiadocker-lavanyas:/home/project/gentu-microservices_practiceProj/universities$ ibmcloud
application get -n listing
For troubleshooting information visit: https://cloud.ibm.com/docs/codeengine?topic=codeengine-tr
leshoot-apps.
Run 'ibmcloud ce application events -n listing' to get the system events of the application insta
es.
Run 'ibmcloud ce application logs -f -n listing' to follow the logs of the application instances.
OK

Name:          listing
ID:            f2bf0509-d89f-4968-8761-dce35ab2e3ba
Project Name:  Code Engine - sn-labs-lavanyas
Project ID:    1b755f2b-32d2-4aa2-93f5-ce505b849674
Age:           3m37s
Created:       2023-02-17T01:30:30-05:00
URL:           https://listing.z02zthu24x2.us-south.codeengine.appdomain.cloud
Cluster Local URL: http://listing.z02zthu24x2.svc.cluster.local
Console URL:   https://cloud.ibm.com/codeengine/project/us-south/1b755f2b-32d2-4aa2-93f5-ce5
b849674/application/listing/configuration
Status Summary: Application deployed successfully
```

7. Now try to access the endpoint you copied previously by replacing it in the following command. This command will fetch all the colleges which have the name Trinity in it.

- 1
1. curl <your deploymenturl>/colleges/Trinity

Copied!

This will take a while as no instances are running.

```
theia@theiadocker-lavanyas:/home/project/gentu-microservices_practiceProj/universities$ curl
https://listing.z02zthu24x2.us-south.codeengine.appdomain.cloud/colleges/Trinity
{
  "Colleges": [
    "Trinity College of Music",
    "Trinity College Carmarthen",
    "Leeds Trinity University",
    "Trinity College of Music",
    "Trinity College Carmarthen",
    "Leeds Trinity University"
  ]
}
```

8. Now get the details of the deployment again.

- 1
1. ibmcloud ce app get --name listing -q

Copied!

Executed!

You can see that the instances are listed now as you just accessed the endpoint. But this instance will be terminated again when there are no more requests. You can verify the same by running the same command after a gap of 5 mins.

9. Run the following command to deploy the websites as a microservice.

- 1
1. ibmcloud ce app create --name websites --image us.icr.io/\${SN_ICR_NAMESPACE}/websites --registry-secret icr-secret --port 5000 --build-context-dir websites --build-source .

Copied!

Executed!

This takes a while and once it successfully deploys, you will get a URL that you can access the endpoints through. Copy the URL.

10. Now try to access the endpoint you copied previously by replacing it in the following command. This command will fetch all the websites for the college Trinity College of Music.

- 1
1. curl <your deploymenturl>/websites/Trinity%20College%20of%20Music

Copied!

%20 is URL encoding for blank space.

11. There is no keyword search for the websites. Only if the college name exactly matches the value that is passed, the websites are fetched. Try to access the endpoint you copied previously by replacing it in the following command. This command will fetch no websites.

- 1
1. curl <your deploymenturl>/websites/Trinity

Copied!

It will return nothing.

Scale and update the microservices in the application

Unlike a monolithic application, having the application composed of multiple microservices is very advantageous. It allows us to scale the microservices and/or update them independently without affecting the whole application, having to redeploy it or restart it. For example, if one service is seeing a lot of hits then we can scale just that one.

In this example, we will imagine two scenarios:

1. The listings API endpoint gets a lot of hits and has to be scaled.
2. After creating the websites API endpoint, based on user experience, you decide to offer a keyword search instead of an exact word search.
3. Run the following command to scale the listing application to three instances.

- 1
1. ibmcloud ce app update --name listing --min 3

This will scale the minimum number of instances to 3. At any point in time, there will be 3 instances of the application running.

2. Run the following command to see if the `listing` application has scaled.

1. 1
1. `ibmcloud ce app get --name listing -q`

Copied! Executed!

Wait for the application to scale and all instances to run.

3. Now when you run the `curl` command to access the endpoint, it will be much faster as the instance is already running 3 instances.

Updating the microservice

You will now update the microservice serving websites API Endpoint to take keyword search as a parameter instead of giving the whole college name.

1. Click on the button below to open `app.py` in the websites directory.

Open app.py in IDE

2. Replace the implementation of the API endpoint, with the following code. This will enable the microservice to provide keyword searches for websites.

The updated code should be as below.

```

1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
9. 9
10. 10
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
20. 20
21. 21
22. 22

1. from flask import Flask
2. from flask_cors import CORS
3. import json
4.
5. app = Flask("University Websites")
6. CORS(app)
7.
8. with open("UK_Universities.json", "r") as unfile:
9.     data = json.load(unfile)
10.
11.
12. @app.route("/websites/<name>")
13. def getCollegesWebsites(name):
14.     webpages = []
15.     for college in data:
16.         if college["name"].__contains__(name):
17.             for website in college["web_pages"]:
18.                 webpages.append(website)
19.     return json.dumps({"Webpages": webpages}, indent=4)
20.
21. if __name__ == "__main__":
22.     app.run(debug=True, port=5001)

```

Copied!

3. Run the following command to update the application.

1. 1
1. `ibmcloud ce app update --name websites --image us.icr.io/${SN_ICR_NAMESPACE}/websites --registry-secret icr-secret --port 5000 --build-context-dir websites --build-source .`

Copied! Executed!

4. This will update the existing application. To verify the same run the following command. You may recollect from the previous exercise that this did not return any websites as the previous code did not support keyword search.

1. 1
1. `curl <your deploymenturl>/websites/Trinity`

Copied!

You will see the websites of all colleges which have Trinity in their name listed.

Practice Exercise:

1. Scale the website microservice to have 5 minimum instances.
2. Update the `listings` microservices to retrieve the list of universities in a case-insensitive manner. Verify the same passing "TriniTY", as a parameter. If successfully updated, the endpoint should return all the universities with the work 'Trinity'.

Congratulations! You have completed this lab successfully and deployed your application composed of two microservices on Code Engine, scaled on microservice, and updated the other microservice.

Author(s)

Lavanya T S

Changelog

Date	Version	Changed by	Change Description
2023-02-07	0.1	Lavanya	Initial version created

© IBM Corporation 2023. All rights reserved.