SOHAM MEHTA
CSEN 241 CLOUD COMPUTING
ASSIGNMENT – II
CSEN 241

Figlet Deployment

```
soham@soham-System-Product-Name: ~/faasd
Calling the OpenFaaS server to validate the credentials...
credentials saved for admin http://127.0.0.1:8080
soham@soham-System-Product-Name:~/faasd$ faas-cli store list

FUNCTION              AUTHOR       DESCRIPTION
nodeinfo              openfaas     NodeInfo
env                   openfaas     env
sleep                 openfaas     sleep
shasum                openfaas     shasum
figlet                openfaas     figlet
printer               openfaas     printer
curl                  openfaas     curl
external-ip           openfaas     external-ip
youtube-dl            openfaas     youtube-dl
sentimentanalysis     openfaas     SentimentAnalysis
hey                   openfaas     hey
nslookup              openfaas     nslookup
certinfo              stefanprodan SSL/TLS cert info
colorise              alexellis    Colorization
inception             alexellis    Inception
alpine                openfaas     alpine
face-detect-pigo      esimov       Face Detection with Pigo
ocr                   viveksyngh   Tesseract OCR
qrcode-go             alexellis    QR Code Generator - Go
```

```
soham@soham-System-Product-Name: ~/faasd

soham@soham-System-Product-Name:~/faasd$ faas-cli store deploy figlet

Deployed. 200 OK.
URL: http://127.0.0.1:8080/function/figlet

soham@soham-System-Product-Name:~/faasd$ faas-cli store inspect figlet
Title:       figlet
Author:      openfaas
Description:
Generate ASCII logos with the figlet CLI

Image:    ghcr.io/openfaas/figlet:latest
Process:  figlet
Repo URL: https://github.com/openfaas/store-functions
soham@soham-System-Product-Name:~/faasd$ echo "Hello FaaS world" | faas-cli invo
ke figlet
 _   _      _ _        _____          ____                  _      _
| | | | ___| | | ___   |  ___|_ _  __ _/ ___|  __      _____  _ __| | __| |
| |_| |/ _ \ | |/ _ \  | |_ / _` |/ _` \___ \  \ \ /\ / / _ \| '__| |/ _` |
|  _  |  __/ | | (_) | |  _| (_| | (_| |___) |  \ V  V / (_) | |  | | (_| |
|_| |_|\___|_|_|\___/  |_|  \__,_|\__,_|____/    \_/\_/ \___/|_|  |_|\__,_|

soham@soham-System-Product-Name:~/faasd$
```

Deploy figlet

```
$ faas-cli store deploy figlet
```

Find the URLs for the function

```
$ faas-cli store inspect figlet
```

Create some ASCII

```
$ echo "Hello, FaaS, world" | faas-cli invoke figlet
```

Slack Functions (Building, Pushing & Deploying of slack-interactive and slack-request)

For Slack integrations, we're going to develop bespoke Slack functions that operate within the OpenFaaS framework, akin to how the figlet function does. While figlet is a pre-made function that can be readily deployed from the OpenFaaS Function Store, the Slack functions we intend to create will be built from the ground up.

The following commands were used to deploy the functions:

```
$ faas-cli build -f ./slack-interactive.yml
$ faas-cli push -f ./slack-interactive.yml
$ faas-cli deploy -f ./slack-interactive.yml
$ faas-cli build -f ./slack-request.yml
$ faas-cli push -f ./slack-request.yml
$ faas-cli deploy -f ./slack-request.yml
```

To initiate the development of a slack-interactive function, the initial step involves generating the foundational code structure. This can be accomplished with the OpenFaaS CLI using the command below:

```
faas-cli new --lang python slack-interactive
```

After modifying the files with the necessary code changes, the subsequent action is to roll out the functions to your OpenFaaS environment.

handler.py
~/functions/slack-interactive

```python
import json
import urllib

def handle(req):
    urlstring = urllib.unquote(req).decode('utf8').strip('payload=')
    response = json.loads(urlstring)
    data = {
        "attachments": [
            {
                "replace_original": True,
                "response_type": "ephemeral",
                "fallback": "Required plain-text summary of the attachment.",
                "color": "#36a64f",
                "pretext": "Ahh yeah! Great choice, COEN 241 is absolutely amazing!",
                "author_name": "Soham Mehta",
                "author_link": "https://github.com/sohammehta777/cloud-computing.git",
                "author_icon": "https://avatars.githubusercontent.com/u/94602854”,
                "title": "COEN 241",
                "title_link": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/",
                "text": "Head over to COEN 241",
                "image_url": "https://www.scu.edu/media/offices/umc/scu-brand-guidelines/visual-identity-amp-photography/visual-identity-toolkit/logos-amp-seals/Mission-Dont3.png",
                "thumb_url": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/",
                "footer": "Slack Apps built on OpenFaas",
                "footer_icon": "https://a.slack-edge.com/45901/marketing/img/_rebrand/meta/slack_hash_256.png",
                "ts": 123456789
            }
        ]
    }
    return json.dumps(data)
```

Loading file “/home/soham/functions/slack-interactive/handler... Python 2 Tab Width: 8 Ln 1, Col 1 INS

To begin crafting a slack-request function, initiate the process by generating the base code using the OpenFaaS CLI. Execute the following in your terminal:

```
faas-cli new --lang python slack-request
```

This command will create the initial code template. Once you've tailored the handler files to your specific requirements, the next step is to proceed with deploying the updated functions to your OpenFaaS platform.

handler.py
~/functions/slack-request

```python
import json

def handle(req):
    data = {
        "text": "Serverless Message",
        "attachments": [{
            "title": "The Awesome world of Cloud Computing! COEN 241",
            "fields": [{
                "title": "Amazing Level",
                "value": "100",
                "short": True
            }],
            "author_name": "Soham Mehta",
            "author_icon": "https://avatars.githubusercontent.com/u/94602854",
            "image_url": "https://avatars.githubusercontent.com/u/94602854"
        },
        {
            "title": "About COEN 241",
            "text": "COEN 241 is the most awesome class ever!."
        },
        {
            "fallback": "Would you recommend COEN 241 to your friends?",
            "title": "Would you recommend COEN 241 to your friends?",
            "callback_id": "response123",
            "color": "#3AA3E3",
            "attachment_type": "default",
            "actions": [
                {
                    "name": "recommend",
                    "text": "Of Course!",
                    "type": "button",
                    "value": "recommend"
                },
                {
                    "name": "definitely",
```

Loading file "/home/soham/functions/slack-request/handler.py... Python 2 Tab Width: 8 Ln 1, Col 1 INS

soham@soham-System-Product-Name: ~/faasd

```
Docker version 25.0.3, build 4debf41
soham@soham-System-Product-Name:~/faasd$ sudo usermod -aG docker ${USER}
soham@soham-System-Product-Name:~/faasd$ newgrp docker
soham@soham-System-Product-Name:~/faasd$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
```

```
soham@soham-System-Product-Name: ~/functions

Processing triggers for man-db (2.10.2-1) ...
soham@soham-System-Product-Name:~/faas$ docker --version
Docker version 25.0.3, build 4debf41
soham@soham-System-Product-Name:~/faas$ sudo usermod -aG docker ${USER}
soham@soham-System-Product-Name:~/faas$ newgrp docker
soham@soham-System-Product-Name:~/faas$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

soham@soham-System-Product-Name:~/faas$ mkdir -p ~/functions && cd ~/functions
faas-cli new --lang python3 slack-request
faas-cli new --lang python3 slack-interactive
2024/02/27 11:33:52 No templates found in current directory.
2024/02/27 11:33:52 Attempting to expand templates from https://github.com/openfaas/templates.git
2024/02/27 11:33:53 Fetched 17 template(s) : [bun csharp dockerfile go java11 java11-vert-x node node14 node16 node17 node18 php7 php8 python python3 python3-debian ruby] from https://github.com/openfaas/templates.git
Folder: slack-request created.

Function created in folder: slack-request
Stack file written: slack-request.yml

Notes:
You have created a Python3 function using the Classic Watchdog.

To include third-party dependencies create a requirements.txt file.

For high-throughput applications, we recommend using the python3-flask
or python3-http templates.

Folder: slack-interactive created.

Function created in folder: slack-interactive
Stack file written: slack-interactive.yml

Notes:
You have created a Python3 function using the Classic Watchdog.

To include third-party dependencies create a requirements.txt file.

For high-throughput applications, we recommend using the python3-flask
or python3-http templates.

soham@soham-System-Product-Name:~/functions$
```

```
soham@soham-System-Product-Name: ~/functions

20a7b70bdf2f: Mounted from library/python

3fc750b41be7: Mounted from library/python

beee9f30bc1f: Mounted from library/python

latest: digest: sha256:9c2354a45c48596dce22567ef3091a2cb44101e27fe8e07c477437c44
bd34e7d size: 4693
[0] < Pushing slack-request [sohamm7/slack-request:latest] done.
[0] Worker done.
soham@soham-System-Product-Name:~/functions$
```

```
soham@soham-System-Product-Name: ~/functions

3fc750b41be7: Mounted from library/python

beee9f30bc1f: Mounted from library/python

latest: digest: sha256:9c2354a45c48596dce22567ef3091a2cb44101e27fe8e07c477437c44
bd34e7d size: 4693
[0] < Pushing slack-request [sohamm7/slack-request:latest] done.
[0] Worker done.
soham@soham-System-Product-Name:~/functions$ faas-cli deploy -f slack-request.ym
l
Deploying: slack-request.

Deployed. 200 OK.
URL: http://127.0.0.1:8080/function/slack-request

soham@soham-System-Product-Name:~/functions$
```

```
soham@soham-System-Product-Name: ~/functions
n
#21 CACHED

#22 [stage-1 11/18] RUN mkdir -p function
#22 CACHED

#23 [stage-1  5/18] RUN addgroup -S app && adduser app -S -G app
#23 CACHED

#24 [stage-1 12/18] RUN touch ./function/__init__.py
#24 CACHED

#25 [stage-1 18/18] RUN chown -R app:app ./ &&   chmod -R 777 /home/app/python
#25 CACHED

#26 exporting to image
#26 exporting layers done
#26 writing image sha256:0fd31a4bd027df8e4a8584c751c7099ef496de3add42759aa69fd25
98e6666bd done
#26 naming to docker.io/sohamm7/slack-interactive:latest done
#26 DONE 0.0s
Image: sohamm7/slack-interactive:latest built.
[0] < Building slack-interactive done in 1.21s.
```

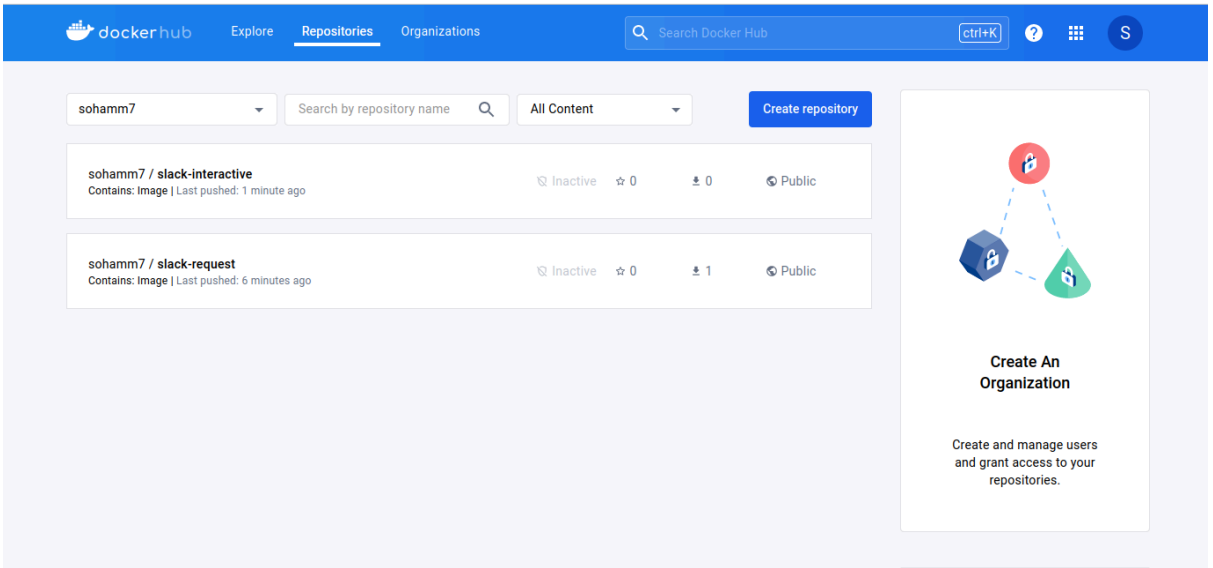
```
soham@soham-System-Product-Name: ~/functions

0c20d92f03f5: Pushing [==================================================>]  8.7
48MB

5305019f4685: Mounted from library/python

0c20d92f03f5: Pushed

d2968c01735e: Mounted from library/python
0c9bfb14c909: Mounted from library/python
678cac8b069e: Mounted from library/python
d4fc045c9e3a: Mounted from library/python
latest: digest: sha256:a5a4192647526d9cf0b7325cf4301083ed9e9a33a4ba7170a55f033ed
9ae805b size: 4900
[0] < Pushing slack-interactive [sohamm7/slack-interactive:latest] done.
[0] Worker done.
soham@soham-System-Product-Name:~/functions$
```

soham@soham-System-Product-Name: ~/functions

```
5305019f4685: Mounted from library/python


0c20d92f03f5: Pushed

d2968c01735e: Mounted from library/python
0c9bfb14c909: Mounted from library/python
678cac8b069e: Mounted from library/python
d4fc045c9e3a: Mounted from library/python
latest: digest: sha256:a5a4192647526d9cf0b7325cf4301083ed9e9a33a4ba7170a55f033ed
9ae805b size: 4900
[0] < Pushing slack-interactive [sohamm7/slack-interactive:latest] done.
[0] Worker done.
soham@soham-System-Product-Name:~/functions$ faas-cli deploy -f slack-interactiv
e.yml
Deploying: slack-interactive.

Deployed. 200 OK.
URL: http://127.0.0.1:8080/function/slack-interactive

soham@soham-System-Product-Name:~/functions$
```

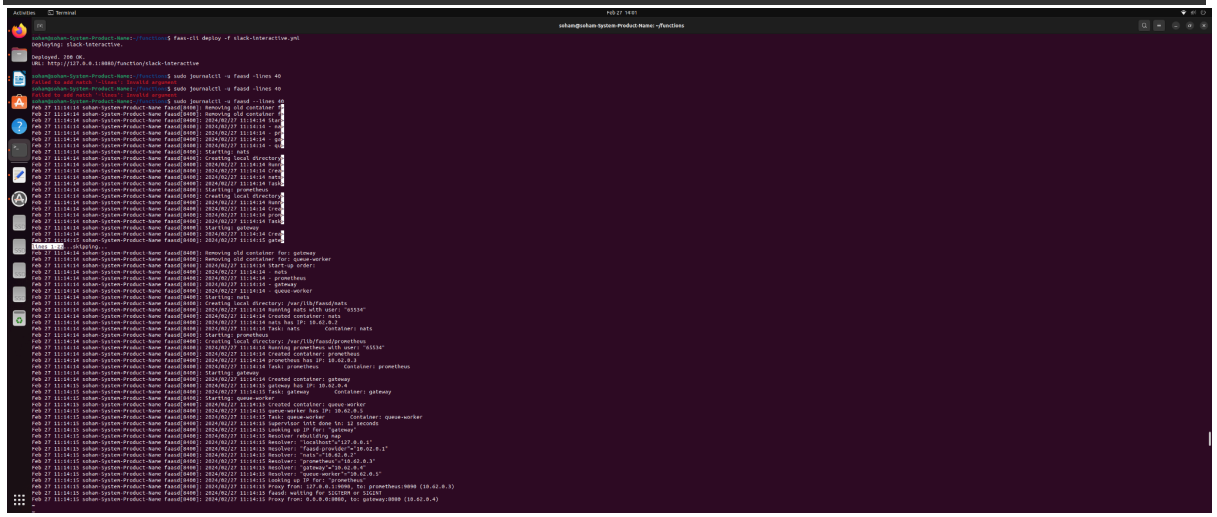
soham@soham-System-Product-Name: ~/functions

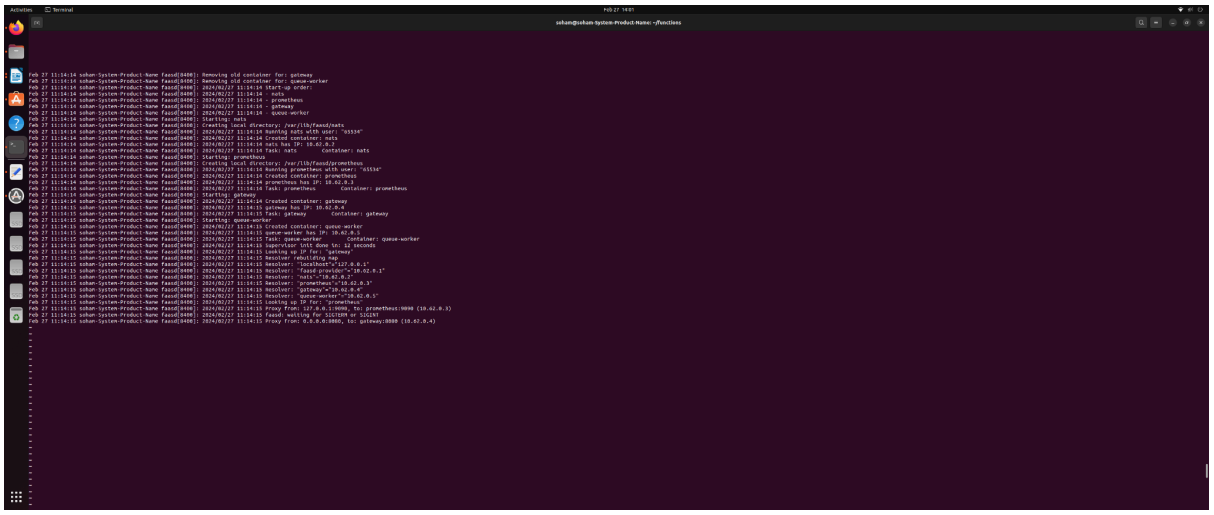
```
~
~
~
~
~
~
lines 1-40/40 (END)
soham@soham-System-Product-Name:~/functions$ ls
build  slack-interactive  slack-interactive.yml  slack-request  slack-request.ym
l  template
soham@soham-System-Product-Name:~/functions$ faas-cli new --lang python chatbot
Folder: chatbot created.
  ___                   _____           ____
 / _ \ _ __  ___ _ __ |  ___|_ _  __ _/ ___|
| | | | '_ \ / _ \ '_ \| |_ / _` |/ _` \___ \
| |_| | |_) |  __/ | | |  _| (_| | (_| |___) |
 \___/| .__/ \___|_| |_|_|  \__,_|\__,_|____/
      |_|


Function created in folder: chatbot
Stack file written: chatbot.yml
soham@soham-System-Product-Name:~/functions$
```

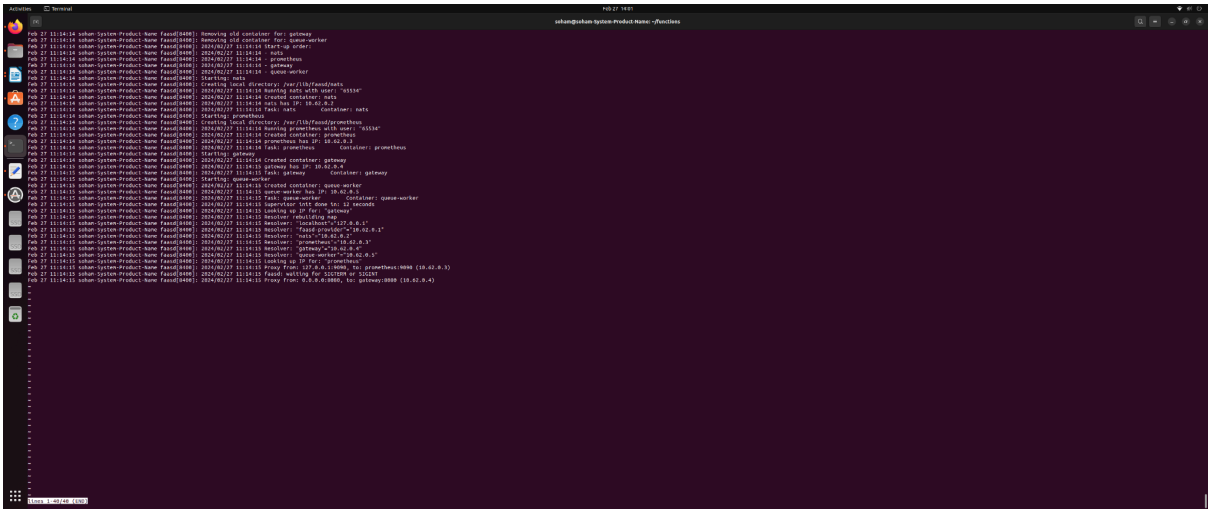


Now, we are running journal CTL command to query and display the logs from journald and sysemd's logging service.

```
$ sudo journalctl -u faasd -lines 40
```



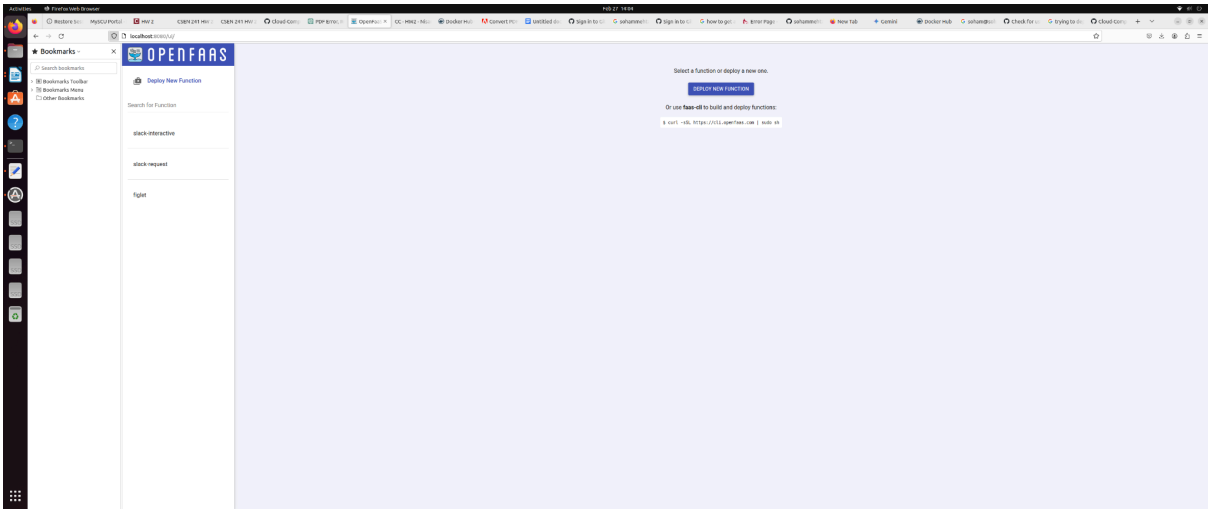




To access the user interface for OpenFaaS, navigate to the following URL in a web browser:

```
http://localhost:8080/ui
```

This will bring up the OpenFaaS graphical interface where you can manage and interact with your deployed functions.



Invoking the functions:

5. Passing Different Arguments to Functions

To pass different arguments to functions in OpenFaaS, you typically utilize the stdin input method. When invoking a function via the OpenFaaS CLI or HTTP request, you can pass arguments through the standard input stream. For example, using the OpenFaaS CLI:

```
echo "author_name,  author_icon" | faas-cli invoke slack-interactive
```

In this command, "argument1 argument2" represents the arguments you want to pass to the function.

Additionally, you can also use environment variables to provide configuration or parameter values to your function. These variables can be set in the function's YAML file or via the OpenFaaS CLI when deploying the function.

6. Changing the slack-interactive Function to React to Different Inputs (3 pts):

To make the slack-interactive function react to different inputs, you would modify the handler code to handle various types of input messages or commands from Slack users. This could involve implementing logic to parse and interpret Slack messages, then perform different actions based on the content of those messages.

For example, you might use conditional statements or switch-case constructs in your Python code to check the content of incoming Slack messages and trigger different functions or responses accordingly.

Additionally, you could utilize the interaction features provided by Slack, such as buttons, menus, or dialogs, to capture user input and provide different responses or actions based on the selected options.

Overall, adapting the slack-interactive function to react to different inputs involves designing and implementing the necessary logic within the function's handler code to handle the various scenarios and interactions expected from Slack users.

Chatbot

handler.py

chatbot > handler.py

```python
import sys
import datetime
import re
import random

# Function to generate FIGlet text
def generate_figlet(input_text):
    # Simulated response for demonstration purposes
    return f"Generated FIGlet for: {input_text}"

# Function to handle user queries
def handle_request(request):
    query = request.lower()

    # Respond to queries based on keywords
    if 'figlet for name' in query:
        response = generate_figlet("ChatBot")
    elif 'figlet for date' in query:
        current_date = datetime.datetime.now().date()
        response = generate_figlet(str(current_date))
    elif 'figlet for time' in query:
        current_time = datetime.datetime.now().time()
        response = generate_figlet(str(current_time))
    elif 'name' in query:
        response = random.choice(["Hello, my name is ChatBot",
                                  "You can call me ChatBot",
                                  "Hello, I am ChatBot"])
    elif 'date' in query:
        current_date = datetime.datetime.now().date()
        response = random.choice(["Current date is: " + str(current_date),
                                  "Today's date is: " + str(current_date),
                                  "At this moment the date reads as " + str(current_date)])
    elif 'time' in query:
        current_time = datetime.datetime.now().time()
        response = random.choice(["Time is: " + str(current_time),
                                  "Right now the clock shows " + str(current_time),
                                  "Current time is: " + str(current_time)])
    elif 'figlet' in query:
        pattern = r"figlet for (.*)"
        match = re.search(pattern, query)
        if match:
            response = generate_figlet(match.group(1))
        else:
            response = "Write figlet query in format: figlet for <text>"
    else:
        response = """I apologize, I didn't understand your question.
                      Please try again with the following questions:
                      1. What is your name?
                      2. What is the current time?
                      3. Generate figlet for Hello World"""

    return response

if __name__ == "__main__":
    input_text = sys.stdin.read().strip()
    print(handle_request(input_text))
```

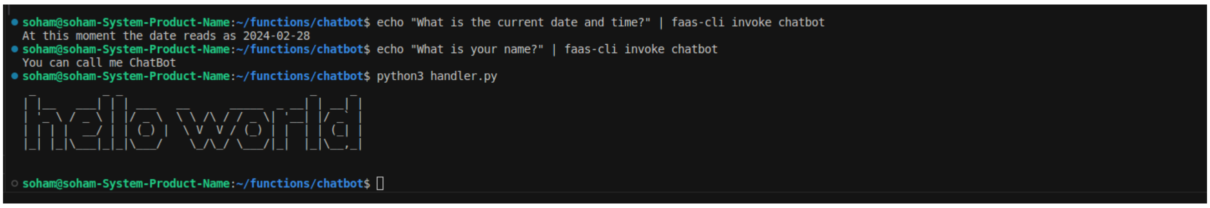
Building & Deploying

```
soham@soham-System-Product-Name:~/functions$ faas-cli up -f chatbot.yml
[0] > Building chatbot.
Building: sohamm7/chatbot:latest with python3 template. Please wait..
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 1.38kB done
#1 DONE 0.0s

#2 [auth] library/python:pull token for registry-1.docker.io
#2 DONE 0.0s

#3 [internal] load metadata for ghcr.io/openfaas/classic-watchdog:0.2.3
#3 DONE 0.6s

#4 [internal] load metadata for docker.io/library/python:3-alpine
#4 DONE 0.6s

#5 [internal] load .dockerignore
#5 transferring context: 2B done
#5 DONE 0.0s

#6 [watchdog 1/1] FROM ghcr.io/openfaas/classic-watchdog:0.2.3@sha256:c3d6717039f6ae49f7041363e629d92a2b95c662f4e626247b9b3647b02cf18e
#6 DONE 0.0s

#7 [stage-1  1/18] FROM docker.io/library/python:3-alpine@sha256:1a0501213b470de000d8432b3caab9d8de5489e9443c2cc7ccaa6b0aa5c3148e
#7 DONE 0.0s

#8 [internal] load build context
#8 transferring context: 3.78kB done
#8 DONE 0.0s

#9 [stage-1  6/18] WORKDIR /home/app/
#9 CACHED

#10 [stage-1  5/18] RUN addgroup -S app && adduser app -S -G app
#10 CACHED

#11 [stage-1 10/18] RUN pip install -r requirements.txt --target=/home/app/python
#11 CACHED

#12 [stage-1  2/18] COPY --from=watchdog /fwatchdog /usr/bin/fwatchdog
#12 CACHED

#13 [stage-1 11/18] RUN mkdir -p function
#13 CACHED

#14 [stage-1  4/18] RUN apk --no-cache add ca-certificates ${ADDITIONAL_PACKAGE}
#14 CACHED

#15 [stage-1  7/18] COPY index.py           .
#15 CACHED

#16 [stage-1 15/18] RUN pip install -r requirements.txt --target=/home/app/python
#16 CACHED

#17 [stage-1 13/18] WORKDIR /home/app/function/
#17 CACHED

#18 [stage-1 12/18] RUN touch ./function/__init__.py
#18 CACHED

#19 [stage-1  3/18] RUN chmod +x /usr/bin/fwatchdog
#19 CACHED

#20 [stage-1 14/18] COPY function/requirements.txt      .
#20 CACHED

#21 [stage-1  8/18] COPY requirements.txt   .
#21 CACHED

#22 [stage-1  9/18] RUN chown -R app /home/app &&   mkdir -p /home/app/python && chown -R app /home/app
#22 CACHED
```

```
#14 [stage-1  4/18] RUN apk --no-cache add ca-certificates ${ADDITIONAL_PACKAGE}
#14 CACHED

#15 [stage-1  7/18] COPY index.py           .
#15 CACHED

#16 [stage-1 15/18] RUN pip install -r requirements.txt --target=/home/app/python
#16 CACHED

#17 [stage-1 13/18] WORKDIR /home/app/function/
#17 CACHED

#18 [stage-1 12/18] RUN touch ./function/__init__.py
#18 CACHED

#19 [stage-1  3/18] RUN chmod +x /usr/bin/fwatchdog
#19 CACHED

#20 [stage-1 14/18] COPY function/requirements.txt      .
#20 CACHED

#21 [stage-1  8/18] COPY requirements.txt   .
#21 CACHED

#22 [stage-1  9/18] RUN chown -R app /home/app &&   mkdir -p /home/app/python && chown -R app /home/app
#22 CACHED

#23 [stage-1 16/18] WORKDIR /home/app/
#23 CACHED

#24 [stage-1 17/18] COPY function           function
#24 DONE 0.0s

#25 [stage-1 18/18] RUN chown -R app:app ./ &&   chmod -R 777 /home/app/python
#25 DONE 1.9s

#26 exporting to image
#26 exporting layers 0.1s done
#26 writing image sha256:9398b962550ba8ad72e9a96c0b39df9531f48cf5e747073b57ce23023273f477 done
#26 naming to docker.io/sohamm7/chatbot:latest done
#26 DONE 0.1s
Image: sohamm7/chatbot:latest built.
[0] < Building chatbot done in 2.87s.
[0] Worker done.

Total build time: 2.87s
[0] > Pushing chatbot [sohamm7/chatbot:latest]
The push refers to repository [docker.io/sohamm7/chatbot]
a44e3735e0fc: Pushed
b4f91c79e66f: Pushed
5f70bf18a086: Layer already exists
8f91ecb0884d: Layer already exists
37b26e91160b: Layer already exists
f3ccccd0d2153: Layer already exists
f8b68c1cdd91: Layer already exists
95bccdfc9e04: Layer already exists
1dd3d2a0f54f: Layer already exists
c4d3b781eded: Layer already exists
381134bbb56a: Layer already exists
08f2f95b8a1c0: Layer already exists
60becc3f1055: Layer already exists
a833d50fa15d: Layer already exists
0c20d92f03f5: Layer already exists
5305019f4605: Layer already exists
d2908c01735e: Layer already exists
0c9bfb14c909: Layer already exists
678cac8b069e: Layer already exists
d4fc045c9e3a: Layer already exists
latest: digest: sha256:5fe1a857f880ff30d7a36e21d586340a58b4974f4d1dcc75cde47bf4e4ffc7c1 size: 4907
[0] < Pushing chatbot [sohamm7/chatbot:latest] done.
[0] Worker done.
Deploying: chatbot.
```

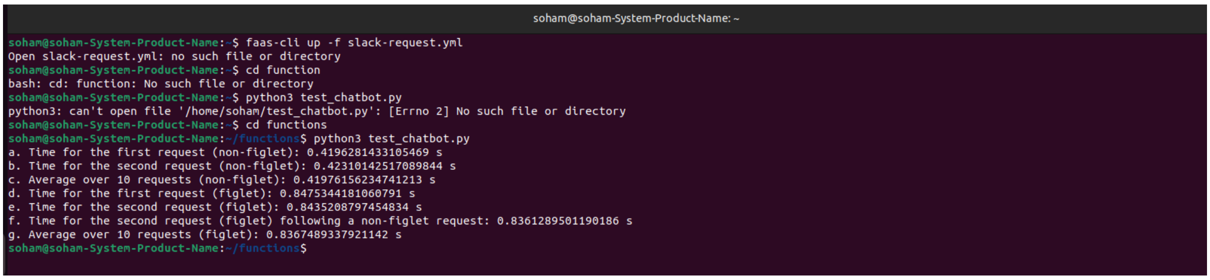
```
Total build time: 0.55s
[0] > Pushing chatbot [sohamm7/chatbot:latest]
The push refers to repository [docker.io/sohamm7/chatbot]
a44e3735e0fc: Layer already exists
b4f91c79e66f: Layer already exists
5f70bf18a086: Layer already exists
8f91ecb0884d: Layer already exists
97b26e91160b: Layer already exists
f3cccd0d2153: Layer already exists
f8b68c1cdd91: Layer already exists
95bccdfc9e04: Layer already exists
1dd3d2a0f54f: Layer already exists
c4d3b781eded: Layer already exists
381134bb556a: Layer already exists
08f2f958a1c6: Layer already exists
66becc3f1055: Layer already exists
a833d50fa15d: Layer already exists
0c20d92f03f5: Layer already exists
5305019f4685: Layer already exists
d2968c01735e: Layer already exists
0c9bfb14c909: Layer already exists
678cac8b069e: Layer already exists
d4fc045c9e3a: Layer already exists
latest: digest: sha256:5fe1a857f888ff30d7a36e21d586340a58b4974f4d1dcc75cde47bf4e4ffc7c1 size: 4907
[0] < Pushing chatbot [sohamm7/chatbot:latest] done.
[0] Worker done.
Deploying: chatbot.
Function chatbot already exists, attempting rolling-update.

Deployed. 200 OK.
URL: http://127.0.0.1:8080/function/chatbot

soham@soham-System-Product-Name:~/functions$
```

Working:



```
● soham@soham-System-Product-Name:~/functions/chatbot$ echo "What is the current date and time?" | faas-cli invoke chatbot
  At this moment the date reads as 2024-02-28
● soham@soham-System-Product-Name:~/functions/chatbot$ echo "What is your name?" | faas-cli invoke chatbot
  You can call me ChatBot
● soham@soham-System-Product-Name:~/functions/chatbot$ python3 handler.py
   _          _ _                            _     _ 
  | |__   ___| | | ___   __      _____  _ __| | __| |
  | '_ \ / _ \ | |/ _ \  \ \ /\ / / _ \| '__| |/ _` |
  | | | |  __/ | | (_) |  \ V  V / (_) | |  | | (_| |
  |_| |_|\___|_|_|\___/    \_/\_/ \___/|_|  |_|\__,_|
                                                     

○ soham@soham-System-Product-Name:~/functions/chatbot$ ▯
```

Average Time & Concurrency



```
soham@soham-System-Product-Name:~$ faas-cli up -f slack-request.yml
Open slack-request.yml: no such file or directory
soham@soham-System-Product-Name:~$ cd function
bash: cd: function: No such file or directory
soham@soham-System-Product-Name:~$ python3 test_chatbot.py
python3: can't open file '/home/soham/test_chatbot.py': [Errno 2] No such file or directory
soham@soham-System-Product-Name:~$ cd functions
soham@soham-System-Product-Name:~/functions$ python3 test_chatbot.py
a. Time for the first request (non-figlet): 0.4196281433105469 s
b. Time for the second request (non-figlet): 0.42310142517089844 s
c. Average over 10 requests (non-figlet): 0.41976156234741213 s
d. Time for the first request (figlet): 0.8475344181060791 s
e. Time for the second request (figlet): 0.8435208797454834 s
f. Time for the second request (figlet) following a non-figlet request: 0.8361289501190186 s
g. Average over 10 requests (figlet): 0.8367489337921142 s
soham@soham-System-Product-Name:~/functions$
```