# Cloud Computing
# COEN 241

*Name: Saurabh Thalkari*

*SCU ID: W1648455*

# *HomeWork – 2*

Github , folder name: HW2

1. Provide a screenshot of invoking the Figlet function (5 pts)


2. Provide a screenshot of running the following command (5 pts)
   sudo journalctl -u faasd --lines 40

```
ubuntu@faasd: $ sudo journalctl -u faasd --lines 40
Feb 22 22:15:55 faasd faasd[4869]: Removing old container for: nats
Feb 22 22:15:55 faasd faasd[4869]: Removing old container for: prometheus
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Start-up order:
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 - nats
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 - prometheus
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 - gateway
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 - queue-worker
Feb 22 22:15:55 faasd faasd[4869]: Starting: nats
Feb 22 22:15:55 faasd faasd[4869]: Creating local directory: /var/lib/faasd/nats
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Running nats with user: "65534"
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Created container: nats
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 nats has IP: 10.62.0.2
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Task: nats        Container: nats
Feb 22 22:15:55 faasd faasd[4869]: Starting: prometheus
Feb 22 22:15:55 faasd faasd[4869]: Creating local directory: /var/lib/faasd/prometheus
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Running prometheus with user: "65534"
Feb 22 22:15:55 faasd faasd[4869]: 2024/02/22 22:15:55 Created container: prometheus
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 prometheus has IP: 10.62.0.3
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Task: prometheus        Container: prometheus
Feb 22 22:15:56 faasd faasd[4869]: Starting: gateway
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Created container: gateway
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 gateway has IP: 10.62.0.4
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Task: gateway        Container: gateway
Feb 22 22:15:56 faasd faasd[4869]: Starting: queue-worker
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Created container: queue-worker
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 queue-worker has IP: 10.62.0.5
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Task: queue-worker        Container: queue-worker
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Supervisor init done in: 12 seconds
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Looking up IP for: "prometheus"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver rebuilding map
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "localhost"="127.0.0.1"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "faasd-provider"="10.62.0.1"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "nats"="10.62.0.2"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "prometheus"="10.62.0.3"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "gateway"="10.62.0.4"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Resolver: "queue-worker"="10.62.0.5"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Looking up IP for: "gateway"
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Proxy from: 0.0.0.0:8080, to: gateway:8080 (10.62.0.4)
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 faasd: waiting for SIGTERM or SIGINT
Feb 22 22:15:56 faasd faasd[4869]: 2024/02/22 22:15:56 Proxy from: 127.0.0.1:9090, to: prometheus:9090 (10.62.0.3)
ubuntu@faasd: $
```

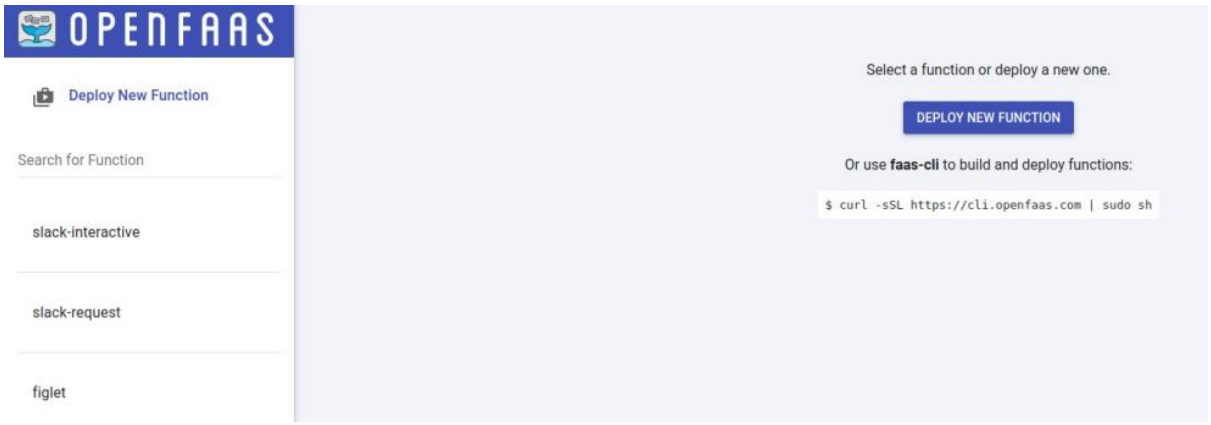
3. Complete slack-request/handler.py (10 pts)

```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": "Saurabh Thalkari",
            "author_icon": "https://github.com/tsuarabh.png",
            "image_url": "https://github.com/tsuarabh.png"
        },
        {
            "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",
                    "text": "Most Definitely!",
                    "type": "button",
                    "value": "definitely"
                }
            ]
        }]
    }
    return json.dumps(data)
~
```

4. Complete slack-interactive/handler.py (10 pts)

```python
import json
import urlparse

def handle(req):
    urlstring = urlparse.unquote(req).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": "",
                "author_link": "https://github.com/tsuarabh",
                "author_icon": "https://github.com/tsuarabh.png",
                "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)
~
```

5. Provide a screenshot of your OpenFaaS gateway AFTER deploying figlet, slack-handler and slack-interactive functions (5 pts)




6. Provide a screenshot of invoking slack-request and slack-interactive functions (5 pts)
Invoking Slack-Request:

Response status

200

Round-trip (s)

0.05

Response body


```
{
  "text": "Serverless Message",
  "attachments": [
    {
      "fields": [
        {
          "short": true,
          "value": "100",
          "title": "Amazing Level"
        }
      ],
      "author_icon": "https://github.com/tsuarabh.png",
      "image_url": "https://github.com/tsuarabh.png",
      "author_name": "Saurabh Thalkari",
      "title": "The Awesome world of Cloud Computing! COEN 241"
    },
    {
      "text": "COEN 241 is the most awesome class ever!.",
      "title": "About COEN 241"
    },
    {
      "title": "Would you recommend COEN 241 to your friends?",
      "color": "#3AA3E3",
      "actions": [
        {
          "text": "Of Course!",
          "type": "button",
          "name": "recommend",
          "value": "recommend"
        },
        {
          "text": "Most Definitely!",
          "type": "button",
          "name": "definitely",
          "value": "definitely"
        }
      ],
      "callback_id": "response123",
      "fallback": "Would you recommend COEN 241 to your friends?",
      "attachment_type": "default"
    }
  ]
}
```


Invoking Slack-Interactive:

Text JSON Download

Request body

"Saurabh"

Response status: 200

Round-trip (s): 0.057

Response body

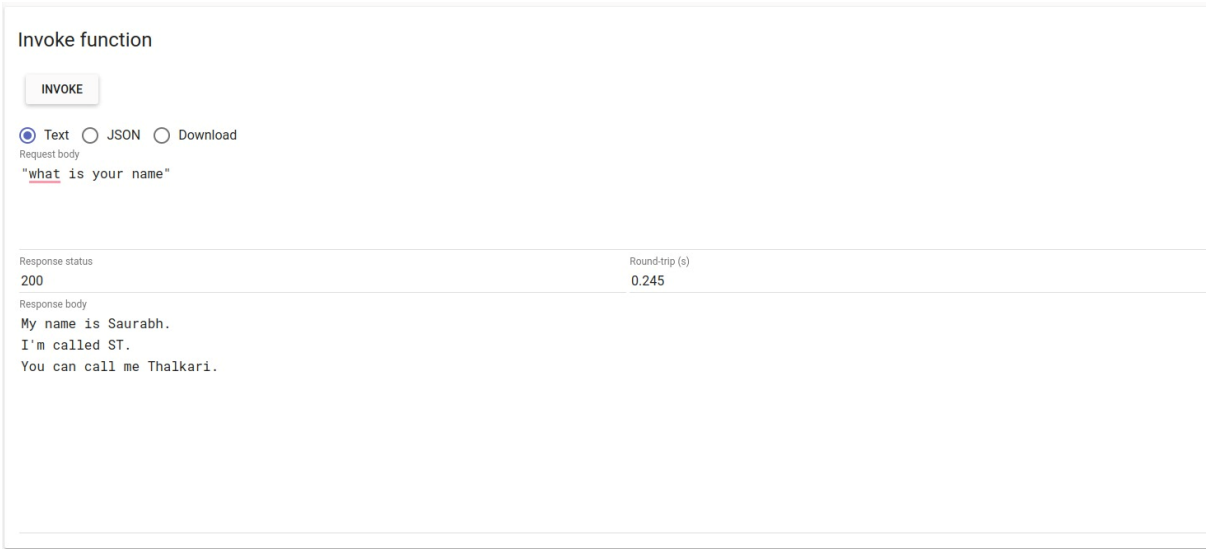

```
{
  "attachments": [
    {
      "footer": "Slack Apps built on OpenFaas",
      "author_link": "https://github.com/tsuarabh",
      "color": "#36a64f",
      "text": "Head over to COEN 241",
      "title": "COEN 241",
      "ts": 123456789,
      "author_name": "",
      "title_link": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/",
      "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",
      "response_type": "ephemeral",
      "replace_original": true,
      "footer_icon": "https://a.slack-edge.com/45901/marketing/img/_rebrand/meta/slack_hash_256.png",
      "pretext": "Ahh yeah! Great choice, COEN 241 is absolutely amazing!",
      "fallback": "Required plain-text summary of the attachment.",
      "thumb_url": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/",
      "author_icon": "https://github.com/tsuarabh.png"
    }
  ]
}
```

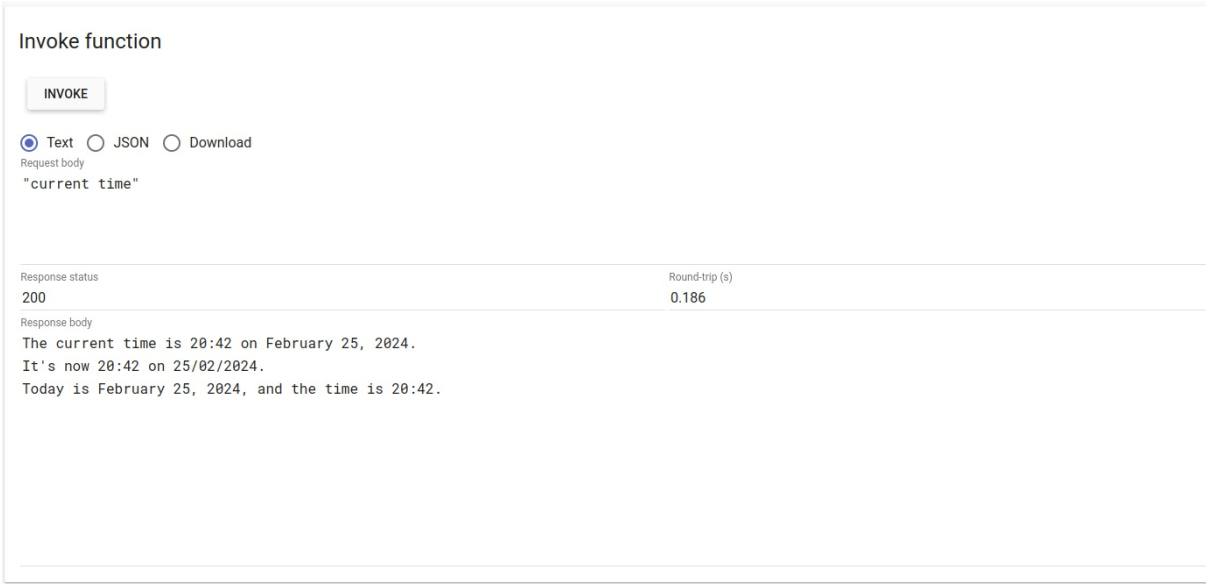

7. Complete the chatbot with a yml file (25pt)

```python
import datetime
from pyfiglet import Figlet
import sys

def handle(req):
    """Process incoming requests based on the input text"""
    if "name" in req.lower() or "what is your name" in req.lower():
        # Respond with the bot's name in 3 different ways
        resp = [
            "My name is Saurabh.",
            "I'm called ST.",
            "You can call me Thalkari."
        ]
        return "\n".join(resp)
    elif "current time" in req.lower() or "current date" in req.lower():
        # Respond with the current date and time in 3 different ways
        now = datetime.datetime.now()
        resp = [
            now.strftime("The current time is %H:%M on %B %d, %Y."),
            now.strftime("It's now %H:%M on %d/%m/%Y."),
            now.strftime("Today is %B %d, %Y, and the time is %H:%M.")
        ]
        return "\n".join(resp)
    elif req.lower().startswith("generate a figlet for"):
        # Extract the text to generate figlet
        text = req[len("generate a figlet for"):].strip("\" ")
        # For the purpose of this example, we'll simulate figlet output using PyFiglet
        f = Figlet(font='slant')
        return f.renderText(text)
    else:
        return "I'm not sure how to process that request."

if __name__ == "__main__":
    # For local testing, input can be sent directly through command line
    req = sys.argv[1] if len(sys.argv) > 1 else ""
    print(handle(req))
```

```
version:1.0
provider:
    name:openfaas
    gateway: http://10.101.54.43:8080
functions:
    chatbot:
        lang:python
        handler:./chatbot
        image:tsaurabh25/chatbot:latest
```

8. Provide a screenshot of invoking three different cases of the chatbot (5 pts)





1. What is the command to invoke the slack-request function (2 pts)?
   a. Via curl :- echo "Saurabh Great" | curl -d @-http://10.101.54.43:8080/function/slack-request
   b. Via faas :- echo "Saurabh Great" | faas-cli invoke slack-request

2. What is the output you see when you invoke the slack-request function? (2 pts)

```
ubuntu@faasd:~/functions$ echo "Saurabh Great" | faas-cli invoke slack-request
{"text": "Serverless Message", "attachments": [{"fields": [{"short": true, "value": "100", "title": "Amazing Level"}], "author_icon": "https://github.com/tsuarabh.png", "image_url": "https://github.com/tsuarabh.png", "author_name": "Saurabh Thalkari", "title": "The Awesome world of Cloud Computing! COEN 241"}, {"text": "COEN 241 is the most awesome class ever!.", "title": "About COEN 241"}, {"title": "Would you recommend COEN 241 to your friends?", "color": "#3AA3E3", "actions": [{"text": "Of Course!", "type": "button", "name": "recommend", "value": "recommend"}, {"text": "Most Definitely!", "type": "button", "name": "definitely", "value": "definitely"}], "callback_id": "response123", "fallback": "Would you recommend COEN 241 to your friends?", "attachment_type": "default"}]}
ubuntu@faasd:~/functions$
```

3. What is the command to invoke the slack-interactive function? (2 pts)
   a. Via curl :- curl -d '{"Saura
   b. ":"COEN 241"}' http://10.101.54.43:8080/function/slack-interactive
   b. Via faas-cli :- sudo faas-cli invoke slack-interactive
4. What is the output you see when you invoke the slack-interactive function? (2 pts)

```
ubuntu@faasd:~/functions$ curl -d '{"Saura":"Coen 241"}' http://10.101.54.43:8080/function/slack-interactive
{"attachments": [{"footer": "Slack Apps built on OpenFaas", "author_link": "https://github.com/tsuarabh", "color": "#36a64f", "text": "Head over to COEN 241", "title": "COEN 241", "ts": 123456789, "author_name": "", "title_link": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/", "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", "response_type": "ephemeral", "replace_original": true, "footer_icon": "https://a.slack-edge.com/45901/marketing/img/_rebrand/meta/slack_hash_256.png", "pretext": "Ahh yeah! Great choice, COEN 241 is absolutely amazing!", "fallback": "Required plain-text summary of the attachment.", "thumb_url": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/", "author_icon": "https://github.com/tsuarabh.png"}]}
ubuntu@faasd:~/functions$
```

```
ubuntu@faasd:~/functions$ sudo faas-cli invoke slack-interactive
Reading from STDIN - hit (Control + D) to stop.
"Thalkari"{"attachments": [{"footer": "Slack Apps built on OpenFaas", "author_link": "https://github.com/tsuarabh", "color": "#36a64f", "text": "Head over to COEN 241", "title": "COEN 241", "ts": 123456789, "author_name": "", "title_link": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/", "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", "response_type": "ephemeral", "replace_original": true, "footer_icon": "https://a.slack-edge.com/45901/marketing/img/_rebrand/meta/slack_hash_256.png", "pretext": "Ahh yeah! Great choice, COEN 241 is absolutely amazing!", "fallback": "Required plain-text summary of the attachment.", "thumb_url": "https://www.scu.edu/engineering/academic-programs/department-of-computer-engineering/graduate/course-descriptions/", "author_icon": "https://github.com/tsuarabh.png"}]}
```

5. How would you pass different arguments to the functions? (3 pts)
   Here are two ways of passing arguments to functions: 1. Via the faas-cli: Arguments can be provided to the figlet function directly through the command line by utilizing the faas-cli. This can be done by piping the string "Hello, FaaS World" into the command like so: echo "Hello, FaaS World" | faas-cli invoke figlet. This method sends the string "Hello, FaaS World" directly to the figlet function as its argument. 2. Through curl: Arguments can be transmitted via a POST request to a web server using curl. For instance: curl -d '{"text":"Hello COEN 241"}' http://example.com/function/slack-request enables sending the specified argument in the request body.

6. How would you change the slack-interactive function to react to different inputs? (3 pts)
   To decode a URL-encoded string into a more readable format, the urllib package's unquote function is employed, typically defaulting to UTF-8 encoding. Following this, the json.load() function is used to convert the JSON string from the URL into a Python dictionary, which is then assigned to the variable named response. Previously, this variable wasn't utilized further. To adapt the slack-interactive function for varied inputs, it's suggested to integrate the contents of the response variable into the function's existing data dictionary. This integration facilitates the function's ability to adapt and reply to a wide range of inputs by outputting json.dumps(data), thus enhancing its response capability based on the data contained in the response
7. How long does it take for the chat response to come back? (10pts)

```python
import requests
import time

# Endpoint of your chatbot
CHATBOT_ENDPOINT = "http://10.101.54.43:8080/function/chatbot"

def calculate_response_time(input_data, iterations=1):
    """Calculate the response time for either a single or an averaged over multiple requests."""
    cumulative_time = 0
    for _ in range(iterations):
        start = time.time()
        response = requests.post(CHATBOT_ENDPOINT, data=input_data)
        finish = time.time()
        cumulative_time += (finish - start)
        if iterations == 1:  # Directly return for a single request
            return finish - start
    return cumulative_time / iterations  # Average time if multiple iterations

def main():
    # Measuring response times for various scenarios
    # a. Time for initial request without invoking figlet
    time_first_no_figlet = calculate_response_time("What is your name?")
    print(f"a. Response time for the initial request (without figlet): {time_first_no_figlet:.4f} seconds")

    # b. Time for second request, also without figlet invocation
    time_second_no_figlet = calculate_response_time("What is your name?")
    print(f"b. Response time for the second request (without figlet): {time_second_no_figlet:.4f} seconds")

    # c. Average time across 10 requests, not using figlet
    avg_time_no_figlet = calculate_response_time("What is your name?", iterations=10)
    print(f"c. Average response time for 10 requests (without figlet): {avg_time_no_figlet:.4f} seconds")

    # d. Response time for initial request that does involve figlet
    time_first_with_figlet = calculate_response_time("Generate a figlet for Hello")
    print(f"d. Response time for the initial request (with figlet): {time_first_with_figlet:.4f} seconds")

    # e. Time for the subsequent request, invoking figlet
    time_second_with_figlet = calculate_response_time("Generate a figlet for Hello")
    print(f"e. Response time for the second request (with figlet): {time_second_with_figlet:.4f} seconds")

    # f. Time for a figlet-invoking request following a non-figlet request
    # Record time for initial non-figlet request
    calculate_response_time("What is your name?")
    # Then, for a figlet request
    time_following_no_figlet = calculate_response_time("Generate a figlet for Hello")
    print(f"f. Response time for the subsequent request (with figlet, after a non-figlet request): {time_following_no_figlet:.4f} seconds")

    # g. Average response time over 10 requests invoking figlet
    avg_time_with_figlet = calculate_response_time("Generate a figlet for Hello", iterations=10)
    print(f"g. Average response time for 10 requests (with figlet): {avg_time_with_figlet:.4f} seconds")
```

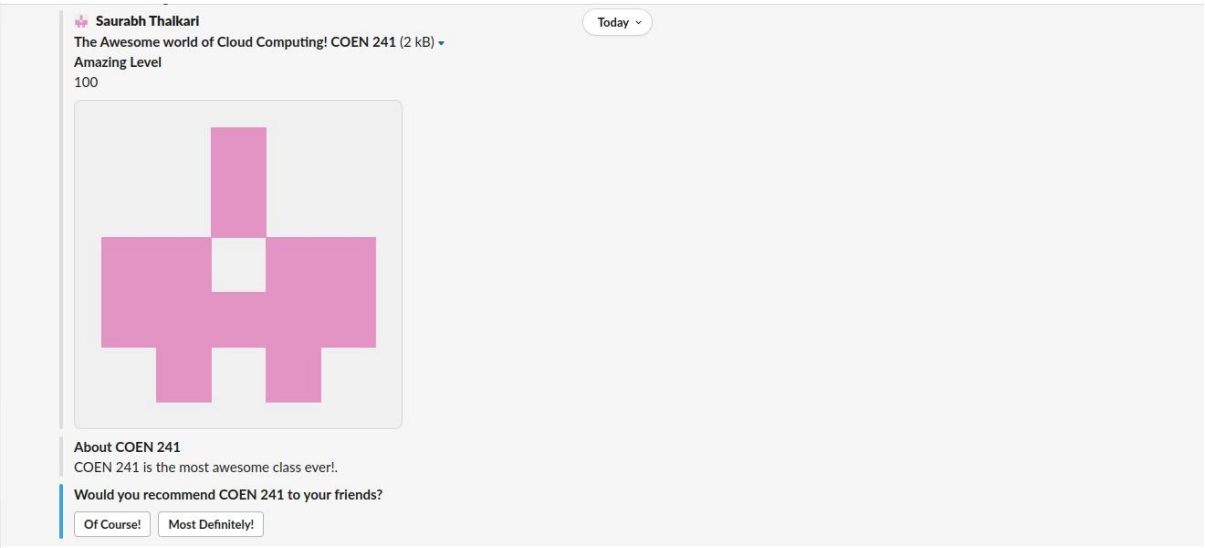
```
a. Response time for the initial request (without figlet): 0.3106 seconds
b. Response time for the second request (without figlet): 0.1738 seconds
c. Average response time for 10 requests (without figlet): 0.1614 seconds
d. Response time for the initial request (with figlet): 0.1893 seconds
e. Response time for the second request (with figlet): 0.1666 seconds
f. Response time for the subsequent request (with figlet, after a non-figlet request): 0.1534 seconds
g. Average response time for 10 requests (with figlet): 0.1559 seconds
```

8. Now try sending a series of requests to the chatbot in parallel. At what queries per second does OpenFaaS add a new instance of the function? (6 pts)

```python
import concurrent.futures
import requests
import time

# Define the URL where the chatbot service is hosted
SERVICE_ENDPOINT = "http://10.101.54.43:8080/function/chatbot-service"

def send_request_to_service(data_payload):
    """Function to send data to the chatbot service endpoint."""
    try:
        response = requests.post(SERVICE_ENDPOINT, data=data_payload)
        return response.status_code
    except Exception as e:
        return str(e)

def perform_concurrent_requests(req_per_sec, duration_sec=10):
    """Function to send concurrent requests to the service."""
    with concurrent.futures.ThreadPoolExecutor() as executor:
        future_tasks = []
        time_start = time.time()

        while time.time() - time_start < duration_sec:
            for _ in range(req_per_sec):
                future = executor.submit(send_request_to_service, "What's the weather like?")
                future_tasks.append(future)
            time.sleep(1)  # Wait before sending the next batch of requests

        results = [future.result() for future in future_tasks]

    successful_results = [result for result in results if result == 200]
    print(f"Total requests sent: {len(results)}")
    print(f"Successful responses: {len(successful_results)}")
    print(f"Success Rate: {(len(successful_results) / len(results)) * 100:.2f}%")

# Example of calling the function:
request_rate = 5  # You can adjust this value to test different load scenarios
perform_concurrent_requests(request_rate)
```

```
Total requests sent: 50
Successful responses: 50
Success Rate: 100.00%
```

# Extra Credit





Invite to workspace:

https://join.slack.com/t/csen241cloudc-e4n4464/shared_invite/zt-2dr1qdfcx-NKWw5kmpbgo4obeyilU3GQ

Link to application: https://app.slack.com/client/T06LT54CLMQ/C06LVJMU684

Slash-command url: https://f737-24-23-244-181.ngrok-free.app/function/slack-request