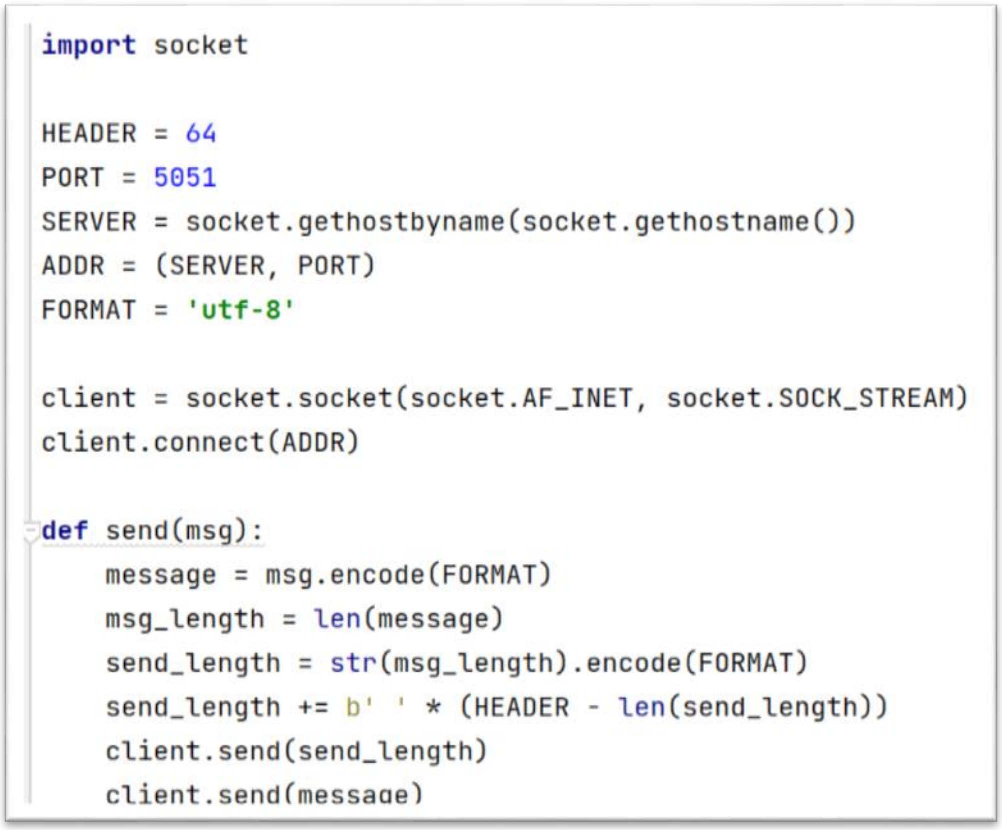
OSU_CS361_partner_microservice

PROJECT TITLE: Implementation of Microservice for Partner

PROJECT DESCRIPTION: Random number generator microservice using python sockets

INSTALLATION/REQUIREMENTS:

- Python
- Program/Client and the Microservice/Server running on the same computer.
- Microservice/Server MUST be started first to avoid connection error.
- PORT may need be changed if the computer is already using the PORT.
- Firewalls may need to be disabled to allow for python sockets to run properly
- Program/Client must contain the following code to connect to the server.



```
import socket

HEADER = 64
PORT = 5051
SERVER = socket.gethostbyname(socket.gethostname())
ADDR = (SERVER, PORT)
FORMAT = 'utf-8'

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(ADDR)

def send(msg):
    message = msg.encode(FORMAT)
    msg_length = len(message)
    send_length = str(msg_length).encode(FORMAT)
    send_length += b' ' * (HEADER - len(send_length))
    client.send(send_length)
    client.send(message)
```

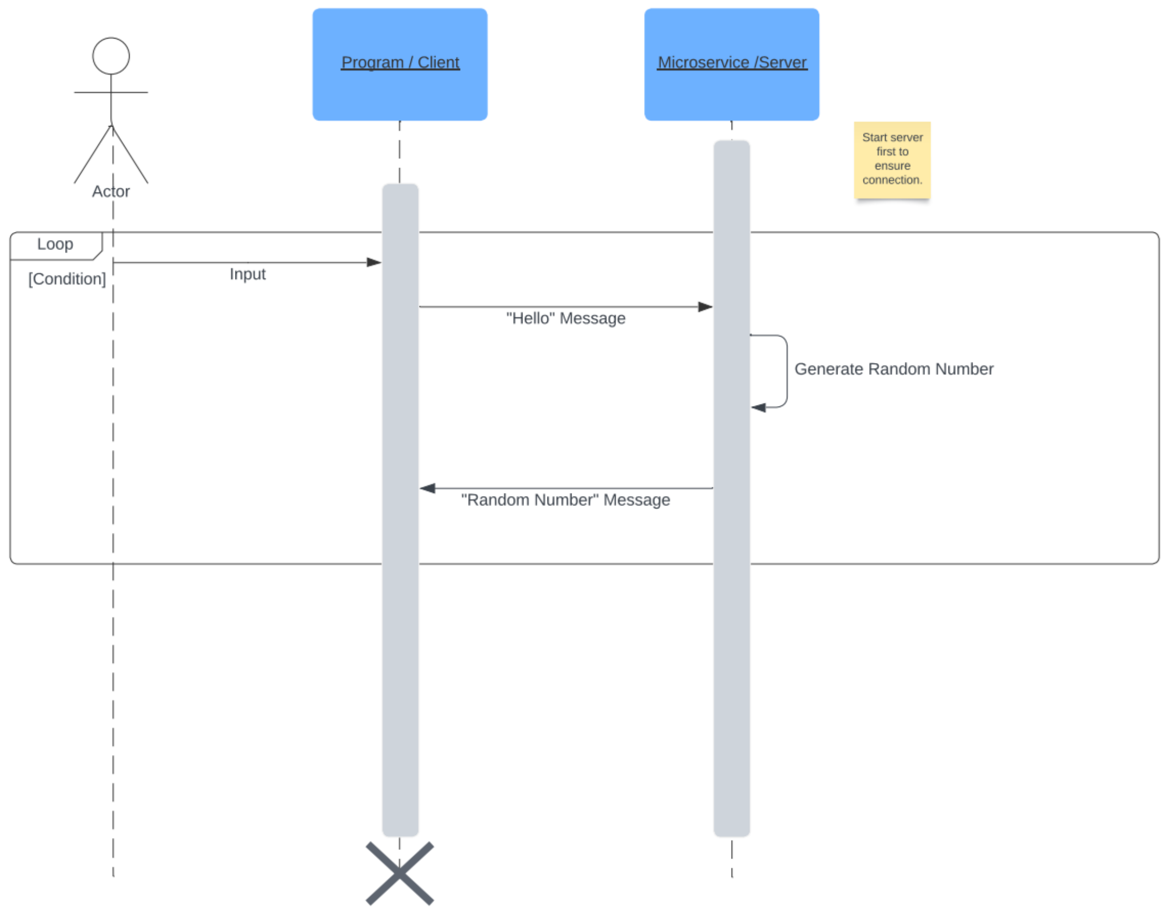
USAGE INSTRUCTIONS TO REQUEST DATA FROM THE MICROSERVICE:

- Using the send() function as embedded/defined in the Program/Client to request data from the microservice.
- Example call **send("Hello")**: Using the example call **send("Hello")** of the send() function with the text message "Hello" will cause the Microservice/Service to generate a random number.
- In the template Program/Client code, an input() function is added prior to the send() so that the User can determine when the random number is being generated. This input() function can be deleted/modified in the final Program/Client code as necessary.

USAGE INSTRUCTIONS TO RECIEVE DATA:

- The data received is the code "client.recv(64).decode(FORMAT))."
- In the template Program/Client code, a print statement was used to show the received data. Delete the print() statement, but utilized the code "client.recv(64).decode(FORMAT)" to use the received data.

UML SEQUENCE DIAGRAM



COMMUNICATION CONTRACT: Jointly-Developed by Makenna and Eddy:

- Preference for how we should be communicating: First option, communicating over Microsoft Teams Chat/Video. Second option, communicating over the phone.
- Expectations for responsiveness: Response target of around one day. If no response after one day, reminder text or phone call to the partner's mobile phone.
- Work synchronously vs asynchronously: Preference for working synchronously over Microsoft Teams or over the phone.
- Method of sharing code: Over GitHub.
- Struggles with course: Discuss together and brainstorm solutions.