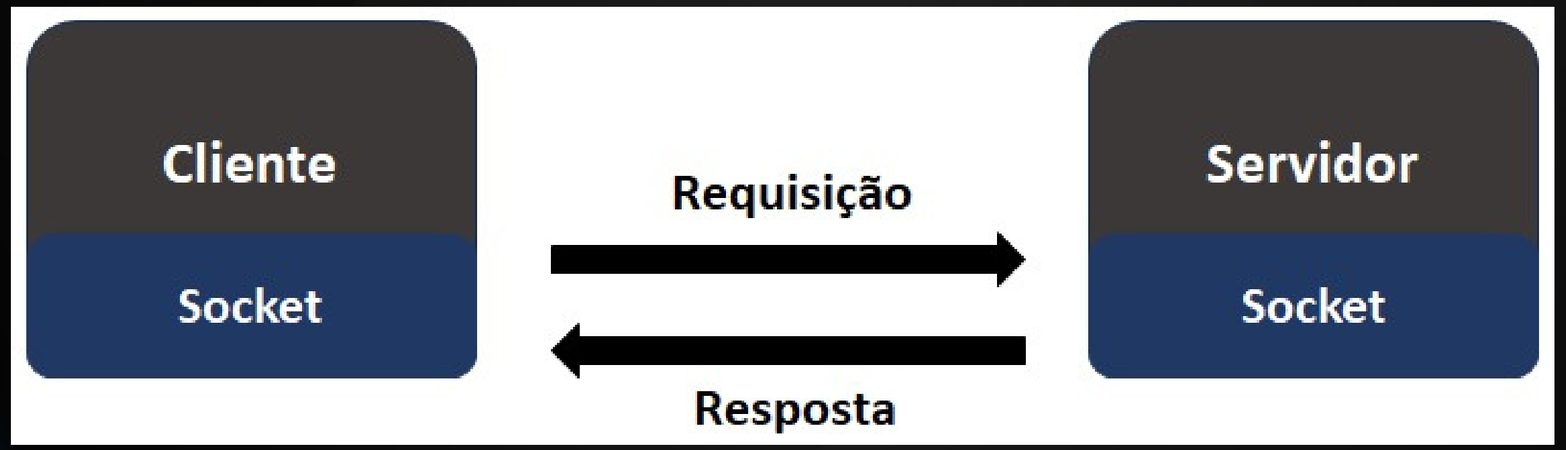
COMUNICAÇÃO

CLIENTE E SERVIDOR

CRIANDO SOCKETS COM PYTHON

Ideia do Projeto

Criar uma comunicação Cliente-Servidor entre o usuário e um serviço, utilizando SOCKETS TCP. O serviço é responsável por prover três questões ao usuário, o qual é responsável por respondê-las ao Servidor. Dessa Maneira, o servidor verifica quais questões foram respondidas corretamente e retorna para o usuário.



Vantagens do uso de SOCKETS

Comunicação em Tempo Real

Sockets permitem a comunicação em tempo real entre dispositivos ou sistemas distribuídos, tornando-os ideais para aplicações que exigem resposta imediata, como chat em tempo real e jogos online.

Implementação de protocolos personalizados

Desenvolvedores têm controle total sobre a comunicação usando sockets, permitindo a implementação de protocolos personalizados e otimizados para atender às necessidades específicas da aplicação. Dito isso, é possível implementar protocolos específicos a nível de aplicação.

Amplo Suporte

Sockets são suportados em várias linguagens de programação e sistemas operacionais, garantindo a interoperabilidade e facilitando o desenvolvimento de aplicações multiplataforma.

Desvantagens do uso de SOCKETS

Complexidade de Programação

A implementação da comunicação por sockets pode ser um tanto quanto complexa e pronpensa a erros. Dessa maneira, pode exigir um conhecimento porfundo a cerca da pilha TCP/IP.

Escalabilidade Limitada

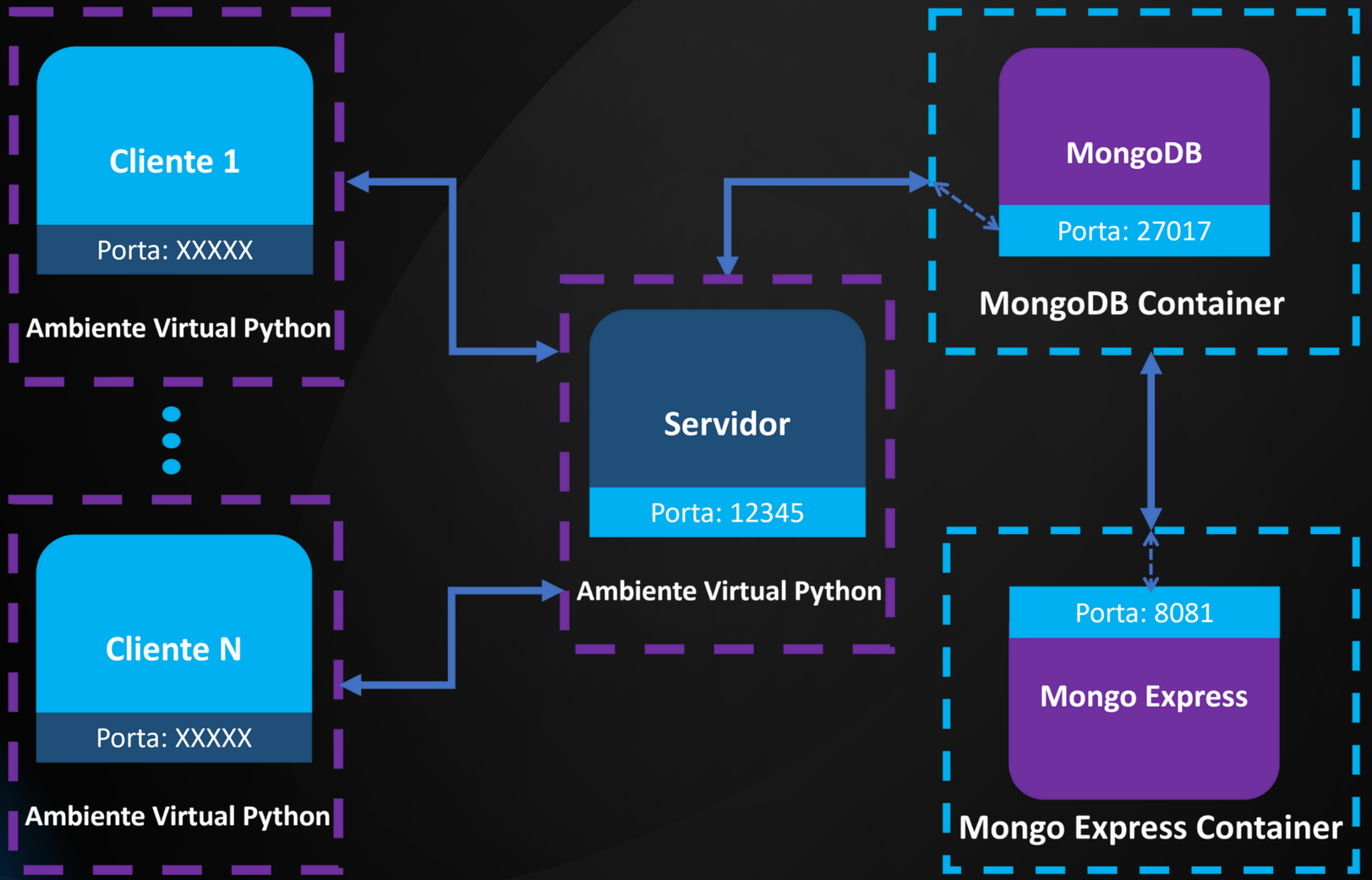
Pode ser um desafio o tratamento de múltiplas conexões simultâneas. Em aplicações de alta escalabilidade, pode ser necessário implementar funcionalidades específicas para o balanceamento de carga.

Gerenciamento de Conexões

O gerenciamento de conexões, incluindo a detecção e o tratamento de desconexões inesperadas, pode ser uma tarefa desafiadora e requer um código robusto para garantir a estabilidade da aplicação.

Desenvolvimento

Arquitetura



docker-compose.yml

```yaml
# Use root/example as user/password credentials
version: '3.1'

services:

  mongo:
    image: mongo
    container_name: mongodb
    restart: always
    ports:
      - 27017:27017
    environment:
      MONGO_INITDB_ROOT_USERNAME: root
      MONGO_INITDB_ROOT_PASSWORD: example
      MONGO_INITDB_DATABASE: C115-Trabalho1

  mongo-express:
    image: mongo-express
    restart: always
    ports:
      - 8081:8081
    environment:
      ME_CONFIG_MONGODB_ADMINUSERNAME: root
      ME_CONFIG_MONGODB_ADMINPASSWORD: example
      ME_CONFIG_MONGODB_URL: mongodb://root:example@mongo:27017/
```

database.py

```python
import pymongo
import json

with open('./src/server/db/samples.json',encoding='utf8') as f:
    dataset = json.load(f)

class Database:
    def __init__(self, database, collection):
        self.connect(database, collection)

    def connect(self, database, collection):
        try:
            connectionString = "mongodb://root:example@localhost:27017"
            self.clusterConnection = pymongo.MongoClient(
                connectionString,
                tlsAllowInvalidCertificates=True
            )
            self.db = self.clusterConnection[database]
            self.collection = self.db[collection]
            print("Database connected successfully!")
        except Exception as e:
            print(e)

    def resetDatabase(self):
        try:
            self.db.drop_collection(self.collection)
            self.collection.insert_many(dataset)
            print("Database reseted successfully!")
        except Exception as e:
            print(e)
```

questionDAO.py

```python
from db.database import Database
from entity.question import Question
from service.questionService import QuestionService
import json

class QuestionDAO:
    def __init__(self, databaseName, collectionName):
        self.__db = Database(databaseName, collectionName)

    def createQuestion(self, question:Question) -> str:
        # Convert a question to JSON document
        questionDocument = QuestionService.setJsonDocumentByQuestion(question=question)

        try:
            result = self.__db.collection.insert_one(questionDocument)
            question_id = result.inserted_id
            print(f"Question {question.getId()} created with id: {question_id}")
        except Exception as error:
            print(f"An error occurred while creating question: {error}")

    def readQuestion(self, id:int):
        try:
            result = self.__db.collection.find_one({"_id": id})
            result = json.dumps(result)
            if result:
                print(f"Question found: {result}")

                return result
            else:
                print(f"No question found with id {id}")
                return None
        except Exception as error:
            print(f"An error occurred while reading question: {error}")
```

client.py

```python
from socket import *
import json

SERVER_PORT = 12345
SERVER_IP = 'localhost'

# Creating the client socket in order to make request to the server
client_socket = socket(AF_INET, SOCK_STREAM)

# Connect to the server (Three-way-handshake)
client_socket.connect((SERVER_IP,SERVER_PORT))
# Receiving questions
while True:
    question = client_socket.recv(1024).decode()
    if question == 'FIM':
        break
    question = json.loads(question)
    print(question['statement'])
    for index, option in enumerate(question['options']):
        print(str(index+1) + '. '+ option)
    answer = int(input('Resposta:'))
    client_socket.send(str(answer-1).encode())
# Receiving Feedback
while True:
    text = client_socket.recv(1024).decode()
    client_socket.send('OK'.encode())
    if text == 'FIM':
        break
    print(text)

client_socket.close()
```

question.py

```python
from typing import List

class Question:
    def __init__(self, id:int, statement:str, options:list, answer:int):
        self.__id = id
        self.__statement = statement
        self.__options = options
        self.__answer = answer

    '''
        Getters
    '''
    def getId(self) -> int:
        return self.__id

    def getStatement(self) -> str:
        return self.__statement

    def getOptions(self) -> List[str]:
        return self.__options

    def getAnswer(self) -> int:
        return self.__answer

    '''
        Setters
    '''
    def setId(self, id):
        self.__id = id

    def setStatement(self, statement):
        self.__statement = statement

    def setOptions(self, options):
        self.__options = options

    def setAnswer(self, answer):
        self.__answer = answer
```

questionService.py

```python
from entity.question import Question
from typing import Dict

class QuestionService:
    @staticmethod
    def setQuestionByJson(document) -> Question:
        # Question information
        id = document["_id"]
        statement = document["statement"]
        options = document["options"]
        answer = document["answer"]

        # Create a question
        question = Question(id=id, statement=statement, options=options, answer=answer)

        return question

    @staticmethod
    def setJsonDocumentByQuestion(question:Question) -> Dict:
        # Create a JSON document based on Question
        documentoQuestion = {
            "_id": question.getId(),
            "statement": question.getStatement(),
            "options": question.getOptions(),
            "answer": question.getAnswer()
        }

        return documentoQuestion
```

server.py

1

```python
from socket import *
import threading
import random
import json

from typing import List

from entity.question import Question
from db.questionDAO import QuestionDAO
from db.database import Database

def colored_text(text, color_code):
    return f"{color_code}{text}\033[0m"

def connection_handler(client_socket, client_address):
    # Using random to get which questions will be solved by the user
    sorted_questions = random.sample(range(1,11), 3)
    # Connect to the Database and get questions
    questions_json_str = getQuestionsFromDatabase(sorted_questions)
    # Send questions to the client
    answered_questions, questions_json = sendQuestions(client_socket, questions_json_str)
    # Send feedback to the client
    sendFeedback(client_socket, answered_questions, questions_json)
    # Close connection
    client_socket.close()

def sendQuestions(client_socket, questions_json_str):
    answered_questions = []
    questions_json = []
    # Iterate through questions sorted in other to send to the client
    for question in questions_json_str:
        client_socket.send(question.encode())
        answer = client_socket.recv(1024)
        answered_questions.append(int(answer.decode()))
        questions_json.append(json.loads(question))
    client_socket.send('FIM'.encode())
    return answered_questions, questions_json

def sendFeedback(client_socket, answered_questions, questions_json):
    # Iterate through questions sorted in other to send feedback to the client
    for index, question in enumerate(questions_json):
        if question['answer'] == answered_questions[index]:
            text = colored_text('Você acertou a questão ' + str(index+1) + '\n', "\033[32m")
        else:
            text = colored_text('Você errou a questão ' + str(index+1) + ' a resposta correta é ' +
str(question['answer'] + 1) + '. ' + question['options'][question['answer']] + '\n', "\033[31m")
        client_socket.send(text.encode())
        client_socket.recv(1024)
    client_socket.send('FIM'.encode())
```

2

```python
def getQuestionsFromDatabase(sorted_questions) -> List[Question]:
    db = QuestionDAO('C115-Trabalho1', 'Questions')
    questions = []
    for question_number in sorted_questions:
        question = db.readQuestion(id=question_number)
        questions.append(question)
        print('Question {question_number} got from DB')
    return questions

# Server IP and PORT for connections
SERVER_PORT = 12345
SERVER_IP = 'localhost'

# Reset database and create new one with the corresponding collection 'Questions'
db = Database('C115-Trabalho1', 'Questions')
db.resetDatabase()

# Create the server socket
server_socket = socket(AF_INET, SOCK_STREAM)

# Bind the socket to the port
server_socket.bind((SERVER_IP, SERVER_PORT))

# Start listening with a maximum of 10 clients in the queue
server_socket.listen(10)

print(f'Server listening on port {SERVER_PORT}')

while True:
    # Accepting connections
    client_socket, client_address = server_socket.accept()
    print(f'Accepted connection from client {client_address}')
    # Create a thread to treat the connection established
    client_thread = threading.Thread(target=connection_handler, args=(client_socket, client_address))
    client_thread.start()

server_socket.close()
```

samples.json

1

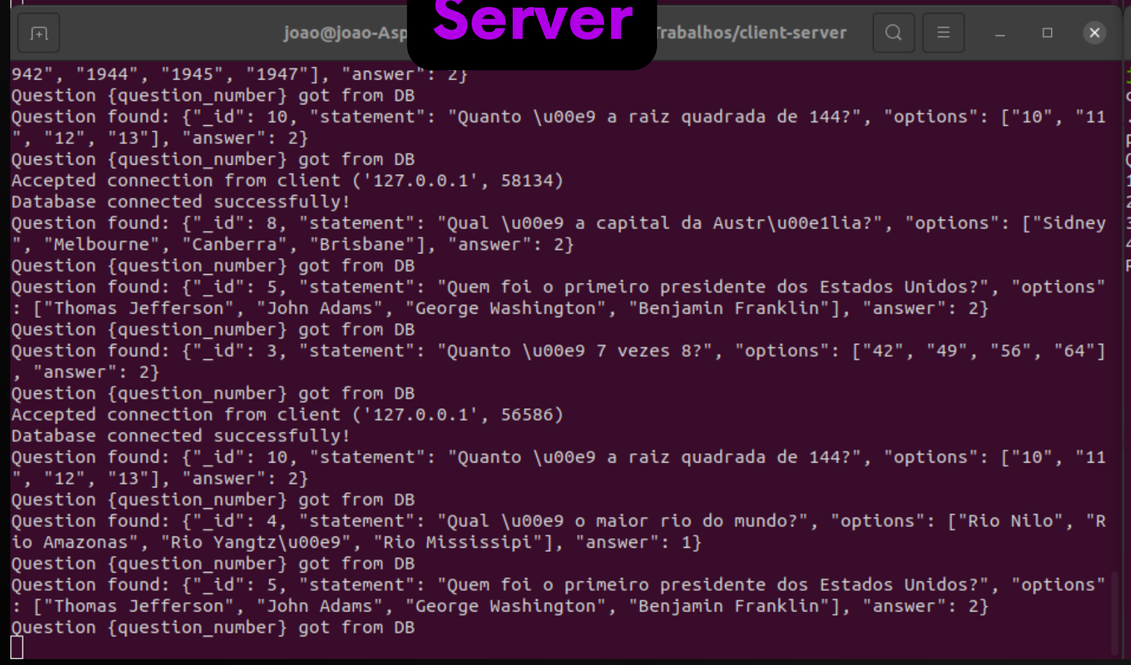

```
[
    {
      "_id": 1,
      "statement": "Qual é a capital da França?",
      "options": ["Berlim", "Madrid", "Paris", "Londres"],
      "answer": 2
    },
    {
      "_id": 2,
      "statement": "Em que ano a Segunda Guerra Mundial terminou?",
      "options": ["1942", "1944", "1945", "1947"],
      "answer": 2
    },
    {
      "_id": 3,
      "statement": "Quanto é 7 vezes 8?",
      "options": ["42", "49", "56", "64"],
      "answer": 2
    },
    {
      "_id": 4,
      "statement": "Qual é o maior rio do mundo?",
      "options": ["Rio Nilo", "Rio Amazonas", "Rio Yangtzé", "Rio Mississipi"],
      "answer": 1
    },
    {
      "_id": 5,
      "statement": "Quem foi o primeiro presidente dos Estados Unidos?",
      "options": ["Thomas Jefferson", "John Adams", "George Washington", "Benjamin Franklin"],
      "answer": 2
    },
```


2

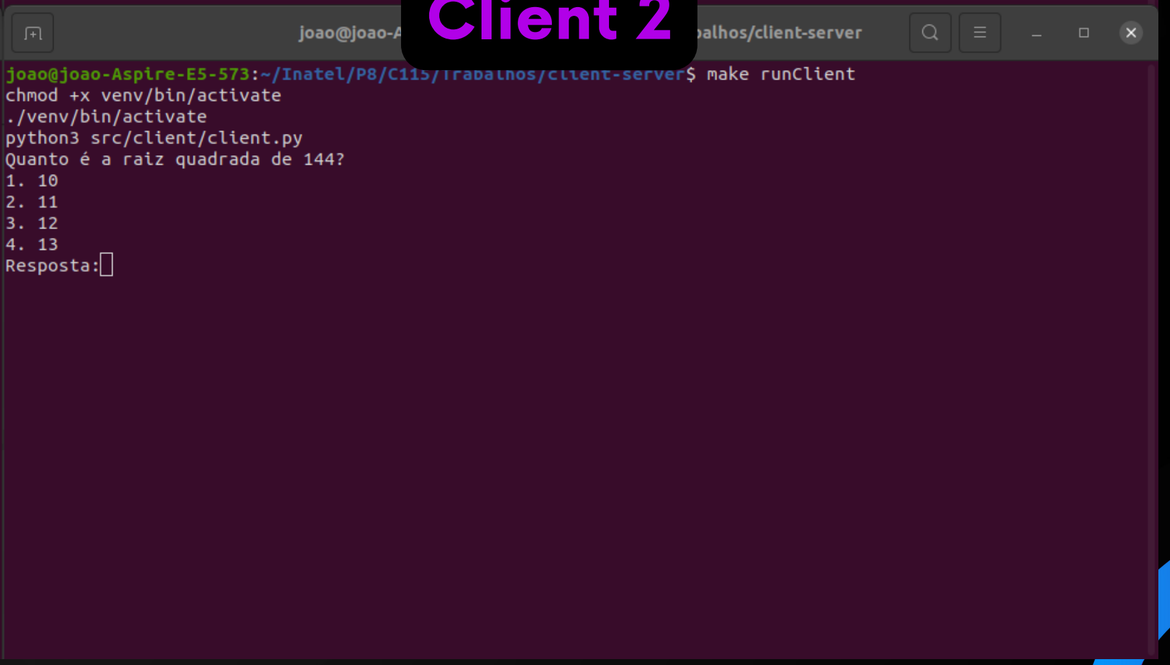

```
    {
      "_id": 6,
      "statement": "Qual é o país mais populoso do mundo?",
      "options": ["Índia", "Estados Unidos", "China", "Brasil"],
      "answer": 2
    },
    {
      "_id": 7,
      "statement": "Quem escreveu a peça de teatro 'Romeu e Julieta'?",
      "options": ["William Shakespeare", "Leo Tolstoy", "Charles Dickens", "Jane Austen"],
      "answer": 0
    },
    {
      "_id": 8,
      "statement": "Qual é a capital da Austrália?",
      "options": ["Sidney", "Melbourne", "Canberra", "Brisbane"],
      "answer": 2
    },
    {
      "_id": 9,
      "statement": "Qual é o maior deserto do mundo?",
      "options": ["Deserto de Gobi", "Deserto do Saara", "Deserto de Atacama", "Deserto da Arábia"],
      "answer": 1
    },
    {
      "_id": 10,
      "statement": "Quanto é a raiz quadrada de 144?",
      "options": ["10", "11", "12", "13"],
      "answer": 2
    }
]
```

Testes


Containers Docker
:"x86_64","version"
mongodb | {"t":{"$date":"2023-09-04T16:45:53.215+00:00"},"s":"I", "c":"NETWORK", "id"
:51800, "ctx":"conn42","msg":"client metadata","attr":{"remote":"172.18.0.1:59288","client":"conn42",
"doc":{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture"
:"x86_64","version":"5.15.0-82-generic"},"platform":"CPython 3.8.10.final.0"}}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.215+00:00"},"s":"I", "c":"ACCESS", "id"
:6788604, "ctx":"conn42","msg":"Auth metrics report","attr":{"metric":"acquireUser","micros":0}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.228+00:00"},"s":"I", "c":"ACCESS", "id"
:5286306, "ctx":"conn42","msg":"Successfully authenticated","attr":{"client":"172.18.0.1:59288","isSpec
ulative":true,"isClusterMember":false,"mechanism":"SCRAM-SHA-256","user":"root","db":"admin","result":0
,"metrics":{"conversation_duration":{"micros":13508,"summary":{"0":{"step":1,"step_total":2,"duration_m
icros":85},"1":{"step":2,"step_total":2,"duration_micros":36}}}},"extraInfo":{}}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.229+00:00"},"s":"I", "c":"NETWORK", "id"
:6788700, "ctx":"conn42","msg":"Received first command on ingress connection since session start or aut
h handshake","attr":{"elapsedMillis":0}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.235+00:00"},"s":"I", "c":"NETWORK", "id"
:22944, "ctx":"conn42","msg":"Connection ended","attr":{"remote":"172.18.0.1:59288","uuid":{"uuid":{"
$uuid":"d3bfeb72-b46d-4e72-ab04-e9da32294192"}},"connectionId":42,"connectionCount":6}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.714+00:00"},"s":"I", "c":"-", "id"
:20883, "ctx":"conn41","msg":"Interrupted operation as its client disconnected","attr":{"opId":7942}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.715+00:00"},"s":"I", "c":"NETWORK", "id"
:22944, "ctx":"conn43","msg":"Connection ended","attr":{"remote":"172.18.0.1:59286","uuid":{"uuid":{"
$uuid":"60f593c3-7d4c-486f-8b34-387748ecd6d6"}},"connectionId":43,"connectionCount":5}}
mongodb | {"t":{"$date":"2023-09-04T16:45:53.716+00:00"},"s":"I", "c":"NETWORK", "id"
:22944, "ctx":"conn41","msg":"Connection ended","attr":{"remote":"172.18.0.1:59276","uuid":{"uuid":{"
$uuid":"8fd089ed-558d-4b0a-9c8d-581c15
":41,"connectionCount":4}}


Server
942", "1944", "1945", "1947"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 10, "statement": "Quanto \u00e9 a raiz quadrada de 144?", "options": ["10", "11
", "12", "13"], "answer": 2}
Question {question_number} got from DB
Accepted connection from client ('127.0.0.1', 58134)
Database connected successfully!
Question found: {"_id": 8, "statement": "Qual \u00e9 a capital da Austr\u00e1lia?", "options": ["Sidney
", "Melbourne", "Canberra", "Brisbane"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 5, "statement": "Quem foi o primeiro presidente dos Estados Unidos?", "options"
: ["Thomas Jefferson", "John Adams", "George Washington", "Benjamin Franklin"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 3, "statement": "Quanto \u00e9 7 vezes 8?", "options": ["42", "49", "56", "64"]
, "answer": 2}
Question {question_number} got from DB
Accepted connection from client ('127.0.0.1', 56586)
Database connected successfully!
Question found: {"_id": 10, "statement": "Quanto \u00e9 a raiz quadrada de 144?", "options": ["10", "11
", "12", "13"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 4, "statement": "Qual \u00e9 o maior rio do mundo?", "options": ["Rio Nilo", "R
io Amazonas", "Rio Yangtz\u00e9", "Rio Mississipi"], "answer": 1}
Question {question_number} got from DB
Question found: {"_id": 5, "statement": "Quem foi o primeiro presidente dos Estados Unidos?", "options"
: ["Thomas Jefferson", "John Adams", "George Washington", "Benjamin Franklin"], "answer": 2}
Question {question_number} got from DB


Client 1
joao@joao-Aspire-E5-573:~/Inatel/P8/C11
$ make runClient
chmod +x venv/bin/activate
./venv/bin/activate
python3 src/client/client.py
Qual é a capital da Austrália?
1. Sidney
2. Melbourne
3. Canberra
4. Brisbane
Resposta:


Client 2
joao@joao-Aspire-E5-573:~/Inatel/P8/C11
$ make runClient
chmod +x venv/bin/activate
./venv/bin/activate
python3 src/client/client.py
Quanto é a raiz quadrada de 144?
1. 10
2. 11
3. 12
4. 13
Resposta:

Containers Docker

```
:"x86_64","version"
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.215+00:00"},"s":"I",  "c":"NETWORK",  "id"
:51800,   "ctx":"conn42","msg":"client metadata","attr":{"remote":"172.18.0.1:59288","client":"conn42",
"doc":{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture"
:"x86_64","version":"5.15.0-82-generic"},"platform":"CPython 3.8.10.final.0"}}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.215+00:00"},"s":"I",  "c":"ACCESS",   "id"
:6788604, "ctx":"conn42","msg":"Auth metrics report","attr":{"metric":"acquireUser","micros":0}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.228+00:00"},"s":"I",  "c":"ACCESS",   "id"
:5286306, "ctx":"conn42","msg":"Successfully authenticated","attr":{"client":"172.18.0.1:59288","isSpec
ulative":true,"isClusterMember":false,"mechanism":"SCRAM-SHA-256","user":"root","db":"admin","result":0
,"metrics":{"conversation_duration":{"micros":13508,"summary":{"0":{"step":1,"step_total":2,"duration_m
icros":85},"1":{"step":2,"step_total":2,"duration_micros":36}}}},"extraInfo":{}}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.229+00:00"},"s":"I",  "c":"NETWORK",  "id"
:6788700, "ctx":"conn42","msg":"Received first command on ingress connection since session start or aut
h handshake","attr":{"elapsedMillis":0}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.235+00:00"},"s":"I",  "c":"NETWORK",  "id"
:22944,   "ctx":"conn42","msg":"Connection ended","attr":{"remote":"172.18.0.1:59288","uuid":{"uuid":{"
$uuid":"d3bfeb72-b46d-4e72-ab04-e9da32294192"}},"connectionId":42,"connectionCount":6}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.714+00:00"},"s":"I",  "c":"-",        "id"
:20883,   "ctx":"conn41","msg":"Interrupted operation as its client disconnected","attr":{"opId":7942}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.715+00:00"},"s":"I",  "c":"NETWORK",  "id"
:22944,   "ctx":"conn43","msg":"Connection ended","attr":{"remote":"172.18.0.1:59286","uuid":{"uuid":{"
$uuid":"60f593c3-7d4c-486f-8b34-387748ecd6d6"}},"connectionId":43,"connectionCount":5}}
mongodb                 | {"t":{"$date":"2023-09-04T16:45:53.716+00:00"},"s":"I",  "c":"NETWORK",  "id"
:22944,   "ctx":"conn41","msg":"Connection ended","attr":{"remote":"172.18.0.1:59276","uuid":{"uuid":{"
$uuid":"8fd089ed-558d-4b0a-9c8d-581c15                                                    d":41,"connectionCount":4}}
```

Client 1

```
./venv/bin/activate
python3 src/client/client.py
Qual é a capital da Austrália?
1. Sidney
2. Melbourne
3. Canberra
4. Brisbane
Resposta:1
Quem foi o primeiro presidente dos Estados Unidos?
1. Thomas Jefferson
2. John Adams
3. George Washington
4. Benjamin Franklin
Resposta:1
Quanto é 7 vezes 8?
1. 42
2. 49
3. 56
4. 64
Resposta:3
Você errou a questão 1 a resposta correta é 3. Canberra

Você errou a questão 2 a resposta correta é 3. George Washington

Você acertou a questão 3
```

Feedback para o Client 1

Server

```
942", "1944", "1945", "1947"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 10, "statement": "Quanto \u00e9 a raiz quadrada de 144?", "options": ["10", "11
", "12", "13"], "answer": 2}
Question {question_number} got from DB
Accepted connection from client ('127.0.0.1', 58134)
Database connected successfully!
Question found: {"_id": 8, "statement": "Qual \u00e9 a capital da Austr\u00e1lia?", "options": ["Sidney
", "Melbourne", "Canberra", "Brisbane"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 5, "statement": "Quem foi o primeiro presidente dos Estados Unidos?", "options"
: ["Thomas Jefferson", "John Adams", "George Washington", "Benjamin Franklin"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 3, "statement": "Quanto \u00e9 7 vezes 8?", "options": ["42", "49", "56", "64"]
, "answer": 2}
Question {question_number} got from DB
Accepted connection from client ('127.0.0.1', 56586)
Database connected successfully!
Question found: {"_id": 10, "statement": "Quanto \u00e9 a raiz quadrada de 144?", "options": ["10", "11
", "12", "13"], "answer": 2}
Question {question_number} got from DB
Question found: {"_id": 4, "statement": "Qual \u00e9 o maior rio do mundo?", "options": ["Rio Nilo", "R
io Amazonas", "Rio Yangtz\u00e9", "Rio Mississipi"], "answer": 1}
Question {question_number} got from DB
Question found: {"_id": 5, "statement": "Quem foi o primeiro presidente dos Estados Unidos?", "options"
: ["Thomas Jefferson", "John Adams", "George Washington", "Benjamin Franklin"], "answer": 2}
Question {question_number} got from DB
```

Client 2

```
chmod +x venv/bin/activate
./venv/bin/activate
python3 src/client/client.py
Quanto é a raiz quadrada de 144?
1. 10
2. 11
3. 12
4. 13
Resposta:3
Qual é o maior rio do mundo?
1. Rio Nilo
2. Rio Amazonas
3. Rio Yangtzé
4. Rio Mississipi
Resposta:2
Quem foi o primeiro presidente dos Estados Unidos?
1. Thomas Jefferson
2. John Adams
3. George Washington
4. Benjamin Franklin
Resposta:3
Você acertou a questão 1

Você acertou a questão 2

Você acertou a questão 3

joao@joao-Aspire-E5-573:~/Inatel/P8/C115/Trabalhos/client-server$
```

Feedback para o Client 2

Obrigado!