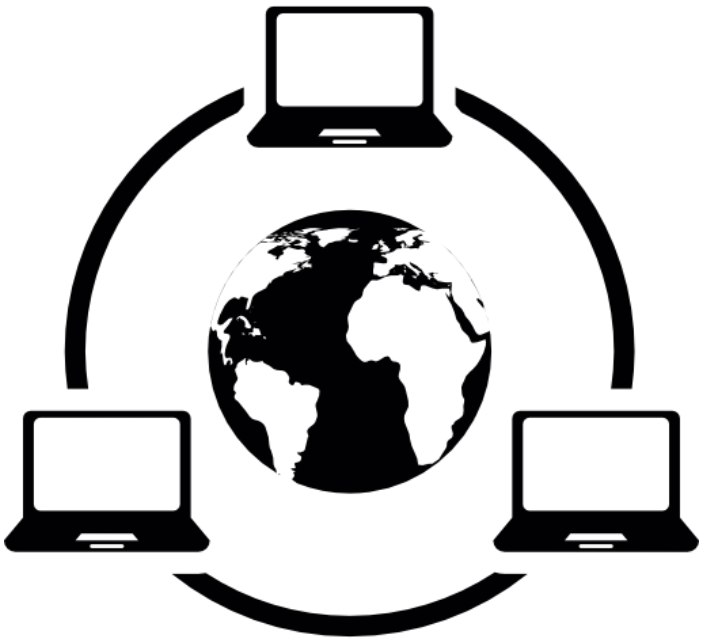
Relatório trabalho prático

Redes de Computadores



Trabalho realizado por:
Ruben Farinha nº48329

Docentes:
Professor Pedro Patinho
Professor Vasco Pedro

1. Introdução

Este relatório apresenta a proposta de implementação de um sistema de gestão de aulas com base num servidor TCP/IP. O objetivo principal deste sistema é permitir a gestão das presenças dos alunos durante as aulas, bem como facilitar a interação através de perguntas e respostas e a partilha de arquivos relevantes para a aula.

O sistema será baseado num servidor TCP que aceita conexões de múltiplos clientes simultaneamente, utilizando a porta 5555 como ponto de acesso. O cliente TCP será desenvolvido para fornecer uma interface amigável aos usuários, permitindo o acesso e utilização de todas as funcionalidades do servidor de forma intuitiva, seja por meio de comandos simples de um menu de opções.

A comunicação entre o cliente e o servidor será realizada utilizando um protocolo de pedido-resposta, com mensagens limitadas a 1024 bytes cada.

O sistema será composto por três principais componentes: gestão de presenças, gestão de perguntas e respostas, e gestão de arquivos. Cada uma dessas componentes possui funcionalidades específicas que serão detalhadas ao longo deste relatório. Além disso, será implementado um mecanismo de persistência de dados, que permitirá a recuperação das informações armazenadas em caso de reinicialização do servidor. Os dados a serem salvos incluem as perguntas, respostas, o registo de presenças e os ficheiros. O sistema de persistência de dados tem um período de intervalo de 5 minutos.

O sistema foi desenvolvido utilizando a linguagem de programação Python devido à facilidade e familiaridade com a mesma, o que proporcionou uma maior agilidade e eficiência na execução do projeto. Além disso, foram incorporadas algumas funcionalidades opcionais, que serão explicadas ao longo deste relatório.

2. Funcionamento e Comandos Disponíveis

No momento em que um cliente estabelece conexão com o servidor, este é recebido e aceite, permitindo a interação entre ambas as partes. No chat, são exibidos todos os comandos disponíveis, juntamente com as instruções sobre como cada comando deve ser escrito pelo usuário. Essa adição foi feita com o objetivo de facilitar o processo de comunicação com o servidor e o entendimento do funcionamento do chat.
A seguir, são apresentados os comandos disponíveis no chat:

1. "IAM": Utilizado para efetuar o login no sistema. O servidor responderá com "HELLO" após o comando ser enviado. Também foi implementada a opção de autenticação com password. Se a senha não existir para o respetivo "id_aluno", a primeira password inserida será registada.
   Exemplo:
   Cliente: IAM l12345 password123
   Servidor: HELLO l12345

2. "ASK": Permite que os alunos façam perguntas durante a aula. As perguntas podem ser respondidas pelos colegas ou pelo professor.
   Exemplo:
   Cliente: ASK Quem foi o primeiro Rei de Portugal?
   Servidor: Question_ID: 1
   Cliente: ASK Quanto é 6 a dividir por 2?
   Servidor: Question_ID: 2

3. "ANSWER": Utilizado para fornecer respostas às perguntas feitas. Se utilizada para uma pergunta já respondida, é adicionada também a nova resposta.
   Exemplo:
   Cliente: ANSWER 1 D. Afonso Henriques
   Servidor: Resposta registada para a pergunta 1
   Cliente: ANSWER 1 D. Afonso Henriques “O Conquistador” 1143-1185
   Servidor: Resposta registada para a pergunta 1

4. "LISTQUESTIONS": Exibe a lista de perguntas feitas pelos alunos e suas respetivas respostas.
   Exemplo:
   Cliente: LISTQUESTIONS
   Servidor:LISTQUESTIONS
   (1) Quem foi o primeiro Rei de Portugal?
   D. Afonso Henriques
   D. Afonso Henriques “O Conquistador” 1143-1185
   (2) Quanto é 6 a dividir por 2?
   (NOTANSWERED)
   ENDQUESTIONS

5. "PUTFILE": Permite fazer o upload de arquivos relevantes para a aula. Este comando está implementado, porém não está a funcionar corretamente.

6. "LISTFILES": Exibe a lista de arquivos disponíveis para download. Comando parece estar a funcionar corretamente apesar de não ter sido testado corretamente devido a falta de funcionalidade do comando PUTFILE.

7. "GETFILE": Permite fazer o download de um arquivo específico, indicado pelo número. Este comando está implementado, porém pode não estar a funcionar corretamente e impossível de testar devido a falta de funcionalidade do comando PUTFILE.

8. "EXIT": Desconecta o cliente do servidor.

3. Decisões tomadas ao longo do desenvolvimento do trabalho

Neste trabalho, foi tomada a decisão de implementar o projeto utilizando a linguagem de programação Python. Essa escolha foi feita levando em consideração a facilidade e familiaridade com a mesma, o que resultou em maior agilidade e eficiência na execução do projeto.

Para o armazenamento dos dados, recorreu-se a dicionários, que proporcionam uma estrutura simples e flexível para organizar as informações do sistema. Foram utilizados quatro dicionários principais: "presencas" para armazenar os dados de presença, "questions" para armazenar as questões e respostas, "files" para armazenar os arquivos, e "passwords" para armazenar os dados de autenticação dos clientes.

Uma das preocupações do projeto foi garantir a persistência dos dados. Para isso, implementou-se um mecanismo de persistência utilizando um intervalo pré-definido de 5 minutos. Nesse intervalo, os dados são armazenados num arquivo chamado "dados.pickle", dessa forma, mesmo que o servidor seja reiniciado, é possível carregar os dados do arquivo para restaurar o estado anterior do sistema.

No que diz respeito à comunicação com os clientes, recorreu-se a threads para lidar com múltiplos clientes de forma simultânea. Cada conexão de cliente é tratada em uma thread separada, permitindo uma comunicação assíncrona e evitando que uma conexão bloqueie o atendimento de outros clientes. Além disso, utilizamos uma função “mutex” para garantir que os dados sejam acessados de forma exclusiva e evitar sobreposições ou erros decorrentes do acesso simultâneo por diferentes clientes.

O uso da linguagem Python, juntamente com os dicionários e as threads, proporcionou uma solução ágil e flexível para atender aos requisitos do projeto.

4. Descrição do código e funcionamento

Este relatório descreve o funcionamento do sistema cliente-servidor implementado utilizando sockets TCP. O sistema consiste num servidor que aguarda a conexão de clientes e um cliente que se conecta ao servidor para interagir com ele por meio de comandos.

1. Inicialização do Servidor: Para iniciar o servidor, é necessário executar o comando "python srv.py" no terminal. Após a execução, o servidor ficará em modo de escuta, esperando a conexão de clientes.
2. Inicialização do Cliente: Após o servidor estar em execução, pode-se iniciar o cliente utilizando o comando "python cli.py" no terminal. O cliente se conectará ao servidor por meio de sockets TCP.
3. Interação com o Cliente: Após a conexão ser estabelecida, o cliente receberá uma mensagem de boas-vindas com a lista de comandos disponíveis para interagir com o servidor. Os comandos disponíveis são os seguintes:

```
# Mensagem de boas-vindas com a lista de comandos
welcome_message = """
Bem-vindo ao Servidor!

- IAM <aluno_id> <password>:
  Comando para identificação do aluno e registo de presença.
  Utilize junto com a sua password para autenticação.

- ASK <pergunta>:
  Comando para fazer uma pergunta ao servidor.

- ANSWER <question_id> <resposta>:
  Comando para adicionar uma resposta a uma pergunta específica.

- LISTQUESTIONS:
  Comando para listar todas as perguntas feitas.

- PUTFILE:
  Comando para enviar/upload de um arquivo para o servidor.

- LISTFILES:
  Comando para mostrar os arquivos disponíveis no servidor.

- GETFILE <nome_do_arquivo>:
  Comando para download/receber um arquivo do servidor.

- EXIT:
  Desconectar do servidor

O servidor realiza a persistência de dados a cada 5 minutos. Guardando os dados num ficheiro .pickle

Digite um comando para interagir com o servidor.
"""
```

4. Para ter acesso aos vários comandos, é necessário executar o comando IAM pelo cliente, utilizando uma identificação e senha para se registar. Caso a identificação não tenha sido registada anteriormente, será realizada a criação de um novo registo. Uma vez registado, o cliente poderá autenticar-se, registando a sua presença com base no tempo de início da aula/servidor e obter acesso aos comandos.
   O comando ASK possibilita adicionar uma pergunta que pode ser respondida por qualquer outro usuário. Já o comando ANSWER permite responder às perguntas, com a funcionalidade de permitir múltiplas respostas, possibilitando a correção de respostas anteriores caso necessário. O comando LISTQUESTIONS lista todas as perguntas, incluindo aquelas que ainda não foram respondidas.
   O comando PUTFILE tinha como objetivo permitir o envio/upload de arquivos para o servidor. No entanto, após tentativas de implementação tanto no cliente quanto no servidor, essa funcionalidade não está funcionando corretamente. O mesmo problema ocorre com o comando GETFILE, que não apresenta o efeito desejado após tentativas de implementação

tanto no cliente quanto no servidor. Esses comandos exigiriam a funcionalidade de receber e enviar arquivos pelo cliente. Por outro lado, o comando LISTFILES lista todos os arquivos disponíveis para download no servidor e parece estar a funcionar corretamente, embora não seja possível confirmar a sua total funcionalidade devido ao problema com o comando PUTFILE.

Por fim, o comando EXIT permite que o cliente se desconecte do servidor a qualquer momento, encerrando a conexão de forma adequada.

5. Conclusão

Neste trabalho, além dos comandos solicitados, foram adicionadas três extensões importantes ao sistema cliente-servidor. A primeira extensão consiste na utilização de uma senha para autenticar e registar a presença do aluno no sistema. A segunda extensão implementada permite adicionar múltiplas respostas às perguntas realizadas no sistema, oferecendo maior flexibilidade e interação. Por fim, foi implementada a funcionalidade de também armazenar os dados das presenças anteriores, permitindo um histórico de registos.

No geral, todas as funcionalidades e componentes foram implementados com relativa facilidade, com exceção dos comandos relacionados à gestão de arquivos. Após diversas tentativas de implementação, tanto no servidor atuando como recetor/envio quanto no cliente também como possível recetor/envio de arquivos, os comandos PUTFILE e GETFILE não funcionaram conforme o esperado. No entanto, o comando LISTFILES aparenta estar a funcionar corretamente, embora não tenha sido testado na sua totalidade.

Apesar dos desafios encontrados na implementação dos comandos de gestão de arquivos, o trabalho como um todo foi bem-sucedido, cumprindo os requisitos solicitados e proporcionando extensões valiosas para aprimorar a funcionalidade do sistema.