**UNIVERSIDADE DO MINHO**

DIOGO FONTES OLIVEIRA SILVA SANTOS, A100714

LUÍS DE SÁ RIBEIRO, A100608

MARTIM QUINTELAS PINTO FÉLIX, A100647

VICENTE COSTA MARTINS, A100713

**Sistemas Distribuídos**

Trabalho Prático

BRAGA

2023

Índice

[Introdução 3](#_Toc154678766)

[Arquitetura do Sistema 3](#_Toc154678767)

[Cliente 3](#_Toc154678768)

[Servidor 3](#_Toc154678769)

[Conexão Cliente-Servidor 4](#_Toc154678770)

[Funcionalidades 4](#_Toc154678771)

[Registo e Login 4](#_Toc154678772)

[Status 4](#_Toc154678773)

[Execução 4](#_Toc154678774)

# Introdução

O serviço cloud computing desempenha um papel fundamental na atual era da tecnologia, oferecendo soluções flexíveis e escaláveis para uma variedade de aplicações. Neste contexto, nosso projeto visa a implementação de um serviço de computação em nuvem com funcionalidade Function-as-a-Service (FaaS). O objetivo principal é permitir que os utilizadores executem tarefas computacionais remotamente, otimizando o uso de recursos disponíveis, com ênfase na gestão eficiente de memória.

O serviço utiliza uma abordagem baseada em fila de espera para a execução de tarefas, garantindo que a utilização de recursos seja otimizada, enquanto assegura que as tarefas em execução não ultrapassem os limites de memória disponível. Cada utilizador interage com o sistema por meio de uma interface fornecida por uma biblioteca cliente, possibilitando o envio de códigos de tarefas para execução e a recuperação dos resultados correspondentes.

# Arquitetura do Sistema

O sistema em questão é uma aplicação de servidor que gerência a execução de tarefas concorrentes em um ambiente distribuído.

O código está organizado em pacotes distintos: ‘**Server**’, ‘**Client**’, ‘**Shared**’, ‘**ThreaadTools**’ e ‘**Protocol**’. Cada pacote desempenha um papel fundamental na arquitetura geral do sistema.

* O pacote ‘Server’ contém as classes relacionadas à lógica do servidor, como gerenciamento de tarefas, autenticação de usuários e a lógica de execução das tarefas.
* O pacote ‘Client’ engloba as classes responsáveis pela interação entre o cliente e o servidor, incluindo a submissão de tarefas e a manipulação de respostas recebidas.
* O pacote ‘Shared’ contém classes compartilhadas entre o cliente e o servidor.
* O pacote ‘ThreadTools’ fornece ferramentas para o gerenciamento de threads, incluindo a classe ‘ThreadControl’ que controla o estado de execução do sistema.
* O pacote ‘Protocol’ define as estruturas e formatos de mensagens utilizados na comunicação entre cliente e servidor.

## Cliente

O componente cliente é responsável por interagir com o servidor, enviar solicitações e receber resultados. Utiliza sockets TCP para estabelecer uma conexão bidirecional com o servidor.

A comunicação entre cliente e servidor é baseada em mensagens definidas pelo protocolo. O cliente envia solicitações codificadas conforme o protocolo definido para o sistema. Essas solicitações podem incluir pedidos de execução de tarefas, verificação de status do sistema, autenticação, entre outros.

O cliente utiliza streams de entrada e saída para enviar e receber mensagens do servidor. O envio de solicitações e a receção de respostas são realizados por meio de objetos DataOutputStream e DataInputStream, respectivamente.

## Servidor

O componente servidor é responsável por aceitar conexões de clientes, interpretar solicitações, executar tarefas e enviar resultados de volta aos clientes. O servidor utiliza a classe **ServerSocket** do Java para aguardar e aceitar conexões de clientes. O **ServerSocke**t é responsável por criar novos sockets para comunicação com cada cliente. O servidor implementa um loop contínuo para aguardar novas conexões. Cada nova conexão é atribuída a uma instância da classe Handler, que gerencia as interações com o cliente correspondente. O servidor implementa um loop contínuo para aguardar novas conexões. Cada nova conexão é atribuída a uma instância da classe **Handler**, que gerencia as interações com o cliente correspondente.

## Conexão Cliente-Servidor

A comunicação entre cliente e servidor é essencialmente baseada na troca de mensagens.

Ambos os lados, cliente e servidor, seguem um protocolo personalizado que define como as mensagens são formatadas, enviadas e interpretadas. Este protocolo estabelece a semântica da comunicação entre as partes.

O cliente envia mensagens codificadas de acordo com o protocolo para o servidor por meio do socket TCP. Essas mensagens podem conter solicitações de execução de tarefas, autenticação, entre outras.

O servidor, ao receber uma mensagem, utiliza a lógica definida pelo protocolo para interpretar a solicitação do cliente. Após processar a solicitação, o servidor envia uma resposta de volta ao cliente, contendo, por exemplo, resultados de tarefas executadas.

# Funcionalidades

## Registo e Login

## Status

## Execução