utad UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO

MESTRADO EM
ENGENHARIA INFORMÁTICA E
TECNOLOGIA WEB

Euromil Register

Relatório

Aluno: André Costa (Nº 2302571)

Unidade Curricular: Integração de Sistemas

Docentes: Arsénio Reis e Ricardo Baptista

Dezembro de 2024

Índice

1. Introdução

O sistema Euromil *Register* foi desenvolvido com o objetivo de registar apostas online no jogo Euromilhões, utilizando as tecnologias *Spring Boot* e gRPC. Esta aplicação foi concebida para permitir a comunicação eficiente entre o cliente e o servidor, processando e validando apostas do Euromilhões, assegurando a integração e o registo das apostas de forma rápida e fiável.

2. Arquitetura da Aplicação

A aplicação é composta por dois componentes principais, o servidor gRPC e o cliente que interage com este servidor. Para a implementação do servidor, utilizámos a *framework Spring Boot*, que proporciona uma base sólida e de fácil configuração para a criação de aplicações em *Java*. O serviço gRPC permite uma comunicação remota de alta performance, utilizando o protocolo *Buffers* (*Protobuf*) para a definição das mensagens e métodos.

3. Tecnologias Utilizadas

Estas são as tecnologias principais utilizadas na construção da aplicação:

- ***Spring Boot*:** Utilizado para configurar e iniciar a aplicação do lado do servidor. Esta *framework* facilita a criação de serviços *web* e integrações com outras tecnologias;

- **gRPC:** Uma *framework* para comunicação remota, baseada no protocolo HTTP/2, que oferece uma forma eficiente de comunicação entre serviços. Utilizou-se o gRPC-*Spring-Boot-Starter* para a integração fácil entre *Spring Boot* e gRPC;

- ***Protocol Buffers*:** Utilizado para definir os dados da comunicação entre o cliente e o servidor. O ficheiro euromil.*proto* define as mensagens *RegisterRequest* e *RegisterReply*, e o serviço Euromil, que expõe o método *Register*EuroMil;

- ***Maven*:** Sistema de *build* que foi usado para gerir as dependências do projeto e gerar o código compilado da aplicação.

4. Estrutura do Código

A estrutura do código é organizada em vários componentes, que interagem entre si para alcançar os objetivos da aplicação:

- **Euromil*Proto* (*Protobuf*):** O ficheiro euromil.*proto* define o contrato de comunicação entre o cliente e o servidor, especificando o serviço e as mensagens necessárias. No serviço Euromil, define-se o método *Register*EuroMil, que é responsável por processar as apostas;

- **Servidor gRPC (Euromil*Service*):** A classe Euromil*Service*Impl, que estende a classe base EuromilGrpc.EuromilImplBase, implementa o método *register*EuroMil. Este método recebe um pedido de registo de aposta, processa os dados (*key* e *checkid*) e envia uma resposta ao cliente com uma mensagem de sucesso;

- **Cliente gRPC (Euromil*Client*):** A classe Euromil*Client* faz uma ligação ao servidor gRPC através de um *ManagedChannel*. Ela cria um pedido *RegisterRequest* contendo os dados necessários, enviando-o ao servidor e recebendo a resposta para exibição;

- **Aplicação *Spring Boot*:** A classe Euromil*RegisterApplication* é responsável por iniciar o servidor *Spring Boot* e configurar a aplicação, enquanto o serviço gRPC é integrado para receber e processar as requisições.

5. Processos de Desenvolvimento

O desenvolvimento seguiu uma abordagem modular, onde cada componente teve uma responsabilidade clara e delimitada:

- **Definição da Estrutura de Comunicação (*Protobuf*):** Primeiro, definiu-se o contrato entre o cliente e o servidor, através do ficheiro euromil.*proto*. Este definiu o método *Register*EuroMil, bem como as mensagens de entrada (*RegisterRequest*) e saída (*RegisterReply*);

- **Implementação do Serviço gRPC:** Com base na definição do ficheiro *.proto*, implementou-se a classe Euromil*Service*Impl. A comunicação é feita de forma assíncrona através de *StreamObserver*, que permite enviar e receber as respostas entre cliente e servidor;

- **Desenvolvimento do Cliente:** O cliente foi desenvolvido para enviar pedidos ao servidor gRPC. Utilizando uma conexão *ManagedChannel* e um *stub* Euromil*BlockingStub*, o cliente envia o registo da aposta e exibe a resposta do servidor;

- **Configuração da Aplicação *Spring Boot*:** A configuração do *Spring Boot* foi feita com o intuito de configurar o servidor para escutar na porta 6565. A dependência grpc-*spring-boot-starter* permitiu integrar de forma prática o *Spring Boot* com gRPC;

- **Testes e Validação:** Após a implementação, a aplicação foi testada para verificar se a comunicação entre o cliente e o servidor estava a funcionar corretamente. Para testar o registo das apostas, utilizou-se dados de exemplo para garantir que o serviço estava a devolver respostas apropriadas e precisas.

6. Conclusão

A aplicação Euromil *Register* foi desenvolvida com o intuito de permitir o registo de apostas no jogo Euromilhões de forma rápida e eficiente. A escolha das tecnologias, como o gRPC e o *Spring Boot*, garantiu uma arquitetura escalável e de alta performance. A solução implementada permite uma comunicação entre o cliente e o servidor segura, eficiente e bem estruturada, capaz de processar grandes volumes de requisições de forma eficaz.

A aplicação apresenta uma boa modularização, com a separação clara entre o servidor e o cliente, permitindo uma futura expansão do sistema, caso se decida integrar novas funcionalidades ou modificar a estrutura do serviço.

A criação desta aplicação evidenciou as vantagens do uso do gRPC em comparação com soluções mais tradicionais de comunicação, proporcionando uma comunicação remota com baixo custo e alta performance. A utilização do *Spring Boot* como *framework* para configurar e iniciar a aplicação simplificou imenso o processo de desenvolvimento, reduzindo a quantidade de código necessário e melhorando a manutenção da aplicação.