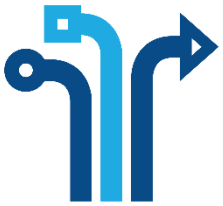

ESCOLA
SUPERIOR
DE TECNOLOGIA
IPCA

COMUNICAÇÃO DE DADOS

Emanuel de Jesus Correia Carvalho - 8820
Joel Filipe Lacerda Martins – 17439
João Nuno Gonçalves Veloso - 17729

Este trabalho prático tem como objetivo estudar o funcionamento de uma API REST. Na nossa aplicação em c#, criamos uma api rest com serviço em background para um chat.

Neste trabalho, dividimos a nossa API em 4 controladores:

- FileController
- GroupController
- MessageController
- UserController

No trabalho também dividimos a nossa aplicação por camadas onde, o controller injeta o serviço onde é implementada toda a lógica, que por si, injeta o repositório que "trata" da inserção, actualização ou remoção da informação na base de dados. Este método é bastante utilizado no atual mercado de trabalho, bem como na organização de código que permite entender e estruturar melhor o código em si.

Relativamente ao serviço, implementamos um serviço que trabalha em background, que se certifica se o utilizador logado tem mensagens por ler e no caso dos ficheiros, se expiraram a sua validade definida por defeito a cada upload de 30 segundos.

FileController

No FileController existe os seguintes métodos:

- FileUpload: Neste método do tipo POST, nós enviámos um ficheiro do tipo PNG para a API e api insere o no servidor e fazemos o seu registo na base de dados.
- GetAllFiles: Do tipo GET, que a API nos responde com o caminho dos ficheiros no servidor.
- FileDownload: Também do tipo GET, enviámos o nome do ficheiro e API responde-nos com o ficheiro que enviámos o nome, caso exista.
- FileDelete: Do tipo POST, enviámos o nome do ficheiro que desejamos apagar e caso ele exista, elimina.

GroupController

Neste controlador, tratamos tudo que tenha a ver com Grupos/Canais e os métodos são os seguintes:

- GetGroups: Do tipo GET, a API retorna todos os Grupos/Canais que existem na base de dados.

- AddGroupOrChannel: Do tipo POST, enviámos o tipo (G para Grupos e C para canais) e o seu nome e api regista na base de dados o seu registo.
- AddUserToGroup: Tipo POST, enviámos o nome do grupo e o nome do utilizador que queremos adicionar ao grupo, para cumprir este método com sucesso tem que ser obrigatoriamente, o dono do grupo.
- SubscribeToChannel: Neste método do tipo POST, enviámos o nome do canal que queremos subscrever e a API subscreve o utilizador autenticado ao canal.
- UnsubscribeToChannel: Neste método do tipo PUT, enviámos o nome do canal ao qual o utilizador autenticado pertence e API trata de nos desvincular do canal.
- RemoveUserFromGroup: Método do tipo POST, neste método enviámos para API o nome do grupo e o utilizador que queremos remover e API remove-nos do grupo. É necessário ser owner do grupo ou ser o utilizador que quer remover.
- RemoveGroupOrChannel: Neste método do tipo PUT, enviámos o nome do grupo e API remove o grupo do servidor mas não da base de dados.

MessageController

Neste controlador, lidámos com tudo o que seja mensagem na API com os seguintes métodos:

- SendMessageToUser: Tipo POST, neste método enviámos mensagem ao utilizador que queremos enviar uma mensagem e API trata do resto.
- SendMessageToGroup: Método do tipo POST, enviámos a mensagem e o nome do grupo que queremos enviar uma mensagem.
- SendMessageToChannel: Outro método do tipo POST, enviámos o nome do canal e a mensagem que desejamos e se formos donos do canal a mensagem será enviada!
- MessagesSentByUser: Método do tipo GET, que nos permite saber quais foram as mensagens que enviámos.
- MessagesReceivedByUse: Também do tipo GET, que nos permite ver as mensagens lidas que temos na caixa de mensagens.
- MessagesReceivedUnreadedByUser: Igualmente do tipo GET, que permite ver as mensagens não lidas na caixa de correio e passa-as para lidas.
- GetNumberOfMessages: Tipo GET, permite saber quantas mensagens lidas temos na caixa de mensagens.
- RemoveMessage: Tipo PUT, em que enviámos o ID da mensagem e a apagámos da nossa caixa de mensagens ou se a enviámos.

UserController

Neste controller, lidámos com todo o tipo de operações relativos a users com os seguintes métodos:

- GetUsers: Neste método do tipo GET, retornámos todos os utilizadores que existem na API.
- AddUser: Neste método do tipo POST, enviámos nome, email, username e password para a API e ela, por sua vez, regista na base de dados. Para utilizar este método é necessário ser admin da API.
- RemoveUser: Neste método do tipo PUT, enviámos o utilizador que pretendemos remover, a API apaga-o do sistema mas não da base dados como pede no enunciado, porque apagar dados de uma base dado, é definitivamente uma má prática.
- Authenticate: Do tipo POST, este método permite fazer login na API para poder aceder a todos os métodos e é o único método que os parâmetros vão pelo body.
- ChangePassword: Este método do tipo PUT, permite que o utilizador mude a sua password de acesso à API.

Conclusão

Neste trabalho-prático, aprendemos a perceber os conceitos de http e como funcionam as API's no geral, e essencialmente a sair da nossa zona de "conforto" que é a nossa matéria e procurar por nós o que necessitámos de saber, o que pra nós é muito importante quer a nível pessoal, quer mais tarde a nível profissional.

Na nossa opinião peca por ser demasiado extenso para o tempo que possuímos para o fazer, mas sem dúvida que é muito importante para o mercado de trabalho, independentemente da posição que estaremos inclinados em escolher no futuro.