Capítulo 1 : Apresentação API

Descrição

Esta API foi desenvolvida com o recurso à OpenAPI, utilizando uma camada de serviços REST.

Este trabalho tem como objetivo criar um serviço web para uma plataforma de jogos online, que está protegida com uma API de autenticação desenvolvida pelo nosso grupo.

Para esta API foram utilizados 6 controladores:

- *jogadorController*
- *partidaController*
- *utilizadorController*

> **jogadorController**

É composto por 6 funcionalidades:

- **countJogadores** -> Conta o número total de jogadores na base de dados e retorna o resultado como uma resposta JSON.
- **retrieveJogadores** -> Recupera todos os jogadores da base de dados e retorna a lista como uma resposta JSON.
- **createJogador** -> Cria um novo jogador na base de dados e retorna o jogador recém-criado como uma resposta JSON.
- **retrieveJogador** -> Recupera um jogador específico da base de dados com base no ID fornecido na requisição e retorna o jogador encontrado como uma resposta JSON. Se não for encontrado, retorna uma resposta de erro com status 404
- **updateJogador** -> Atualiza um jogador existente na base de dados com base no ID fornecido na requisição e nos novos valores fornecidos. Retorna o jogador atualizado como uma resposta JSON. Se não for encontrado, retorna uma resposta de erro com status 404.
- **deleteJogador** -> Exclui um jogador da base de dados com base no ID fornecido na requisição e retorna uma mensagem de sucesso como uma resposta JSON. Se não for encontrado, retorna uma resposta de erro com status 404.

> **partidaController**

É composto por 6 funcionalidades:

- **countPartidas** -> Conta o número total de partidas na base de dados e retorna o resultado como uma resposta JSON.
- **retrievePartidas** -> Recupera todas as partidas da base de dados e retorna a lista como uma resposta JSON.

- **createPartida** -> Cria uma nova partida na base de dados e retorna a partida recém-criada como uma resposta JSON.
- **retrievePartida** -> Recupera uma partida específica da base de dados com base no ID fornecido na requisição e retorna a partida encontrada como uma resposta JSON. Se não for encontrada, retorna uma resposta de erro com status 404.
- **updatePartida** -> Atualiza uma partida existente na base de dados com base no ID fornecido na requisição e nos novos valores fornecidos. Retorna a partida atualizada como uma resposta JSON. Se não for encontrada, retorna uma resposta de erro com status 404.
- **deletePartida** -> Exclui uma partida da base de dados com base no ID fornecido na requisição e retorna uma mensagem de sucesso como uma resposta JSON. Se não for encontrada, retorna uma resposta de erro com status 404.

> utilizadorController

É composto por 14 funcionalidades:

- **login** -> Verifica se as credenciais de login fornecidas (email e senha) correspondem a um utilizador registado na base de dados. Se as credenciais forem válidas, redireciona o utilizador para a página de dashboard. Caso contrário, retorna um erro de autenticação.
- **logout** -> Realiza o logout do utilizador, terminando a sessão e redirecionando para a página de login.
- **registo** -> Verifica se o utilizador já está registado com o email fornecido. Se não estiver, cria um novo utilizador com o nome, email e senha fornecidos. Após o registo bem-sucedido, redireciona o utilizador para a página de login.
- **protected** -> Retorna um ficheiro HTML protegido que só pode ser acedido por utilizadores autenticados.
- **dashboard** -> Retorna um ficheiro HTML correspondente à página de dashboard que só pode ser acedido por utilizadores autenticados.
- **authGitHub** -> Redireciona o utilizador para a autenticação com o GitHub.
- **authCallback** -> Redireciona o utilizador para a página principal após a autenticação bem-sucedida.
- **callback** -> Lida com o pedido de troca do código de autorização pelo token de acesso do GitHub. Verifica se o código de autorização está presente na consulta da URL e faz o pedido para o endpoint do token de acesso. Em seguida, guarda o token de acesso na sessão e redireciona para a página principal ou uma rota protegida.
- **me** -> Retorna os detalhes do utilizador autenticado. Se o utilizador estiver autenticado, retorna os detalhes do utilizador num formato JSON. Caso contrário, retorna um erro indicando que o utilizador não está autenticado.
- **gitHubMe** -> Retorna os detalhes do utilizador autenticado do GitHub, usando o token de acesso fornecido.
- **callbackController** -> Retorna a página de dashboard HTML após a autenticação bem-sucedida.
- **editarPerfil** -> Atualiza os dados do perfil do utilizador, como nome, email e senha. Verifica se a senha atual fornecida é válida antes de fazer as alterações.
- **getPerfil** -> Retorna a página de perfil HTML se o utilizador estiver autenticado. Caso contrário, redireciona para a página de login.
- **perfil** -> Retorna um ficheiro HTML correspondente à página de perfil que só pode ser acedido por utilizadores autenticados.

Grupo *23*

- Bernardo Magalhães A38819@umaia.pt
- Dario Rodrigues A038042@umaia.pt
- João Aragão A0939132@umaia.pt

< Anterior **^ Main** **Próximo >**