Teste Desenvolvedor: Backend

Olá Dev, para seguirmos com a sua candidatura temos um desafio para você, vamos lá?

Sua missão é criar uma API para cadastrar aulas e comentários seguindo os endpoints já estipulados e a documentação dos campos.

Esta API deve ser criada em NodeJS, utilizando banco de dados MongoDB e deverá também conter algum tipo de teste, podendo ser unitário e/ou de integração.

O código fonte deve ser disponibilizado no gitlab.com em um repositório privado, adicionando o email fabio@tindin.com.br como Maintainer do repositório.

Utilize nodejs com typescript, podendo ser qualquer versão estável, preferencialmente na versão 12.x.

Ao concluir, crie em seu repositório um README.md e descreva porque implementou dessa forma, se usou algum framework como base e descreve como você colocaria essa aplicação em produção (Usaria AWS? Se sim qual serviço? EC2? EKS? Lambda? etc), descreva em detalhes o máximo que conseguir explicar.

Endpoints da API:

Método	Endpoint	Descrição
POST	/users	Logar com um usuário (**)
POST	/classes	Criar uma nova aula
GET	/classes	Listar aulas cadastradas (*1)
GET	/classes/:id	Obter detalhes de uma aula pelo o id (*2)
PUT	/classes	Atualizar o cadastro de uma aula
DELETE	/classes/:id	Excluir o cadastro de uma aula
POST	/classes/comments	Cadastrar um comentário de uma aula (*3)
GET	/classes/comments	Listar todos os comentários de uma aula
DELETE	/classes/comments/:id	Excluir um comentário

(**) O login do usuário pode ser utilizado qualquer método de sua preferência, JWT, Cognito, etc. Não é necessário criar rota para cadastrar usuário, partimos do ponto que esse usuário já vai existir no banco de dados fazendo insert manual. Todas as demais rotas só podem ser acessíveis se o usuário tiver logado.

(*1) Na listagem da aula trazer um campo chamado `last_comment`, que é o último comentário que a aula recebeu e o campo `last_comment_date` que é a data que foi feito este comentário. Poder filtrar pelo campo name, description, data_init e data_end.

(*2) No detalhe de uma aula, trazer os últimos 3 comentários dela em um objeto dentro do próprio JSON, como o exemplo abaixo:

```
{
  name: 'Aula xyz',
  comments: [ array com os três últimos comentários ]
}
```

(*3) Ao cadastrar um novo comentário, atualizar o campo `total_comments` da aula que pertence a este comentário.

** Todas as listagens devem ter paginação de 50 em 50 registros.

Documentação dos campos:

Users:

Campo	Tipo	Descrição
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha do usuário

Classes:

Campo	Tipo	Descrição
name	String	Nome da aula
description	String	Descrição da aula
video	string	URL do vídeo da aula
data_init	Date	Data que a aula estará disponível, formato

data_end	Date	Data que a aula não estará mais disponível
date_created	Date	Data que a aula foi criada
date_updated	Date	Data que a aula foi alterada
total_comments	Int	Total de comentários que a aula tem

Comments:

Campo	Tipo	Descrição
id_class	ObjectId	Id da aula que recebeu o comentário
comment	String	Comentário
date_created	Date	Data que a aula foi criada