UNIVERSIDADE CESUMAR - UNICESUMAR

Curso: Engenharia de Software	**Série:** 5S	**Turma:** A	**Turno:** Noite
Professor(a): Thiago Bussola da Silva	**Horário:**		
Acadêmico (a):			**RA:**
Disciplina: Tópicos em Programação			**Data:**

Prova	Prova Prática	Atividades de estudo programadas (AEP)	Prova integrada	Nota final do bimestre
	PRA			

INSTRUÇÕES PARA REALIZAÇÃO DA PROVA:

⇒ Os dados do cabeçalho deverão ser preenchidos com letra maiúscula. E as questões deverão ser respondidas com letra legível.
⇒ É vedado, durante a prova, o porte e/ou o uso de aparelhos sonoros, fonográficos, de comunicação ou de registro eletrônico ou não, tais como: notebooks, celulares, tablets e similares.
⇒ A prova é individual e sem consulta, deverá ser respondida a caneta azul ou preta. Prova escrita a lápis não dá direito à revisão. Não é permitido o uso de corretivo.
⇒ É obrigatória a permanência do acadêmico 1 (uma) hora em sala de aula após o início da prova.
⇒ Não será permitida a entrada na sala de aula após 10 minutos do início da prova.
⇒ É obrigatória a assinatura da lista de presença impressa na qual constam RA, nome e curso.
⇒ O valor de cada questão está ao lado da mesma.
⇒ Todas as respostas devem constar no espaço destinado e autorizado pelo professor, à resposta.
⇒ Em caso de qualquer irregularidade comunicar ao Professor ou fiscal de sala.
⇒ Ao término da prova, levante o braço e aguarde o atendimento do professor ou do fiscal.

1ºbim.		2ºbim.		1ªsub.		2ªsub.		1ºsem.		2º sem.	

Orientações gerais:
Abra um terminal na pasta onde salvou o projeto base enviado para a solução da pasta e rode os comandos:

- npm install (para instalar as dependências)
- npm start (para levantar o servidor e disponibilizar a API.)

Atenção: Consuma os dados disponibilizados pela API do pokemon, presente em: **https://pokeapi.co/api/v2/pokemon?limit=151**

As dependências do projeto são apenas as necessárias para se executar o projeto com typescript, porém, está aberto fazer apenas com javascript.

Crie um novo projeto Node.js para realização da prova.

ORIENTAÇÃO PARA ENVIO:

Exportar a collection do Insomnia com as rotas utilizadas para validar os exercícios da prova

Remover a node_modules antes de zipar

NOME - RA
Zipar junto com os arquivos da prova

1 - Consumindo a API fornecida, realize as seguintes operações.

A) Trate os seguintes dados da API e crie um array de objetos, onde cada objeto representa um pokémon e para cada pokémon os seguintes dados devem ser registrados:
- Nome
- Tipo,
- Status,
- Numero da dex, (Deve ser buscado o número da versão FIRE RED)
- Altura,
- Peso
- E em moves, salve 4 moves.

B) Após criar a função que irá tratar os dados**, crie uma ROTA POST que irá realizar duas operações**
- Salvar os pokémons tratados em um arquivo .json
- Salve esse array de objetos no banco de dados utilizando a função do mongoose chamada insert many
(Essa função permite que você passe um array de objetos e cada índice do array irá se tornar um registro no banco).

Ponto Extra: Tente salvar os 4 moves da cada pokémon de maneira aleatória, basta percorrer o array de moves e pegar apenas 4 deles de maneira randômica.

2 - **Mapeie o arquivo criado na questão 1** e **crie um novo arquivo .json** que contenha **um array para cada tipo de pokémon**, e salve em cada array o pokémon que corresponde a esse tipo, ordenados pelo número da dex. (Você pode utilizar o método map ou reduce)

3 - **Crie um CRUD** para a criação de um time de pokémons. **Cada time deve ter o nome do treinador** responsável pelo time, **e um array de Objetos que contenha o nome dos 6 pokémons** desse time.

Estrutura do TeamSchema:

```
trainerName: String,
team: [{
        name: String
}]
```

a) **Crie uma rota do tipo POST**, onde você passe um objeto json que contenha o nome do treinador e um array de objetos com os nomes de 6 pokémons. Através dessa requisição, recebendo esses nomes, você deve buscar os pokémons em seu banco de dados e criar uma "tabela" chamada team, que contenha os 6 pokémons e suas informações.

b) **Crie uma rota do tipo GET**, onde seja possível visualizar todos os times cadastrados

c) **Crie uma rota do GET**, onde seja possível filtrar um time específico utilizando o nome do treinador como parâmetro.

d) **Crie uma rota do tipo PUT** onde você possa alterar as informações de um time utilizando o nome do treinador como parâmetro

e) **Crie uma rota do tipo DELETE** onde você pode remover um time utilizando o nome do treinador como parâmetro

4 **- Crie uma rota do tipo GET** que recebe como parâmetro o tipo do pokémon e retorne somente os pokémons que contém esse tipo.

5 - **Crie uma rota do tipo GET** que filtre o pokémon por seu número na dex e traga suas informações

6 - **Crie uma rota do tipo GET** que filtre o pokémon pelo nome e traga suas informações

Fatores levados em consideração: Arquitetura e organização de pastas, controllers, services, e POO.

A composição de um projeto estruturado corretamente será mais bem avaliada do que um projeto que não siga boas práticas de programação apresentadas durante o decorrer das aulas. É de sua responsabilidade identificar cada domínio da aplicação e separar cada funcionalidade de maneira correta.