UniSENAI PR	**Segunda Avaliação Parcial** **Segundo Semestre de 2024**

Curso: **Bacharelado em Engenharia de Software**		
Disciplina: **Backend development**	Turma: **B**	Período: **5/6**
Professor: **Paulo Sergio da Conceição Moreira**	Data: **29/11/2024**	Nota:
Aluno (a):		

Instruções:

- A avaliação pode ser realizada **individualmente, em duplas ou em trios**. Quaisquer plágios ou cópias de colegas, se evidenciadas, resultarão na anulação da nota desta avaliação.
- A avaliação deverá ser respondida **contendo todas as informações necessárias para a replicação da solução;**
- Lembre-se: ao enviar para avaliação, **encaminhe os códigos e os arquivos necessários (Excel, JSON, links) em conjunto com o relatório principal**. Você pode compactar tudo em um arquivo (.zip, .rar);
- A entrega da avaliação deverá ser feita até o dia **29/11/2024, às 22:10, via AVA (Moodle).**

Implemente uma aplicação em Node.js que disponibilize informações da grade curricular de um curso superior utilizando duas interfaces de consulta: ***REST* e *GraphQL***. A aplicação deve ser desenvolvida com foco em clareza, modularidade e boas práticas de codificação. Os dados da grade curricular serão carregados de um arquivo JSON e organizados de acordo com a arquitetura MVC (*Model-View-Controller*).

1. **Requisitos técnicos:**

   a) **Padrões de código:**
      - Use **funções e variáveis em Português** para promover clareza no desenvolvimento.
      - Utilize ***arrow functions*** para todas as funções, sempre que aplicável.

   b) **Arquitetura MVC:**
      - ***Model:*** Para lidar com os dados (carregados a partir de um arquivo JSON).
      - ***Controller***: Para centralizar a lógica e organizar as rotas e *resolvers*.

   c) **Tecnologias:**
      - Node.js com Express para REST.
      - Apollo Server para GraphQL.

d) REST:

- Retornar todos os períodos e suas disciplinas;
- Retornar todas as disciplinas do curso;
- Buscar uma disciplina específica pelo ID;
- Criar novas disciplinas;
- Atualizar disciplinas existentes;
- Remover disciplinas.

e) GraphQL:

- Retornar todos os períodos e suas disciplinas;
- Retornar todas as disciplinas do curso;
- Buscar uma disciplina específica pelo ID;
- Criar novas disciplinas;
- Atualizar disciplinas existentes;
- Remover disciplinas.

2. Critérios de avaliação

a. Implementação correta da arquitetura MVC.
b. Funcionamento das *queries* conforme o enunciado.
c. Uso consistente de variáveis e funções em Português.
d. Utilização de *arrow functions* na implementação.
e. Organização e clareza do código e estrutura de pastas.

3. Entregáveis:

a. Código-fonte completo organizado em arquivos para modelo, controlador e servidor;
b. Arquivo JSON contendo os dados da grade curricular;
c. Arquivo com as *queries* utilizadas;
d. Um README.md explicando como instalar dependências, rodar a aplicação e testar suas funcionalidades;
e. Breve relatório descrevendo a sua percepção com relação às diferenças de consultas para cada interface de consulta.

4. Dicas:

a. Empregue funções como *flatMap()* e *find()* para manipular os períodos e as disciplinas;
b. Lembre-se de iniciar o projeto e instalar as dependências;
c. Utilize ferramentas adequadas para testar os *endpoints* da API.
d. Como auxílio, utilize as aulas 10, 11, 12, 13 e 14, do repositório do GitHub da disciplina.