

### Ministério da Educação

# **Universidade Federal de Itajubá** – *Itajubá* COM222/XDES03 – Programação Web/Sistemas Web

### Exercícios —Node com Express

Prof.: Phyllipe Lima



#### Instruções Gerais:

- Atividade Deverá ser Entregue Individualmente.
- Entregar a solução completa em formato ZIP pela tarefa do SIGAA.
- Horário de Entrega definido na Tarefa do SIGAA
- Formato de Entrega: Coloque os arquivos referentes ao *front e back* em um diretório. Apague a pasta node\_modules em ambos "front" e "back". Compacte esses diretórios em um único diretório com formato ".zip" e faça a entrega no SIGAA.
- O front já foi fornecido.
- A estrutura do back também foi fornecida. Sua solução será implementada no arquivo "index.js".

### Objetivos: Familiarizar com Node e Express:

- Criar rotas com express
- Tratar requisições e retornar respostas
- Acessar e manipular arquivos com Node simulando base de dados.

Não possui conta?

### Tarefa Única: Sistema de Cadastro de Usuários para conhecer os cursos de computação da Unifei.

Nessa tarefa você deverá criar o "back" do sistema de cadastro, implementando as rotas necessárias. A Figura 1 contém a página de login. A Figura 2 a página de cadastro. O cadastro pode ser acessado clicando em "Cadastro"



Cadastro

### Bem vindo ao sistema de cadastro da UNIFEI



Crie uma nova conta	
Usuário	
Email	
Senha	
Criar Usuário	
Figura 2	

### **Funcionamento:**

1 - Login.

Após feito o login, o sistema irá abrir uma página que irá listar os cursos de computação como mostrado na Figura 3. O sistema já possui um usuário cadastrado. (note que se encontra no arquivo "banco-dados-usuario.json", mas a rota não está implementada.)

Email: jaum@teste.com.br

Senha: 123

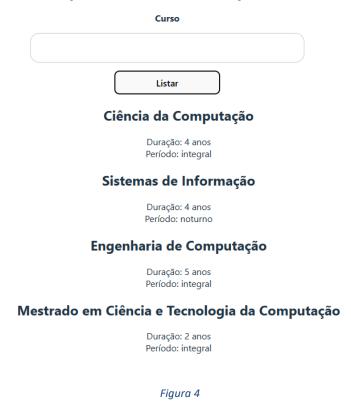
## Busque o curso pelo nome ou deixe vazio para retornar todos.

Curso	
Listar	

Figura 3

2 – Ao clicar no botão "Listar" deixando o campo "Curso" vazio, todos os cursos de computação serão retornados. Esses cursos se encontram no arquivo "cursos.json". A Figura 4 apresenta o retorno esperado (após o servidor implementado corretamente).

### Busque o curso pelo nome ou deixe vazio para retornar todos.



3 – Ao clicar no botão "Listar" colocando um curso (ou parte do nome) válido, os cursos serão retornados como mostrado na Figura 5. Observe que três cursos foram retornados.

### Busque o curso pelo nome ou deixe vazio para retornar todos.

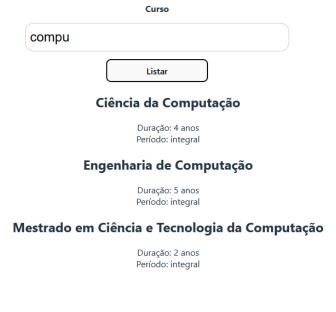


Figura 5

### **Ambiente:**

### 1 - Executando o front

Execute o comando "npm i", na pasta "front" para instalar todas as dependências necessárias. Para executar o "front", digite o comando "npm run dev". Esse comando deve ser executado na raiz do "front", onde se encontra o arquivo "package.json". Nesse momento o sistema não irá funcionar adequdamente pois não existe um servidor implementado (back).

### 2 - Executando o back

Execute o comando "npm i" na pasta "back" para instalar todas as dependências necessárias. Em seguida execute o comando "nodemon index.js". Se o comando não funcionar, instale o "nodemon" de forma global com o comando "npm i -g nodemon".

### **Requisito:**

Sua tarefa consiste em codificar o arquivo "index.js" que se encontra no diretório "back" e implementar todos os requisitos.

- 1 **Rota "/login".** Codifique a rota "/login" que deverá tratar o verbo POST. A rota deverá realizar as seguintes tarefas.
- 1.1 Acessar o arquivo "banco-dados-usuario.json" e verificar se o usuário e senha estão corretos. Se estiver incorreto, deverá retornar a mensagem "Usuario ou senhas incorretas."
- 1.2 Acessar o arquivo "banco-dados-usuario.json" e verificar se o usuário existe. Se não existir, deverá retornar a mensagem "Usuario com email \${email} não existe. Considere criar uma conta!."
- 1.3 Acessar o arquivo "banco-dados-usuario.json" e verificar se usuário existe. Se existir e a senha estiver correta, deverá retornar a mensagem "Autenticado com Sucesso"
- 1.4 Caso tenha sucesso, o front já está preparado para trocar para a página que irá listar as disciplinas. Por isso certifique que a mensagem segue o padrão especificado nos itens 1.1-1.3

- 2 **Rota "/create".** Codifique a rota "/create" que deverá tratar o verbo POST. A rota deverá realizar as seguintes tarefas.
- 1.1 Acessar o arquivo "banco-dados-usuario.json" e verificar se o email já está cadastrado. Se estiver, retornar a mensagem "Usuario com email \${email} já existe."
- 1.2 Caso o email não esteja cadastrado, o usuário será criado com sucesso e inserido no banco de dados com um "id" incremental. Em seguida deverá retornar a mensagem "Tudo certo usuario criado com sucesso."
- 1.3 Caso tenha sucesso, o front já está preparado para mostrar o botão "Fazer Login" logo abaixo do cadastro. Por isso certifique que a mensagem segue o padrão especificado nos itens 1.1-1.2. A Figura 6 apresenta esse botão. Nesse exemplo o usuário com email jaum@teste2.com.br foi criado.

#### Bem vindo ao sistema de cadastro da UNIFEI

	Crie uma nova conta
	Usuário
ja	aumzito
	Email
ja	aum@teste2.com.br
	Senha
••	•
	Criar Usuário
cer	rto usuario criado com sucesso. <u>Fazer Login</u>

Figura 6

- 3 Rota "/cursos". Codifique a rota de "/cursos" que deverá tratar o verbo GET. A rota deverá realizar as seguintes tarefas.
- 1.1 Acessar o arquivo "cursos.json" e o retornar como resposta da requisição todos os cursos. Exemplo: return res.json(cursos);
- **4 Rota "/cursos/:nome".** Codifique a rota de "/cursos/:nome" que deverá tratar o verbo GET e possui um "path parameter". O front já está preparado para enviar o parâmetro.

A rota deverá realizar as seguintes tarefas.

1.1 – Acessar o arquivo "cursos.json" e o retornar como resposta da requisição os cursos cujo parte do nome está no parâmetro enviado pelo "forms".

Para isso será necessário varrer o arquivo "cursos.json" e retornar apenas os cursos que atendem ao colocado no campo de busca.

1.2 – Caso nenhum curso seja encontrado, basta devolver a mensagem: "`curso Não Encontrado!`"