



INE5670-04238B (20201) - Desenvolvimento de Sistemas Móveis e Embarcados

Painel ► Meus cursos ► INE5670-04238B (20201) ►
Unidade III - Desenvolvimento de Aplicativos Móveis ► Trabalho 2 - React Native

Trabalho 2 - React Native

O Governo Federal disponibiliza, através do seu Portal de Transparência, uma série de dados acessíveis por meios de uma API pública do tipo REST. Utilizando esta API, é possível obter uma série de dados governamentais no formato JSON, que é facilmente consumível em aplicações React. Esta API está disponível no seguinte endereço: <http://www.transparencia.gov.br/swagger-ui.html>

Descrição do Trabalho

O trabalho consiste em criar uma aplicação para extrair dados de viagens realizadas por agentes públicos dos órgãos do Governo Federal. A aplicação deve contar com ao menos três interfaces:

A primeira interface deve permitir ao usuário pesquisar os diversos órgãos do Governo Federal, registrados no Sistema Integrado de Administração Financeira (SIAFI). Essa interface deve apresentar um campo onde o usuário pode digitar parte da descrição (nome) do órgão, um botão para realizar a busca, e uma lista que apresente os resultados da busca. Para obter esses dados, verifique a seguinte página da documentação da API: <http://www.transparencia.gov.br/swagger-ui.html#!/211rg227os/orgaosSiafiUsingGET>

Ao acessar essa URL, é possível ver seus detalhes e até invocar a API para entender como ela recebe parâmetros e retorna resultados:

GET /api-de-dados/orgaos-siafi

Consulta de órgãos cadastrados no Sistema Integrado de Administração Financeira do Governo Federal (SIAFI)

Notas de Implementação

Filtros mínimos: Página (padrão = 1);

Classe de resposta (Status 200)

OK

Modelo Example Value

```
[
  {
    "codigo": "string",
    "codigoDescricaoFormatado": "string",
    "descricao": "string"
  }
]
```

Tipo de conteúdo da resposta */*

Parâmetros

Parâmetro	Valor	Descrição	Tipo de parâmetro	Tipo de dados
codigo	<input type="text"/>	Código do Órgão (SIAFI)	query	string
descricao	<input type="text" value="santa catarina"/>	Descrição do Órgão (SIAFI)	query	string
pagina	<input type="text" value="1"/>	Página consultada	query	integer

Mensagens de resposta

Código de status HTTP	Razão	Modelo resposta	Cabeçalhos
401	Unauthorized		
403	Forbidden		
404	Not Found		

[Tente agora!](#)

Para utilizá-la, deve ser feito um *fetch* na URL abaixo, passando como parâmetros a descrição e a página (no exemplo, respectivamente, “santa catarina” e 1);

<http://www.transparencia.gov.br/api-de-dados/orgaos-siafi?descricao=santa%20catarina&pagina=1>

A passagem de parâmetros se dá pela URL da API. Os parâmetros devem ser passados no formato nome=valor, com o primeiro parâmetro iniciando com ponto de interrogação (?) separado dos demais parâmetros por &. Além disso, é necessário codificar os valores usando a função `encodeURIComponent()`, para que estejam no formato padrão de URLs, como no seguinte exemplo:

```
var descricaoUrl = encodeURIComponent(this.state.descricao);
var paginaUrl = encodeURIComponent(this.state.pagina);
var uri = "http://www.transparencia.gov.br/api-de-dados/orgaos-siafi"+
  "?descricao="+descricaoUrl+
  "&pagina="+paginaUrl;
fetch(uri)
  .then((response) => response.json())
  .then((responseJson) => {
  });
```

A segunda interface deve ser exibida ao selecionar algum órgão, e deve permitir realizar uma busca das viagens realizadas com recursos do órgão selecionado. Para isso, deve ser usada a URL abaixo:

<http://www.transparencia.gov.br/swagger-ui.html#!/Viagens32a32servi231o/viagensPorPeriodoEOrgaoUsingGET>

Esta URL recebe como argumentos o código órgão, um período (datas de ida e retorno da viagem) e o número da página. O código do órgão deve ser recebido da interface anterior. As datas devem ser informadas pelo usuário. O número da página deve ser controlado por um esquema de paginação (botões

para navegar entre as páginas, por exemplo).

Esta interface deve apresentar o total gasto em viagens pelo órgão no período em questão (somando o campo `valorTotalViagem` de cada viagem). Depois, deve mostrar uma lista das viagens realizadas no período, apresentando o nome da pessoa, o período da viagem, e o valor da viagem. O valor total das viagens deve ser atualizado conforme mais páginas são carregadas. Ou seja, ao ir para a segunda página, a aplicação deve somar as viagens carregadas e adicionar ao valor total já exibido, tomando-se o cuidado para não somar cada página mais de uma vez. Ao realizar uma nova busca, a soma deve recomeçar de acordo com os novos parâmetros.

Veja abaixo um exemplo de chamada a essa URL para as viagens da UFSC no período de 01/01/2019 a 30/01/2019:

`http://www.transparencia.gov.br/api-de-dados/viagens?`

`dataIdaDe=01%2F01%2F2019&dataIdaAte=30%2F01%2F2019&dataRetornoDe=01%2F01%2F2019&dataRetorn`

A terceira interface deve mostrar os detalhes de uma viagem. Ao selecionar uma viagem na interface anterior, o usuário deve ser levado a uma nova interface que exibe os detalhes da viagem, incluindo ao menos os seguintes campos:

- Datas da viagem (`dataInicioAfastamento` e `dataFimAfastamento`)
- Nome do servidor
- Motivo da viagem (campo `motivo`)
- Valores (campos `valorTotalRestituicao`, `valorTotalTaxaAgenciamento`, etc).

Considerações:

- A pesquisa de órgãos da primeira interface traz resultados paginados. Essa paginação deve ser tratada de alguma forma (por exemplo: pode existir um botão para ir para a próxima página e um botão pra voltar para a página anterior).
- Outras bibliotecas que facilitem a realização de *requests* na API podem ser usadas (como a *Axios*). Também é possível realizar o trabalho em outro ambiente que não seja o Expo, desde que a aplicação seja React Native. A forma de navegação entre as interfaces também é de escolha livre.
- A aplicação deve tratar as entradas do usuário e exibir mensagens amigáveis em caso de erros (ex.: deve exigir preenchimento de campos obrigatórios, avisar ao usuário se uma consulta não trouxe nenhum resultado, etc).
- O período de pesquisa das viagens na API é de, no máximo, um mês. A aplicação deve controlar esse parâmetro e apresentar uma mensagem amigável ao usuário caso ele informe um período superior.
- O acesso às APIs do Portal Transparência é limitado para evitar sobrecargas nos servidores. Caso ocorram erros ou a API retorne resultados vazios indevidamente, aguarde um ou dois minutos e tente novamente.

Avaliação

A atividade deve ser desenvolvida preferencialmente em duplas, ou individualmente.

A apresentação deve ser realizada até o **dia 14/10** no laboratório, durante o horário de aula, ou no horário reservado pelo professor para atendimentos aos alunos.

Todos os integrantes do grupo devem estar presentes no momento da apresentação.

Será verificado o funcionamento do programa e, em seguida, os alunos devem responder oralmente a questões sobre o código fonte do programa.

Podem ser atribuídas notas diferentes aos alunos de uma dupla, dependendo das respostas aos questionamentos sobre o código do programa.

Trabalhos entregues com atraso terão desconto de 2 pontos por aula de atraso. Passadas três aulas do prazo original, o trabalho não será mais aceito, ou seja, terá nota zero.