GRADUAÇÃO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

TP3 da disciplina Frameworks Front-end e conexão com Back-end apresentada ao curso de Análise e Desenvolvimento de Sistemas, do Instituto INFNET.

Aluno: Camila Nunes de Castro Cerqueira

Turma: GRLADS01C2-M1-L1

Professor: Vanessa Cristina Martins da Silva Frattini

18/03/2022

Sumário

TP3 – < Frameworks Front-end e conexão com Back-end >

Questão 1

O Vue.JS é um framework composto por componentes, através deles é possível dividirmos a nossa aplicação em partes e cada uma assumir uma responsabilidade. Podemos utilizá-los através de registro global ou local. A utilização de componentes deixa o código mais simples e coeso através dessas partes que dividem responsabilidades e além disso, aumenta a possibilidade de reutilização do código, já que partes com um único objetivo podem ser aplicadas em outras partes da aplicação.

Questão 2

Para passar dados de um componente pai para o filho é necessária uma comunicação através de propriedades, assim deve-se utilizar a props. Para passar informações do componente filho para o pai é necessário o uso do $emit que irá permitir ao pai ouvir um evento de um componente filho.

Questão 3

Arquivo html (em CDN) em anexo.

Nome do arquivo: t3-q3-camila-cerqueira.html

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://unpkg.com/vue@next"></script>
    <title>Questão 3 - Tp3</title>
</head>

<body>
    <div id="app"> </div>

    <template id="tp3">
```

```
        <text-color color="red" font-size="50px">{{text}} </text-color>

    </template>

    <template id="text-color-component">
            <p :style="style">
                <slot/>
            </p>
    </template>

    <script>
        const app = Vue.createApp({
            template:"#tp3",
            data(){
                return {
                    text:`O Vue é um framework feito em JavaScript que tem como
principal objetivo o reaproveitamento de código. Nele, podemos criar aplicações web
com maior qualidade e agilidade, e sua curva de aprendizagem é muito pequena.
Embora ainda seja menos ativo no mercado, sua comunidade é bastante ativa, e a cada
dia surgem novas extensões e recursos para serem aparelhados a ele,
                    aumentando muito o seu poder.`,

                }
            }
        })

        app.component('text-color', {
            props:{
                color:{
                    type: String,
                    required: true

                },
                fontSize:{
                    type: String,
                    required: true
                }


            },
            computed: {
                    style(){
                       return {color:this.color,fontSize:
this.fontSize}

                    }
```

```
            },

            template: "#text-color-component"
        })
        app.mount("#app")

    </script>
</body>
</html>
```

Questão 4

Arquivo html (em CDN) em anexo.

Nome do arquivo: t3-q4-camila-cerqueira.html

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://unpkg.com/vue@next"></script>
    <title>Questão 4 - Tp3</title>
</head>

<body>
    <div id="app"> </div>

    <template id="q4">
        <square></square>
    </template>

    <template id="square">
        <div>
            <label id="label-color"> Digite um valor hexadecimal para trocar a cor
do quadrado:
                <input v-model="color" type="text" placeholder="Digite um valor
hexadecimal">
            </label>
            <div class="square" :style="squareColors"></div>
        </div>
    </template>
```

```
    <script>
        const app = Vue.createApp({
            template: "#q4"

        })

        app.component('square', {
            data(){
                return{
                color: ""
                }
            },
            computed:{
                squareColors(){
                    return `background-color: #${this.color}`
                }
            },
            template:"#square"

        })

        app.mount("#app")

    </script>

    <style>
        .square{
            width: 200px;
            height: 200px;
            border: solid 1px black;
            margin-top: 50px;
        }

    </style>
</body>
</html>
```

Questão 5

SPA são aplicações onde todas as funcionalidades estão concentradas em uma única página, ou seja, o navegador vai renderizar a aplicação na sua totalidade apenas uma vez, assim a página não precisa recarregar integralmente e constantemente, isso otimiza a

performance, deixa a navegação mais fluída, dinâmica e permite um tráfego de dados mais leve. O Vue Router é a biblioteca responsável por configurar rotas de navegação das SPA, gerenciar essas rotas no navegador e assim determinar o que será exibido na tela de acordo com o caminho da URL.

Questão 6

O modo hash é o padrão do Vue Router e usa o hash (#) na URL para que seja possível mudar de página sem recarregá-la. Considerando que o padrão de uma URL no modo hash é domínio/#/rota, não é recomendado o uso para páginas que precisam trabalhar com SEO, pois geralmente o que está depois do hash é ignorado pelos buscadores e isso impacta negativamente o ranqueamento da página. Já o modo o history não tem esse problema no SEO, pois permite a troca de páginas sem recarregamento e não necessita do uso da hash (#) na url, no entanto é imprescindível uma configuração no servidor para que o usuário consiga navegar na página, caso contrário ele sempre vai receber o erro 404.

Questão 7

A utilização de rotas dinâmicas é indicada para situações que o conteúdo da página é modificado mas segue uma mesma estrutura, ou seja, um componente "X" é renderizado para todos os usuários mas o conteúdo é diferente para cada um devido ao parâmetro da rota dinâmica. Por exemplo:

Páginas de Imobiliárias com diversos imóveis e que você consiga acessar detalhes de cada um na categoria definida: { path: '/aluguel/:id ', name: 'aluguel', Component: ImovelAluguel }

Utilizar página personalizada de perfil de usuário. { path: '/usuario/:id ', name: 'usuario', Component: Perfil }

Postagem de Posts em blogs e similares { path: '/posts/:id ', name: 'posts', Component: Postagens }

E-commerce com diversas categorias e produtos { path: '/celulares/:id ', name: 'celulares', Component: Celulares}

Questão 8

Flux é uma arquitetura de fluxo unidirecional de dados criada pelo Facebook, utilizada pelo Vuex, para criar aplicações Front End que concentra todos os dados em um único ponto e

faz com o que o fluxo de dados seja inteligível e de fácil manutenção. São quatro interfaces distintas que em conjunto realizam o processo:

Actions->Dispatcher->Store->View

Assim, a interação do usuário vai provocar o caminho acima de fluxo de dados.

No Vuex, as operações síncronas devem ocorrer em "mutations" (onde ficam as funções manipuladoras de mutação) e as assíncronas em "actions" (onde ficam as funções manipuladoras de ação)

Questão 9

Na aplicação com Vuex temos apenas uma Store e devemos criar uma instância de store para gravarmos os estados da aplicação, já que é lá que os dados da nossa aplicação se centralizam. No entanto, não é sempre que devemos colocar todo o estado no Vuex, caso exista na aplicação algum estado que pertença a somente um único componente, ou seja, nenhum outro componente vai precisar, podemos deixá-lo como estado local (nas propriedades do próprio componente).

Questão 10

- O que precisamos criar nos objetos Vuex para nos comunicarmos com o banco? Por quê?

  Precisamos criar as actions, pois é lá que podemos trabalhar com operações assíncronas

- O que precisamos criar nos objetos Vuex para alterarmos o estado da nossa store de maneira síncrona?

Precisamos criar as mutações da Store mutations (mutations:{} dentro na nossa instância Store.

- O que precisamos criar nos objetos Vuex quando queremos buscar, por exemplo, uma lista filtrada?

  Precisamos usar Getters (getters:{} dentro na nossa instância Store.