САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронд-энд разработка

Отчет

Лабораторная работа №3

Разработка одностраничного веб-приложения (SPA) с использованием фреймворка Vue.JS

Выполнил:

Конев Антон

К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2022 г.

Задача: Мигрировать ранее написанный сайт на фреймворк Vue.JS. Минимальные требования:

- Должен быть подключён роутер
- Должна быть реализована работа с внешним API
- Разумное деление на компоненты

Ход работы:

Роутер

```
const router = createRouter( options: {
    history: createWebHistory(import.meta.env.BASE_URL),
    routes: [
        {
            path: '/login',
            name: 'login',
            component: LoginView
        },
        {
            path: '/register',
            name: 'register',
            component: RegisterView
        },
        {
            path: '/',
            name: 'main',
            component: MainView
        },
        {
            path: '/personal',
            name: 'wallet',
            component: WalletView
        },
        {
            path: '/charts',
            name: 'chart',
            component: ChartsView
        },
    ]
})

export default router
```

Внешний API

```
import instance from "@/api/instance.js";
import CoinsAPI from "@/api/coins.js";
import UsersAPI from "@/api/users.js";
import ChartsAPI from "@/api/charts.js";

const coinsAPI = new CoinsAPI(instance);
const usersAPI = new UsersAPI(instance);
const chartsAPI = new ChartsAPI(instance);

export {
    coinsAPI,
    usersAPI,
    chartsAPI
}
```

```
import axios from "axios";

const apiURL = 'http://localhost:3000'

const instance = axios.create({
    baseURL: apiURL
})

export default instance
```

```
class CoinsAPI {
    constructor(instance) {
        this.API = instance
    }

    getCoins = async (search, sort, order, page, limit) => {
        return this.API({
            url: `/coins?q=${search}&_sort=${sort}&_order=${order}&_limit=${limit}&_page=${page}`
        })
    }

    getCustomList = async () => {
        return this.API({
            url: `/coins`
        })
    }
}

export default CoinsAPI
```

Store

```
import { persist } from 'pinia-persists'
import { createPinia } from 'pinia'

const pinia = createPinia()

pinia.use(persist())

export default pinia
```

```
import {defineStore} from "pinia";
import {coinsAPI, usersAPI} from "@/api";

const useCoinsStore = defineStore( id: 'coins', options: {
    state: () => ({
        coins: []
    }),
    actions: {
        async loadCoins(search : string = '', sortName : string = '', page : number = 1, limit : number = 10) {
            const sortSplit = sortName.split( separator: ' ');
            const sort = sortSplit[0];
            const order = sortSplit[1];
            const response = await coinsAPI.getCoins(search, sort, order, page, limit);
            this.coins = response.data;
            return response.data;
        },
        async getWallet(idx) {
            const actual = await usersAPI.getCurrentUser(idx);
            const {id, coins} = actual.data;
            return {id, coins};
        },
        async loadCustomCoins() {
            const response = await coinsAPI.getCustomList();
            this.coins = response.data;
            return response.data;
        },
        async loadCoinsList(search : string = '') {
            const response = await coinsAPI.getCoins(search, sort: '', order: '', page: '', limit: '');
            this.coins = response.data;
            return response.data;
        }
    }
})

export default useCoinsStore
```

Компоненты

Login.vue

```vue
<script>
import {mapActions} from "pinia";
import useUsersStore from "@/stores/users";
import router from "@/router";

export default {
  name: 'Login',
  data() {
    return {
      user: {
        email: "",
        password: "",
      }
    }
  },
  methods: {
    ...mapActions(useUsersStore, keyMapper: ['signUp']),
    async onLogSubmit() {

      await this.signUp(this.user)

    }
  },
  computed: {
    formReady() {
      return Object.values(this.user).every(value => value !== '')
    }
  },
  beforeMount() {
    localStorage.getItem( key: 'pinia_users') ? router.push('/personal') : router.push('/login')
  }
}
</script>
```

Register.vue

```
export default {
  name: 'Register',
  data() {
    return {
      user: {
        firstName: "",
        lastName: "",
        email: "",
        password: "",
        coins: [],
      },
      agree: false
    }
  },
  methods: {
    ...mapActions(useUsersStore, keyMapper: ['register']),
    async onRegSubmit() {

      await this.register(this.user);

    }
  },
  computed: {
    formReady() {
      return Object.values(this.user).every(value => value !== '') && this.agree
    }
  },
  beforeMount() {
    localStorage.getItem( key: 'pinia_users') ? router.push('/personal') : router.push('/register')
  }
}
</script>
```

Область графика

```
export default {
  name: 'Chart',
  components: {
    apexcharts: VueApexCharts,
  },
  props: {
    target: {
      type: Object,
      required: true
    }
  },
  data() {
    return {
      chartOptions: {
        chart: {
          id: 'basic-bar'
        },
        xaxis: {
          categories: []
        }
      },
      series: [{
        name: '',
        data: []
      }]
    }
  },
  computed: {},
  watch: {
    target() {
      for (let i = 0; i < this.target.length; i++) {
        this.chartOptions.xaxis.categories[i] = this.target[i][0]
        this.series[0].data[i] = this.target[i][1].toFixed( fractionDigits: 5)
      }
    }
  }
}
</script>
```

Header.vue

```
export default {
  name: 'Header',
  data() {
    return {
      isLogged: false
    }
  },
  methods: {
    logout() {
      if (confirm('Вы действительно хотите выйти?')) {
        localStorage.clear()
        if (this.$route.path === '/personal') {
          router.push('/')
        } else {
          window.location.reload()
        }
      }
    },
    checkAuth() {
      if (!this.isLogged) {
        router.push('/login')
      } else {
        router.push('/personal')
      }
    }
  },
  mounted() {
    localStorage.getItem( key: 'pinia_users') ? this.isLogged = true : this.isLogged = false
  }
}
</script>
```

Лист монет для основной страницы

```
export default {
  props: {
    coins: {
      type: Array,
      required: true,
    }
  },
  emits: ['buyEvent'],
  data() {
    return {
      coin: {id: Number...},
      isPositive: 'text-danger',
    }
  },
  methods: {
    formatDate(value) {...},
    buyEvent(id) {
      if (!localStorage.getItem( key: 'pinia_users')) {
        router.push('/login');
      } else {
        let amount;
        while (true) {
          amount = prompt( message: "Сколько хотите купить?")
          if (!amount) {
            break
          } else if (isNaN(amount)) {
            alert("Количество должно быть числом")
          } else if (amount <= 0) {
            alert("Количество должно быть положительным")
          } else {
            this.$emit('buyEvent', id, parseFloat(amount))
            break
          }
        }
      }
    }
  }
}
```

Область кошелька пользователя

```
export default {
  props: {...},
  emits: ['buyEvent', 'sellEvent'],
  data() {...},
  methods: {
    buyEvent(id) {
      let amount;
      while (true) {
        amount = prompt( message: "Сколько хотите купить?")
        if (!amount) {
          break
        } else if (isNaN(amount)) {
          alert("Количество должно быть числом")
        } else if (amount <= 0) {
          alert("Количество должно быть положительным")
        } else {
          this.$emit('buyEvent', id, parseFloat(amount))
          break
        }
      }
    },
    sellEvent(id, currentAmount) {
      let amount;
      while (true) {
        amount = prompt( message: "Сколько хотите продать?")
        if (!amount) {
          break
        } else if (isNaN(amount)) {
          alert("Количество должно быть числом")
        } else if (amount <= 0) {
          alert("Количество должно быть положительным")
        } else if (amount > currentAmount) {
          alert("У вас нет столько")
        } else if (amount == currentAmount) {
          if (confirm("Хотите продать все?")) {
            this.$emit('sellEvent', id, parseFloat(amount))
            break;
          }
```

Основная страница

```
export default {
  name: 'Main',
  components: {
    CoinsList
  },
  data() {
    return {
      coins: [],
      page: 1,
      limit: 10,
      total: 3,
      search: '',
      sortName: '',
      sortOptions: [
        {value: 'name ASC', name: 'По названию (ASC)'},
        {value: 'name DESC', name: 'По названию (DESC)'},
        {value: 'current_price ASC', name: 'По цене (ASC)'},
        {value: 'current_price DESC', name: 'По цене (DESC)'},
        {value: 'price_change_percentage_24h ASC', name: 'По изменению (ASC)'},
        {value: 'price_change_percentage_24h DESC', name: 'По изменению (DESC)'},
        {value: 'atl_date ASC', name: 'По дате (ASC)'},
        {value: 'atl_date DESC', name: 'По дате (DESC)'},
      ]
    }
  },
  methods: {
    ...mapActions(useCoinsStore,  keyMapper: ['loadCoins']),
    ...mapActions(useUsersStore,  keyMapper: ['commitActions']),
    async getCoins() {
      this.coins = await this.loadCoins(this.search, this.sortName);
    },
    buyCoin(id, amount) {
      let newCoin = true
      for (let i = 0; i < this.user.coins.length; i++) {
        if (id === this.user.coins[i].id) {
          this.user.coins[i].amount += amount;
          newCoin = false;
```

```
      if (newCoin) {
        this.user.coins.push({id: id, amount: amount});
      }

      this.commitActions(this.user);
    },
    async findCoins() {
      this.coins = await this.loadCoins(this.search, this.sortName);
    },
    async getPrevPage() {
      if (this.page > 1) {
        this.page--;
        this.coins = await this.loadCoins(this.search, this.sortName, this.page);
      }

    },
    async getNextPage() {
      if (this.page < this.total) {
        this.page++;
        this.coins = await this.loadCoins(this.search, this.sortName, this.page);
      }
    }
  },
  computed: {
    ...mapState(useUsersStore, keys: ['user'])
  },
  mounted() {
    this.getCoins();
  },
  watch: {
    async sortName(sortName) {
      this.coins = await this.loadCoins(this.search, sortName);
    }
  }
}
</script>
```

Кастомное поле ввода

```vue
<template>
  <input type="text" class="input-group d-flex" placeholder="Поиск по названию"
         :value="modelValue" @input="updateInput" @change="sendSearch"
  >
  <span class="right-pan">
    <button class="bg-light clear" style="..."
            @click="clearInput"
    >
      <svg style="...">
        <use xlink:href="#clear">
        </use>
      </svg>
    </button>
  </span>
</template>

<script>
export default {
  name: 'search-input',
  emits: ['sendSearch'],
  props: {
    modelValue: String,
  },
  methods: {
    updateInput(event) {
      this.$emit('update:modelValue', event.target.value)
    },
    clearInput() {
      this.$emit('update:modelValue', '')
      this.sendSearch()
    },
    sendSearch(event) {
      if (event) {
        this.$emit('sendSearch', event.target.value)
      } else {
        this.$emit('sendSearch', '')
      }

    }
```

Вью для основной страницы

```vue
<template>
  <Header/>
  <Main/>
</template>
<script>

import ...

export default {
  components: {
    Header,
    Main
  }
}
</script>
```

Вью для страницы графиков

```vue
<template>
  <Header/>
  <Chart/>
</template>
<script>
import ...
export default {
  components: {
    Header,
    Chart
  }
}
</script>
```

App.vue

```vue
<template>
  <div id="app">
    <BaseLayout>
      <router-view/>
    </BaseLayout>
  </div>
</template>

<script setup>
import BaseLayout from '@/layouts/BaseLayout.vue'

</script>
```

main.js

```
import { createApp } from 'vue'

import App from '@/App.vue'
import router from '@/router'
import store from "@/stores";
import components from '@/components/UI'

import 'bootstrap/dist/css/bootstrap.min.css'
import 'bootstrap'
import '@/assets/main.css'
import '@/assets/dark.css'
import '@/assets/light.css'
import VueApexCharts from "vue3-apexcharts";

const app = createApp(App)

components.forEach(component => {
    app.component(component.name, component)
})

app.component( name: 'apexchart', VueApexCharts)

app.use(store)
app.use(router)

app.mount( rootContainer: '#app')
```

Вывод: в ходе лабораторной работы сайт криптобиржи был перенесен на Vue.js: выполнено разумное деление на компоненты, подключен роутер и сторы, также есть взаимодействие с внешним API – json-server.