САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронд-энд разработка

Отчет

Лабораторная работа №3

Выполнила:

Мухина Юлия

Группа К33401

Проверил:
Добряков Д. И.

Санкт-Петербург

2021 г.

Задача

Мигрировать ранее написанный сайт на фреймворк Vue.JS.

Ход работы

Реализована регистрация

```
<script>
import { mapActions, mapState } from 'pinia'
import router from '@/router'

import useRegisterStore from '../stores/register'

export default {
    name: 'RegBlock',

data() {
    return {
        form: {
            firstname: "",
            lastname: "",
            email: "",
            password: "",
            login: ""
        }
    };
},

methods: {
    ...mapActions(useRegisterStore, ['userRegister']),

    async register() {
        const response = await this.userRegister(this.form);

        const { accessToken, user } = response.data

        localStorage.accessToken = accessToken
        localStorage.user = JSON.stringify(user)

        localStorage.accessToken ? router.push('/') : router.push('')
    }
}
}
</script>
```

Реализован вход

```
<script>
import { mapActions, mapState } from 'pinia'
import router from '@/router'

import useLoginStore from '../stores/login'

export default {
    name: 'EnterBlock',

    data() {
    return {
        form: {
            email: "",
            password: ""
        }
    };
},

methods: {
    ...mapActions(useLoginStore, ['userLogin']),

    async login() {
        const response = await this.userLogin(this.form);

        const { accessToken, user } = response.data

        localStorage.accessToken = accessToken
        localStorage.user = JSON.stringify(user)

        localStorage.accessToken ? router.push('/') : router.push('/enter')
    }
}
}
</script>
```

Реализована проверка авторизации

```
<script>
export default {
    name: 'HeaderBlock',

    methods: {
        check() {
            const idLogOut = document.querySelector('#logout');
            const idLogIn = document.querySelector('#enter');
            const enterLK = document.querySelector('#enterlk');

            if (localStorage.accessToken){
                idLogIn.classList.add("d-none");
                idLogOut.classList.remove("d-none");
            } else {
                idLogIn.classList.remove("d-none");
                enterLK.classList.add("d-none");
                idLogOut.classList.add("d-none");
            }
        },

        exit() {
            const idLogOut = document.querySelector('#logout');
            const idLogIn = document.querySelector('#enter');
            const enterLK = document.querySelector('#enterlk');
            localStorage.clear();
            idLogIn.classList.remove("d-none");
            enterLK.classList.add("d-none");
            idLogOut.classList.add("d-none");
        }
    },

    mounted() {
        this.check()
    }
}
</script>
```

Реализована фильтрация предложений

```vue
<script>
import {mapActions, mapState} from 'pinia'
import useCardsStore from '@/stores/cards'
import NoteBlock from '@/components/note-card.vue'

export default {
    name: 'MainBlock',

    components: { NoteBlock },

    computed: {
        ...mapState(useCardsStore, ['cards']),
        filteredCards() {
            if (this.selectedCards.length) {
                return this.selectedCards;
            } else {
                return this.cards
            }
        }
    },

    methods: {
        ...mapActions(useCardsStore, ['loadCards']),
        async filter() {
            const eventT = document.querySelector('.event-type')
            const cityT = document.querySelector('.region-type')
            this.selectedCards = []

            console.log(eventT.value, cityT.value)
            this.cards.forEach((card) => {
                if ((eventT.value === card.category && cityT.value === card.category2) || (eventT.value === "ALL" && cityT.value === card.category2) || (eventT.value === card.category && cityT.value === "ALL")) {
                    this.selectedCards.push(card);
                }
            })
        }
    },
    data() {
        return {
            selectedCards: []
        }
    },
    mounted() {
        this.loadCards();
    }
}
</script>
```


Функция записи на мероприятие

```vue
<script>
import {mapActions, mapState} from 'pinia'
import useUserEventsStore from '@/stores/userEvents'

export default {
    name: 'NoteBlock',

    props: {
    mero: {
      type: String,
      required: true
    },
    text: {
      type: String,
      required: true
    },
        metro: {
            type: String,
      required: true
        },
        data: {
            type: String,
      required: true
        },
        address: {
            type: String,
      required: true
        },
        id: {
            type: Number,
      required: true
        },
        img: {
            type: String,
      required: true
        },
  },
```

```
    methods: {
        ...mapActions(useUserEventsStore, ['addUserEvents']),
        async addNote(id) {
            const userEvents = {
                "userId": JSON.parse(localStorage.user).id,
                "eventId": id
            }
            console.log(userEvents)

            const response = await this.addUserEvents(userEvents);

            console.log(response)
        }
    }
}
</script>
```

Показывать мероприятия, на которые записан пользователь

```vue
<script>
import {mapActions, mapState} from 'pinia'

import useUserEventsStore from '@/stores/userEvents'
import useCardsStore from '@/stores/cards'
import PersonalCards from '@/components/PersonalCards.vue'

export default {
    name: 'LkBlock',

    components: {PersonalCards},

    computed: {
        ...mapState(useUserEventsStore, ['userEvents']),
        ...mapState(useCardsStore, ['personalCards']),

    },

    methods: {
        ...mapActions(useUserEventsStore, ['getUserEventsById']),
        ...mapActions(useCardsStore, ['loadCardById', 'doClear']),

        async loadPage() {
            const response = await this.getUserEventsById(JSON.parse(localStorage.user).id);
            const result = Array.from(response.data);

            this.doClear();
            result.forEach((item) => {
                this.loadCardById(item.eventId, item.id)
            })
        }

    },

    mounted() {
        this.loadPage()
    }
}
</script>
```

Отписка от мероприятия

```
<script>
import { mapActions, mapState } from "pinia";
import useUserEventsStore from "@/stores/userEvents.js"

export default {
    name: 'PersonalCards',

    computed: {
        ...mapState(useUserEventsStore, ['userEvents']),
    },

    methods: {
        ...mapActions(useUserEventsStore, ['addUserEvents', 'deleteCardById']),

        async deleteCard(id) {
            console.log('vbnm')
            this.deleteCardById(id)

            location.reload()
        }
    },

    props: {
        mero: {
            type: String,
            required: true
        },
        metro: {
            type: String,
            required: true
        },
        data: {
            type: String,
            required: true
        },
        address: {
            type: String,
            required: true
        },
        primaryId: {
            type: Number,
            required: true
        }
    }
}
</script>
```

Создание мероприятий в календаре

```vue
<script>
import { mapActions, mapState } from 'pinia'
import { Modal } from 'bootstrap'

import useCalendarEventsStore from '@/stores/calendarEvents'

import BaseLayout from '@/layouts/BaseLayout.vue'
import HeaderBlock from '@/components/Header.vue'
import FullCalendar from '@/components/FullCalendar.vue'
import FooterBlock from '@/components/Footer.vue'

export default {
  name: 'CalendarPage',

  components: { BaseLayout, FullCalendar, HeaderBlock, FooterBlock},

  computed: {
    ...mapState(useCalendarEventsStore, {
      calendarEvents: 'calendarEvents',
      selectedEvent: (state) => {
        return {
          ...state.selectedEvent,
          formattedDate: () => {
            const date = state.selectedEvent.date
            return new Date(date).toLocaleDateString('ru-RU')
          }
        }
      }
    })
  },

  data() {
    return {
      form: {
        title: '',
        description: '',
        date: ''
      },

      eventCreateModal: null
    }
  },
```

```
  methods: {
    ...mapActions(useCalendarEventsStore, ['loadCalendarEvents', 'loadEventById', 'createEvent', 'deleteEvent']),

    handleEventChange(payload) {
      console.log('event change', payload)
    },

    handleDateClick(payload) {
      console.log('date clicked', payload)

      const { dateStr } = payload
      this.form.date = dateStr

      this.eventCreateModal = new Modal(this.$refs.eventCreate)
      this.eventCreateModal.show()
    },

    async handleEventClick(payload) {
      await this.loadEventById(payload.event._def.publicId)

      const eventModal = new Modal(this.$refs.detailEvent)
      eventModal.show()
    },

    async submitForm() {
      await this.createEvent(this.form)

      this.$refs.eventForm.reset()
      this.eventCreateModal.hide()

      await this.loadCalendarEvents()
    },

    async deleteMero(id) {
      this.deleteEvent(id);

      location.reload();
    }
  },

  mounted() {
    this.loadCalendarEvents()
  }
}
</script>
```

Вывод

Научились мигрировать ранее написанный сайт на фреймворк Vue.JS.

((зачтите пожалуйста))