САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронд-энд разработка

Отчет

Лабораторная работа 2

Выполнила:

Самчук Анита

К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2021 г.

Задача

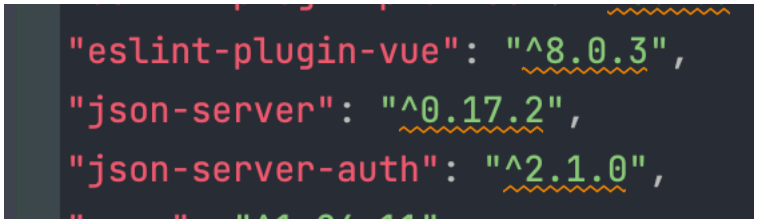
Реализовать взаимодействие с внешним API

Ход работы

В качестве стороннего api с картинками был выбран https://unsplash.com/

Для авторизации и сохранения лайков пользователей установлен и настроен json-server-auth

Запросы отправлялись при помощи axios



Функция регистрации проверяет совпадают ли пароли, все остальное отрабатывает bootstrap и json-server-auth

```
async signup () {
  if (this.form.password === this.form.password2 && this.form.password !== null) {
    try {
      const response = await axios.post(
        'http://localhost:3000/signup',
        {
          email: this.form.email,
          password: this.form.password,
          username: this.form.username
        })

      if (response.status === 201) {
        this.$router.push('/login')
      }
    } catch (e) {
      alert(e.response.data)
    }
  } else {
    alert('Пароли не совпадают')
  }
}
```

Успешный логин сохраняет accessToken и информацию о пользователе в localStorage

```
async login () {
  try {
    const response = await axios.post('http://localhost:3000/login', this.form)

    if (response.status === 200) {
      localStorage.setItem('accessToken', response.data.accessToken)
      localStorage.setItem('user', JSON.stringify(response.data.user))
      this.$router.push('/')
    }
  } catch (e) {
    alert(e.response.data)
  }
}
```

Функция логаута очищает localStorage и перенаправляет на страницу логина

```
methods: {
  logout () {
    localStorage.removeItem( key: 'accessToken')
    localStorage.removeItem( key: 'user')
    this.$router.push('/login')
  }
```

Для главной страницы реализовано 2 запроса на фотографии – поиск и 20 фотографий без поиска. Это ограничение на unsplash, искать фотографии без query нельзя. В параметр client_id передаётся accessToken к api.

```
async searchPhotos (context, { query, filters }) {
  if (query == null) {
    return
  }

  if (filters.color === null) {
    delete filters.color
  }

  try {
    const response = await axios.get('https://api.unsplash.com/search/photos', {
      params: {
        client_id: accessKey,
        per_page: 20,
        query,
        ...filters
      }
    })

    if (response.status === 200) {
      context.commit('setPhotos', response.data.results)
    }
  } catch (e) {
    console.log(e)
  }
},
```

```
async getPhotos (context) {
  try {
    const response = await axios.get('https://api.unsplash.com/photos', {
      params: {
        client_id: accessKey,
        order_by: 'popular',
        per_page: 20
      }
    })

    if (response.status === 200) {
      context.commit('setPhotos', response.data)
    }
  } catch (e) {
    console.log(e)
  }
},
```

Функции лайка и удаления лайка, а также список залайканных фотографий и обновления профиля отправляются на json сервер с авторизацией по bearer токену

```
async like () {
  try {
    const userId = JSON.parse(localStorage.getItem( key: 'user')).id

    const response = await axios.post(
      'http://localhost:3000/600/likedPhotos',
      {
        photo: this.photo.id,
        url: this.photo.urls.small,
        userId
      },
      {
        headers: {
          Authorization: 'Bearer ' + localStorage.accessToken
        }
      }
    )

    if (response.status === 201) {
      await this.checkLike()
    }
  } catch (e) {
    alert(e.response.data)
  }
},
```

```
async updateProfile () {
  if (!this.user.username || !this.user.email) {
    alert('Email and username cannot be empty')
    return
  }

  try {
    const userId = JSON.parse(localStorage.getItem( key: 'user')).id

    const response = await axios.patch(
      'http://localhost:3000/users/' + userId,
      this.user,
      {
        headers: {
          Authorization: 'Bearer ' + localStorage.accessToken
        }
      }
    )

    if (response.status === 200) {
      this.loadProfile()
    }
  } catch (e) {
    alert(e.response.data)
  }
}
}
```

Вывод

Успешно реализовано взаимодействие с внешним api для проекта picterest