

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Рыбалко Олег

Группа K33392

Проверил:

Добряков Д. И.

Санкт-Петербург

2023 г.

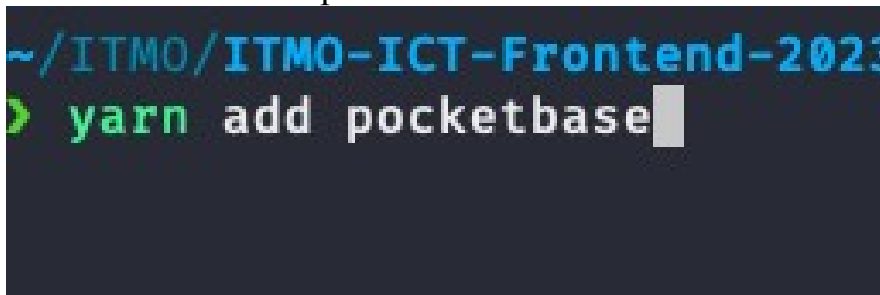
Задание

В данной лабораторной работе было необходимо привязать то, что Вы делали в ЛР1 к внешнему API средствами fetch/axios/xhr.

Решение

В качестве бекенда для реализации авторизации пользователей, хранения данных о публикациях, подписках и лайках мы будем использовать pocketbase (<https://pocketbase.io/>), так как он из коробки предоставляет интерфейс для CRUD операций над SQLite базой данных и автоматически создает API для работы с таблицами, что упростит нашу реализацию веб-сайта без необходимости написания собственного бекенд-сервиса.

Для работы с pocketbase мы будем использовать JS SDK (<https://github.com/pocketbase/js-sdk>), так как он предоставляет удобный интерфейс для работы с pocketbase HTTP API. Команда для установки SDK показана ниже на скриншоте.



```
~/ITMO/ITMO-ICT-Frontend-2023
> yarn add pocketbase
```

На примере профиля пользователя покажем работу JS SDK для получения публикаций пользователя из базы данных.



```
useEffect(() => {
  pb.collection('posts')
    .getFullList({
      filter: `author="${userData.id}"`,
      expand: 'author',
      sort: '-created',
    })
    .then((records) => {
      setPosts(
        records.map((r) => {
          return {
            id: r.id,
            title: r.title,
            body: r.body,
            authorUsername: r.expand!.author.username,
            likesCount: r.likesCount,
          }
        })
      )
    })
}, [userData])
```

Для создания новой публикации воспользуемся методом *create* у объекта *pb.collection(«posts»)*.

```
pb.collection('posts')
  .create({
    title: newPost.title,
    body: newPost.body,
    author: authStore.id,
  })
  .then((record) => {
    setPosts([
      {
        id: record.id,
        title: record.title,
        body: record.body,
        authorUsername: authStore.username,
        likesCount: 0,
      },
      ...posts,
    ])
    setShowNewPostModal(false)
    setNewPost({ title: '', body: '' })
  })
  .catch(() => alert('failed to publish'))
```

Для удаления данных воспользуемся методом *delete* у объекта *pb.collection(«posts»)*

```
const deletePost = (post: PostType) => {
  pb.collection('posts')
    .delete(post.id)
    .then(() => setPosts(posts.filter((el) => el.id !== post.id)))
    .catch(() => alert('failed to delete the post'))
}
```

Для регистрации пользователей в нашей системе воспользуемся методом *create* у коллекции *users*. Pocketbase умеет отправлять коды верификации на почту, дает возможность восстановить аккаунт пользователя и много того, что мы не будем затрагивать в рамках данной лабораторной работы.

Для входа в аккаунт пользователя воспользуемся методом *authWithPassword* коллекции *users*.

```
const signIn = useCallback(() => {  
  pb.collection('users')  
    .authWithPassword(userData.username, userData.password)  
    .then((model) => {  
      store.dispatch(loginAction({ ...userData, id: model.record.id }))  
      navigate(`/profile/${userData.username}`)  
    })  
    .catch(() => alert('failed to log in. check your credentials'))  
}, [userData, navigate])
```

Вывод

В ходе данной лабораторной работы мы реализовали полностью рабочий прототип блога, который связан с базой данных и поддерживает регистрацию пользователей в системе.