**Министерство науки и высшего образования РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

Лабораторная работа №6 по дисциплине

«Технологии Web-программирования»

Тема: Работа с HTTP запросами

Выполнил:

студент группы ПВ-41

Колесников В.Д.

Проверил: Картамышев С. В.

Белгород 2019 г.

**Цель работы:** Изучить принципы работы отправки Ajax запросов. Изучить

работу отправки синхронных и асинхронных запросов в Vue JS. Получить

навыки работы с промисами.

**Выполнение**

Приложение предполагает следующую структуру: Имеется база данных, в которой существуют три таблицы: эмоции, которые отображаются на основной странице, таблица с пользователями (которые имеют доступ к сайту и могут просматривать, добавлять, изменять, удалять и оставлять комментарии к эмоциям) и таблица с комментариями пользователей к эмоциям. Пользователи также могут заходить на сайт или регистрироваться.

Вся данная логика была реализована на стороне фронта, и получился следующий результат:

Регистрация, вход и выход:

SignIn.vue

import ***router*** from "../router";  
  
export default {  
 name: "SignIn",  
 data() {  
 return {  
 username: '',  
 password: '',  
 }  
 },  
 methods: {  
 handleSubmit: function (e) {  
 e.preventDefault()  
 const submissionData = {  
 username: this.username,  
 password: this.password,  
 }  
 this.$http.post('/user/login', submissionData).then((response) => {  
 ***localStorage***.setItem('userData', ***JSON***.stringify(response.data))  
 ***router***.push({ name: 'Home' })  
 }).catch((error) => alert(error.response.data.statusText))  
 }  
 }  
}

SignUp.vue

import ***router*** from "../router";  
  
export default {  
 name: "SignUp",  
 data() {  
 return {  
 username: '',  
 password: '',  
 }  
 },  
 methods: {  
 handleSubmit: function (e) {  
 e.preventDefault()  
 const submissionData = {  
 username: this.username,  
 password: this.password,  
 }  
 this.$http.post('/user/register', submissionData).then((response) => {  
 ***localStorage***.setItem('userData', ***JSON***.stringify(response.data))  
 ***router***.push({ name: 'SignIn' })  
 }).catch((error) => alert(error.response.data.statusText))  
 }  
 }  
}

Работа с эмоциями:

Home.vue

createEmote(submissionData) {  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 this.$http.post('/emote/create', submissionData, {params: {accessToken: userData.accessToken}}).then(() => {  
 this.$http.get('/emote/all').then((response) => this.emotes = [...response.data])  
 }).catch((error) => alert(error.response.data.statusText))  
},  
deleteEmote(emoteID){  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 this.$http.delete('/emote/delete', { params: { emoteID, accessToken: userData.accessToken } }).then(() => {  
 this.$http.get('/emote/all').then((response) => this.emotes = [...response.data])  
 }).catch((error) => alert(error.response.data.statusText))  
},  
editEmote(emoteID, submissionData){  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 this.$http.post('/emote/update', submissionData, { params: { emoteID, accessToken: userData.accessToken } }).then(() => {  
 this.$http.get('/emote/all').then((response) => this.emotes = [...response.data])  
 }).catch((error) => alert(error.response.data.statusText))  
}

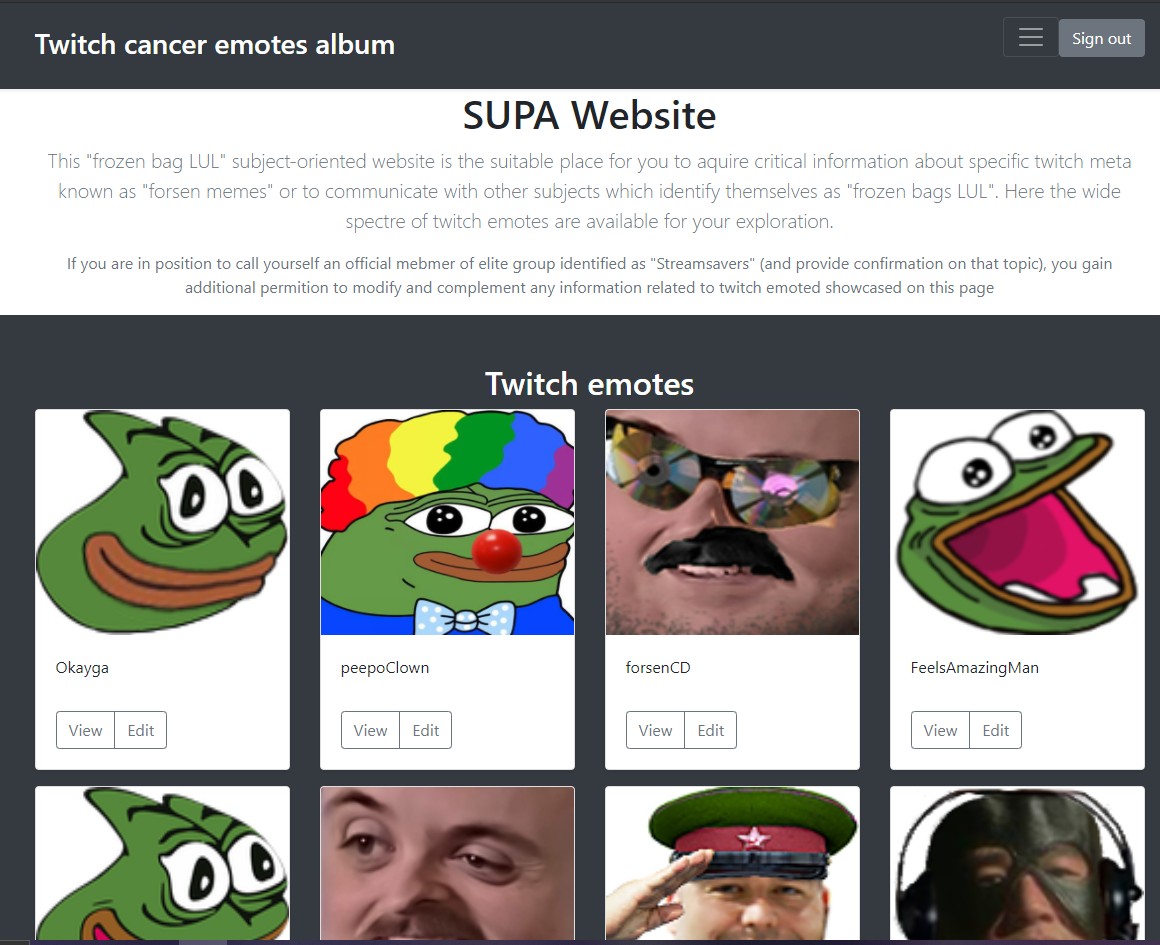
Работа с комментариями:

AlbumCard.vue

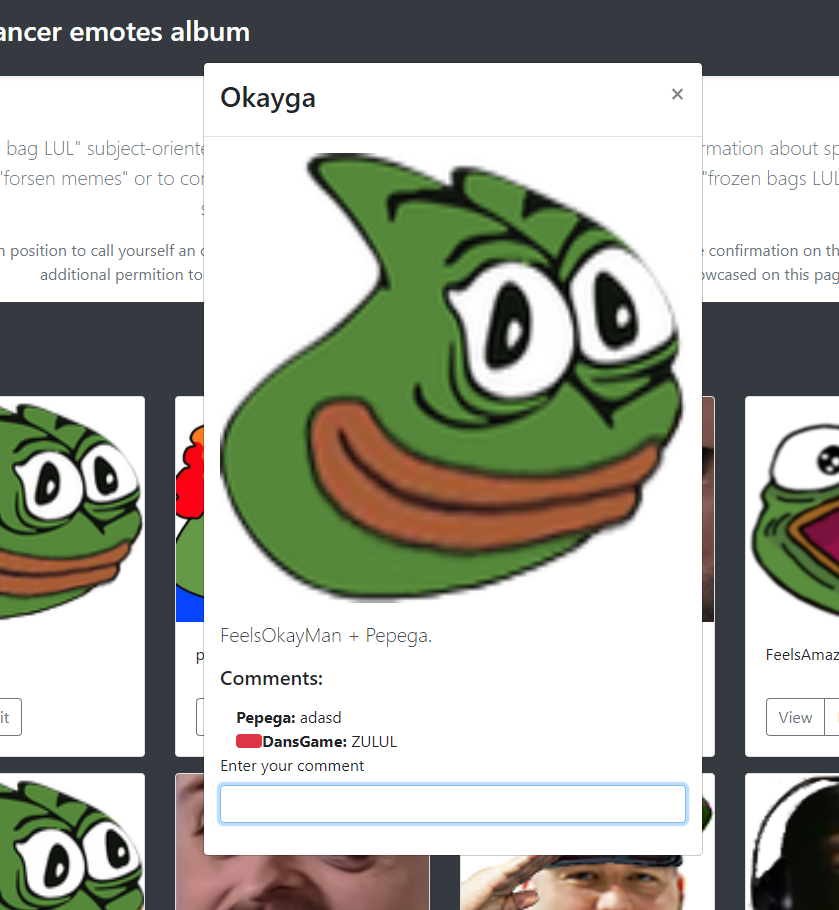
import Comment from "./Comment";  
export default {  
 name: "AlbumCard",  
 data(){  
 return{  
 comments: [],  
 users: [],  
 emote: null,  
 userData: null,  
 newComment: '',  
 url: '',  
 name: '',  
 description: ''  
 }  
 },  
 components: {  
 Comment  
 },  
 props: [  
 'emotesData'  
 ],  
 created(){  
 this.emote = this.emotesData  
 this.$http.get('/comment/emoteid', { params: { emote\_id: this.emote.id } })  
 .then((response) => this.comments = [...response.data])  
 this.$http.get('/user/all')  
 .then((response) => this.users = [...response.data])  
 if (***localStorage***.getItem('userData') === null) {  
 return  
 }  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 this.userData = { ...userData }  
 },  
 methods: {  
 viewModal() {  
 this.$refs['viewModal'].show()  
 },  
 editModal() {  
 this.$refs['editModal'].show()  
 },  
 saveModal() {  
 if (***localStorage***.getItem('userData') === null) {  
 alert('User not authorised.')  
 return  
 }  
 const submissionData = {  
 name: this.name,  
 description: this.description,  
 image: this.url  
 }  
 this.$emit('edit-emote', this.emotesData.id, submissionData)  
 this.name = ''  
 this.description = ''  
 this.url = ''  
 this.$refs['editModal'].hide()  
 },  
 deleteModal(emoteID) {  
 if (***localStorage***.getItem('userData') === null) {  
 alert('User not authorised.')  
 return  
 }  
 this.$emit('delete-emote', emoteID)  
 this.$refs['editModal'].hide()  
 },  
 submitComment(){  
 if (this.newComment !== '') {  
 if (***localStorage***.getItem('userData') === null) {  
 alert('User not authorised.')  
 return  
 }  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 const submissionData = {  
 comment: this.newComment,  
 user\_id: userData.id,  
 emote\_id: this.emote.id  
 }  
 ***console***.log(submissionData)  
 ***console***.log(userData.accessToken)  
 this.$http.post('/comment/create', submissionData, { params: { accessToken: userData.accessToken } }).then(() => {  
 this.$http.get('/comment/emoteid', { params: { emote\_id: this.emote.id } }).then((response) => this.comments = [...response.data])  
 }).catch((error) => alert(error.response.data.statusText))  
 }  
 this.newComment = ''  
 },  
 deleteComment(id){  
 if (***localStorage***.getItem('userData') === null) {  
 alert('User not authorised.')  
 return  
 }  
 const userData = ***JSON***.parse(***localStorage***.getItem('userData'));  
 ***console***.log(id)  
 ***console***.log(userData.accessToken)  
 this.$http.delete('/comment/delete', { params: { id, accessToken: userData.accessToken } }).then(() => {  
 this.$http.get('/comment/emoteid', { params: { emote\_id: this.emote.id } }).then((response) => this.comments = [...response.data])  
 }).catch((error) => alert(error.response.data.statusText))  
 }  
 }  
}

В итоге, получился следующий результат:

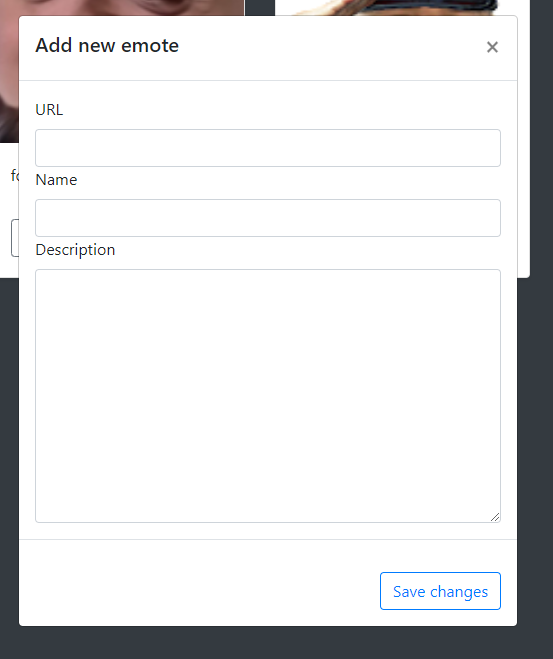
Главная страница



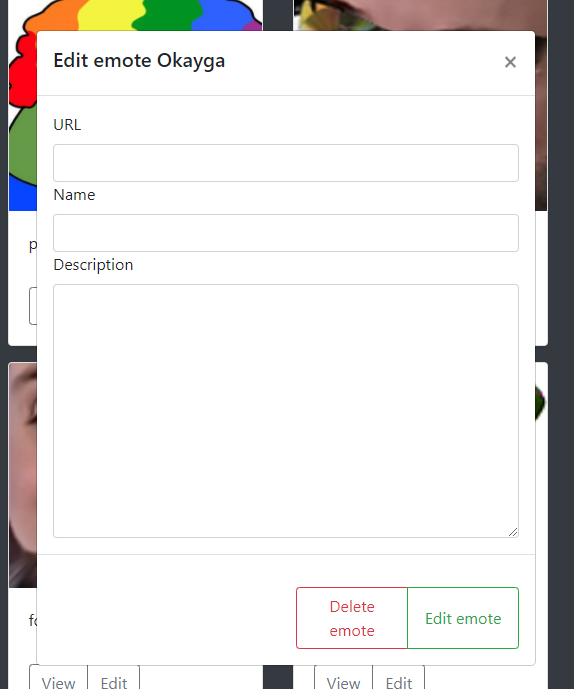
Просмотр эмоции и комментариев (красный прямоугольник означает возможность удаления своего комментария)



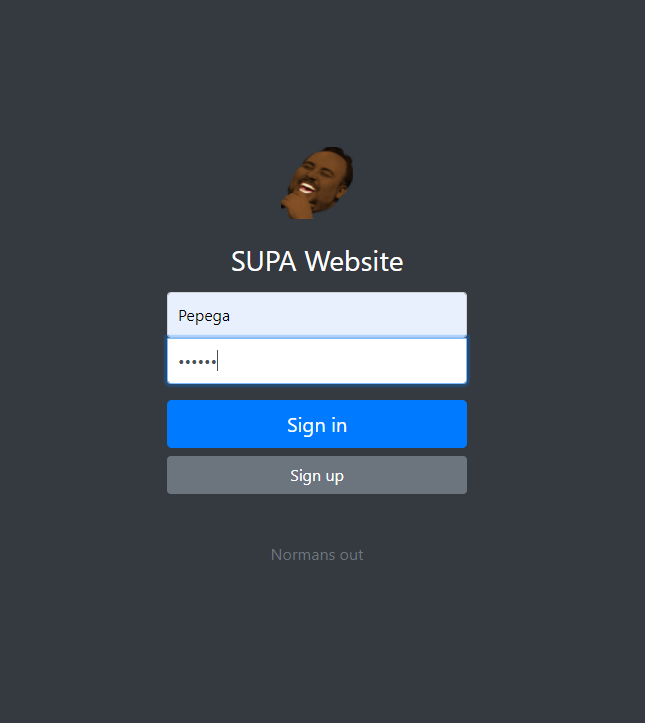
Создание эмоции



Изменение эмоции



Вход



Вывод: в результате выполнения работы были изучены принципы работы отправки ajax-запросов, принцип работы отправки синхронных и асинхронных запросов в vuejs, и получены навыки работы с promise.