**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«**БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ 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: ст. группы ПВ-41

Таран Алексей Николаевич

Проверил: Картамышев С.В.

Белгород 2020

**Цель**: изучить принципы работы отправки Ajax запросов. Изучить работу отправки синхронных и асинхронных запросов в Vue JS. Получить навыки работы с промисами.

**Задание к лабораторной работе:**

1. Изучить возможности Vue js для отправки http запросов.
2. Выбрать подходящую библиотеку для работы с запросами.
3. Реализовать взаимодействие фронтенда с REST API, спроектированном в прошлой лабораторной работе.
4. Продемонстрировать работу взаимодействия фронтенд приложения с REST API.

**Выполнение работы**

Для выполнения http-запросов из Vue.js была выбрана библиотека Axios.

При помощи axios.get(url).then(promise), можно совершить Ajaх get-запрос по заданному url.

При помощи axios.post(url, body-dict).then(promise), можно совершить Ajaх post-запрос по заданному url. body-dict представляет собой словарь, содержащий пары ключ значения переданных параметров.

Реализации ajax-запросов из vue для контроллеров:

1) Работа с пользователем:

Вход в систему:

signIn(e) {

let thisEnv = this

if (this.validate()) {

if (this.needRegistration()){

this.signUp(e)

return

}

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

var token = null

instance.post('/user/signin', {"login": thisEnv.email, "password": thisEnv.password}).then(function(responseToken){

token = responseToken.data

if (token != false){

User.login({

email: thisEnv.email,

accessToken: token

})

e.preventDefault()

thisEnv.$forceUpdate()

thisEnv.parentRefresh()

} else {

thisEnv.errors.email = 'Введены неверные данные'

thisEnv.errors.password = ''

thisEnv.password = ''

e.preventDefault()

}

})

}

},

Регистрация:

signUp(e){

let thisEnv = this

var token = null

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

instance.post('/user/signup', {"login": thisEnv.email, "password": thisEnv.password}).then(function(responseToken){

token = responseToken.data.token

if (responseToken.data.statusText == undefined){

User.login({

email: thisEnv.email,

accessToken: token

})

e.preventDefault()

thisEnv.$forceUpdate()

thisEnv.parentRefresh()

} else {

thisEnv.errors.email = 'Пользователь с таким логином уже существует!'

thisEnv.errors.password = ''

thisEnv.password = ''

document.getElementById('gridCheck1').checked = false

e.preventDefault()

}

})

},

Выход из системы:

logout() {

let thisEnv = this

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

let token = JSON.parse(localStorage.getItem('user')).accessToken

instance.post('/user/exit', {"token": token}).then(function(response){

if (response.data.accessToken == null){

User.logout()

thisEnv.password = ''

thisEnv.$forceUpdate()

thisEnv.parentRefresh()

}

})

}

2) Запросы для комментариев:

removeComment(id){

var thisEnv = this

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

let token = JSON.parse(localStorage.getItem('user')).accessToken

instance.post('/attraction/delete\_comment', {"accesstoken": token, "id": id}).then(function(responseDelete){

let deleteStatus = responseDelete.data

if (deleteStatus.status != null){

thisEnv.commentsList = thisEnv.commentsList.filter(t => t.id != id)

}

})

},

updateComment(id, text){

var thisEnv = this

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

let recievedtext = text

let token = JSON.parse(localStorage.getItem('user')).accessToken

var index = 0

for (index; index < thisEnv.commentsList.length; index++){

if (thisEnv.commentsList[index].id == id){

break

}

}

let oldText = thisEnv.commentsList[index].text

thisEnv.commentsList[index].text = text

instance.post('/attraction/edit\_comment', {"accesstoken": token, "id": id, "text": recievedtext}).then(function(responseComment){

let dataComment = responseComment.data

// При неудачном обновлении, вернуть старый текст

if (dataComment.text != thisEnv.commentsList[index].text){

thisEnv.commentsList[index].text = oldText

}

})

},

addComment(text){

var thisEnv = this

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

let recievedtext = text

let token = JSON.parse(localStorage.getItem('user')).accessToken

instance.post('/attraction/new\_comment', {"accesstoken": token, "text": recievedtext}).then(function(responseComment){

let dataComment = responseComment.data

thisEnv.commentsList.unshift({

id: dataComment.id,

username: dataComment.username,

date: dataComment.date,

text: dataComment.text,

changeMode: false,

textExported: '',

firstRender: true

})

})

},

getComments: function(instance){

var thisEnv = this

instance.get('attraction/comments').then(function(comments){

let dataComments = comments.data

var response = []

dataComments.forEach((comment) => {

comment.changeMode = false,

comment.textExported = '',

comment.firstRender = true

response.push(comment)

})

thisEnv.commentsList = response

})

},

3) Запросы для контроллера страницы достопримечательностей

getAttractionChapters: function(instance){

var thisEnv = this

var result = []

var responseMainContentList = []

instance.get('/attraction/chapters').then(function(responseMainContent){

responseMainContentList = responseMainContent.data

// Обработка данных для каждого раздела

responseMainContentList.forEach((maincontent) => {

maincontent.photoPath = maincontent.photoPath.split("@/assets")[1]

result.push(maincontent)

})

thisEnv.listChapters = result

for (const [index, chapter] of thisEnv.listChapters.entries()) {

thisEnv.getListNotes(instance, chapter.text, index)

}

})

},

getListNotes: function(instance, nameChapter, index){

var thisEnv = this

instance.get('/attraction/note?attractionname='+nameChapter).then(function(responseNote){

var dataResponseNote = responseNote.data

dataResponseNote.photoPath = dataResponseNote.photoPath.split("@/assets")[1]

thisEnv['listTwoNotes'+(Math.floor(index/2) + 1)].push(dataResponseNote)

})

},

getPhotos: function(instance){

var thisEnv = this

instance.get('attraction/photo').then(function(photos){

let dataPhoto = photos.data

for (var index=0; index<dataPhoto.length; index++){

let photoObject = {}

photoObject.photoPath = dataPhoto[index].photo.split("@/assets")[1]

photoObject.activeForFirst = ""

if (index % 2 == 0){

photoObject.activeForFirst = "carousel-item active"

} else {

photoObject.activeForFirst = "carousel-item"

}

thisEnv['listPhotosPath'+(Math.floor(index/2) + 1)].push(photoObject)

}

})

},

4) Запросы для контроллера страницы города

setPageName: function(slugname){

let thisEnv = this

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

if (slugname !== null){

instance.get('/city/getbyname?engname='+slugname).then(function(response){

document.title = response.data.name

thisEnv.pageName = response.data.name

thisEnv.idCurrentCity = response.data.id

})

}

},

getStartImagePath: function(instance, idCity){

var thisEnv = this

instance.get('/city/mainphoto?idCity='+idCity).then(function(mainphoto){

thisEnv.startImagePath = mainphoto.data.photo.split("@/assets")[1]

})

},

getCityChapters: function(instance, idCity){

var thisEnv = this

var result = []

var responseChapterList = []

instance.get('city/chapters?idCity='+idCity).then(function(responseCity){

responseChapterList = responseCity.data

// Обработка данных для каждого раздела

responseChapterList.forEach((chapter) => {

chapter.photoPath = chapter.photoPath.split("@/assets")[1]

result.push(chapter)

})

thisEnv.listChapters = result

thisEnv.setCityNameChapters(instance)

thisEnv.getNotes(instance, idCity)

})

},

setCityNameChapters: function(instance){

for (const [index, chapter] of this.listChapters.entries()) {

this['nameSection'+(index+1)] = chapter.text

}

},

getNotes: function(instance, idCity){

var thisEnv = this

instance.get('city/getnotes?idCity='+idCity).then(function(chapter){

var dataResponseNote = chapter.data

// Обработка данных для каждого раздела

for (const [index, chapter] of dataResponseNote.entries()) {

var allNotes = []

chapter.forEach((note) => {

note.photoPath = note.photoPath.split("@/assets")[1]

allNotes.push(note)

})

thisEnv['listFourNotes'+(index+1)] = allNotes

}

})

},

getPhotos: function(instance, idCity){

var thisEnv = this

instance.get('city/photo?idCity='+idCity).then(function(photos){

let dataPhoto = photos.data

for (var i = 0; i < dataPhoto.length; i++) {

thisEnv['photo'+(i+1)] = dataPhoto[i].photo.split("@/assets")[1]

}

}

)}

5) Запросы для контроллера домашней страницы

getMainContentChapters: function(instance){

var thisEnv = this

var result = []

var responseMainContentList = []

instance.get('/maincontent/chapters').then(function(responseMainContent){

responseMainContentList = responseMainContent.data

// Обработка данных для каждого раздела

responseMainContentList.forEach((maincontent) => {

maincontent.photoPath = maincontent.photoPath.split("@/assets")[1]

result.push(maincontent)

})

thisEnv.listChapters = result

})

},

getListContents: function(instance){

var thisEnv = this

var result = []

var responseListContents = []

instance.get('/maincontent/photo').then(function(startPhotos){

responseListContents = startPhotos.data

responseListContents.forEach((photoWithLink) => {

if (photoWithLink.text != ""){

photoWithLink.photoPath = photoWithLink.photo.split("@/assets")[1]

delete photoWithLink.photo

if (responseListContents.indexOf(photoWithLink) == 0){

photoWithLink.activeForFirst = "carousel-item active"

} else{

photoWithLink.activeForFirst = "carousel-item"

}

result.push(photoWithLink)

}

})

thisEnv.listContents = result

})

},

getDescriptions: function(instance){

var thisEnv = this

instance.get('maincontent/description').then(function(description){

let descriptionData = description.data.text.split('(new-page)')

thisEnv.descriptionFirst = descriptionData[0]

thisEnv.descriptionSecond = descriptionData[1]

instance.get('maincontent/photo').then(function(photos){

var responseListPhotos = photos.data

responseListPhotos = responseListPhotos.filter(function(photo){

return photo.text == ""

})

thisEnv.basePicturePath = responseListPhotos[0].photo.split("@/assets")[1]

})

})

},

loadPictures: function(instance){

var thisEnv = this

var responseListPhotos = []

var result = []

instance.get('maincontent/photo').then(function(photos){

responseListPhotos = photos.data

responseListPhotos = responseListPhotos.filter(function(photo){

return photo.text == ""

})

responseListPhotos.shift()

responseListPhotos.forEach((photo) => {

photo.photoPath = photo.photo.split("@/assets")[1]

delete photo.photo

if (responseListPhotos.indexOf(photo) == 0){

photo.activeForFirst = "carousel-item active"

} else {

photo.activeForFirst = "carousel-item"

}

result.push(photo)

})

thisEnv.listPhotosPath = result

})

},

getListNotes: function(instance){

var thisEnv = this

var responseListNotes = []

instance.get('maincontent/notes').then(function(notes){

let dataNotes = notes.data

thisEnv.listFourNotes = dataNotes

})

}

6) Запросы для контроллера страницы вариантов проживания

getStartImagePath: function(instance){

var thisEnv = this

instance.get('/housing/mainphoto').then(function(mainphoto){

thisEnv.startImagePath = mainphoto.data.photo.split("@/assets")[1]

})

},

getHousingChapters: function(instance){

var thisEnv = this

var result = []

var responseChapterList = []

instance.get('/housing/chapters').then(function(responseHousing){

responseChapterList = responseHousing.data

// Обработка данных для каждого раздела

responseChapterList.forEach((chapter) => {

chapter.photoPath = chapter.photoPath.split("@/assets")[1]

result.push(chapter)

})

thisEnv.listChapters = result

})

},

getListNotes: function(instance, nameChapter){

var thisEnv = this

var result = []

instance.get('/housing/notes?housing='+nameChapter).then(function(responseNote){

var dataResponseNote = responseNote.data

// Обработка данных для каждого раздела

dataResponseNote.forEach((note) => {

note.photoPath = note.photoPath.split("@/assets")[1]

result.push(note)

})

//console.log(result)

switch (nameChapter){

case 'hotels':

thisEnv.listTwoNotes1 = result

break;

case 'kempings':

thisEnv.listTwoNotes2 = result

break;

case 'guest houses':

thisEnv.listTwoNotes3 = result

break;

}

})

},

getPhotos: function(instance){

var thisEnv = this

instance.get('housing/photos').then(function(photos){

let dataPhoto = photos.data

thisEnv.photo1 = dataPhoto[1].photo.split("@/assets")[1]

thisEnv.photo2 = dataPhoto[2].photo.split("@/assets")[1]

thisEnv.photo3 = dataPhoto[3].photo.split("@/assets")[1]

})

}

7) Запросы для контроллера страницы кухни

created(){

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

instance.get('/city/names').then((response) => this.listCity = response.data)

this.getStartImagePath(instance)

this.getKitchenChapters(instance)

this.getListNotes(instance, 'streetfood')

this.getListNotes(instance, 'cafe')

this.getListNotes(instance, 'bars')

this.getPhotos(instance)

},

methods:{

setPageName: function(pagename){},

getStartImagePath: function(instance){

var thisEnv = this

instance.get('/housing/mainphoto').then(function(mainphoto){

thisEnv.startImagePath = mainphoto.data.photo.split("@/assets")[1]

})

},

getKitchenChapters: function(instance){

var thisEnv = this

var result = []

var responseChapterList = []

instance.get('/kitchen/chapters').then(function(responseKitchen){

responseChapterList = responseKitchen.data

// Обработка данных для каждого раздела

responseChapterList.forEach((chapter) => {

chapter.photoPath = chapter.photoPath.split("@/assets")[1]

result.push(chapter)

})

thisEnv.listChapters = result

})

},

getListNotes: function(instance, nameChapter){

var thisEnv = this

var result = []

instance.get('/kitchen/notes?kitchen='+nameChapter).then(function(responseNote){

var dataResponseNote = responseNote.data

// Обработка данных для каждого раздела

dataResponseNote.forEach((note) => {

note.photoPath = note.photoPath.split("@/assets")[1]

result.push(note)

})

switch (nameChapter){

case 'streetfood':

thisEnv.listFourNotes1 = result

break;

case 'cafe':

thisEnv.listFourNotes2 = result

break;

case 'bars':

thisEnv.listFourNotes3 = result

break;

}

})

},

getPhotos: function(instance){

var thisEnv = this

instance.get('kitchen/photos').then(function(photos){

let dataPhoto = photos.data

thisEnv.photo1 = dataPhoto[5].photo.split("@/assets")[1]

thisEnv.photo2 = dataPhoto[4].photo.split("@/assets")[1]

thisEnv.photo3 = dataPhoto[6].photo.split("@/assets")[1]

thisEnv.photo4 = dataPhoto[7].photo.split("@/assets")[1]

thisEnv.photo5 = dataPhoto[8].photo.split("@/assets")[1]

})

}

8) Запросы для контроллера страницы вариантов передвижения

created(){

const instance = Axios.create({

baseURL: 'http://127.0.0.1:1199/api/v1'

})

instance.get('/city/names').then((response) => this.listCity = response.data)

this.getStartImagePath(instance)

this.getTripChapters(instance)

this.getListNotes(instance, 'plane')

this.getListNotes(instance, 'ferry')

this.getListNotes(instance, 'loanAuto')

this.getPhotos(instance)

},

methods:{

setPageName: function(pagename){},

getStartImagePath: function(instance){

var thisEnv = this

instance.get('/trip/mainphoto').then(function(mainphoto){

thisEnv.startImagePath = mainphoto.data.photo.split("@/assets")[1]

})

},

getTripChapters: function(instance){

var thisEnv = this

var result = []

var responseChapterList = []

instance.get('/trip/chapters').then(function(responseTrip){

responseChapterList = responseTrip.data

// Обработка данных для каждого раздела

responseChapterList.forEach((chapter) => {

chapter.photoPath = chapter.photoPath.split("@/assets")[1]

result.push(chapter)

})

thisEnv.listChapters = result

})

},

getListNotes: function(instance, nameChapter){

var thisEnv = this

var result = []

instance.get('/trip/notes?trip='+nameChapter).then(function(responseNote){

var dataResponseNote = responseNote.data

// Обработка данных для каждого раздела

dataResponseNote.forEach((note) => {

note.photoPath = note.photoPath.split("@/assets")[1]

result.push(note)

})

switch (nameChapter){

case 'plane':

thisEnv.listTwoNotes1 = result

break;

case 'ferry':

thisEnv.listTwoNotes2 = result

break;

case 'loanAuto':

thisEnv.listTwoNotes3 = result

break;

}

})

},

getPhotos: function(instance){

var thisEnv = this

instance.get('kitchen/photos').then(function(photos){

let dataPhoto = photos.data

thisEnv.photo1 = dataPhoto[1].photo.split("@/assets")[1]

thisEnv.photo2 = dataPhoto[4].photo.split("@/assets")[1]

thisEnv.photo3 = dataPhoto[3].photo.split("@/assets")[1]

})

}

Пример взаимодействия фронтенда с REST API, запущенного в Docker.

Graphical user interface, application, website

Description automatically generated