ATUSS- Odsek Visoka škola strukovnih studija za informacione i komunikacione tehnologije

Dokumentacija

**Praktikum korisnički interfejs**

Student: Alen Aslani 126/21

Uvod

# Korišćeni programski jezici

Za izradu aplikacije korišćene su sledeće tehnologije:

* HTML5
* CSS3
* Bootstrap 4
* Ajax
* Laravel
* Vue 3

# Opis funkcionalnosti

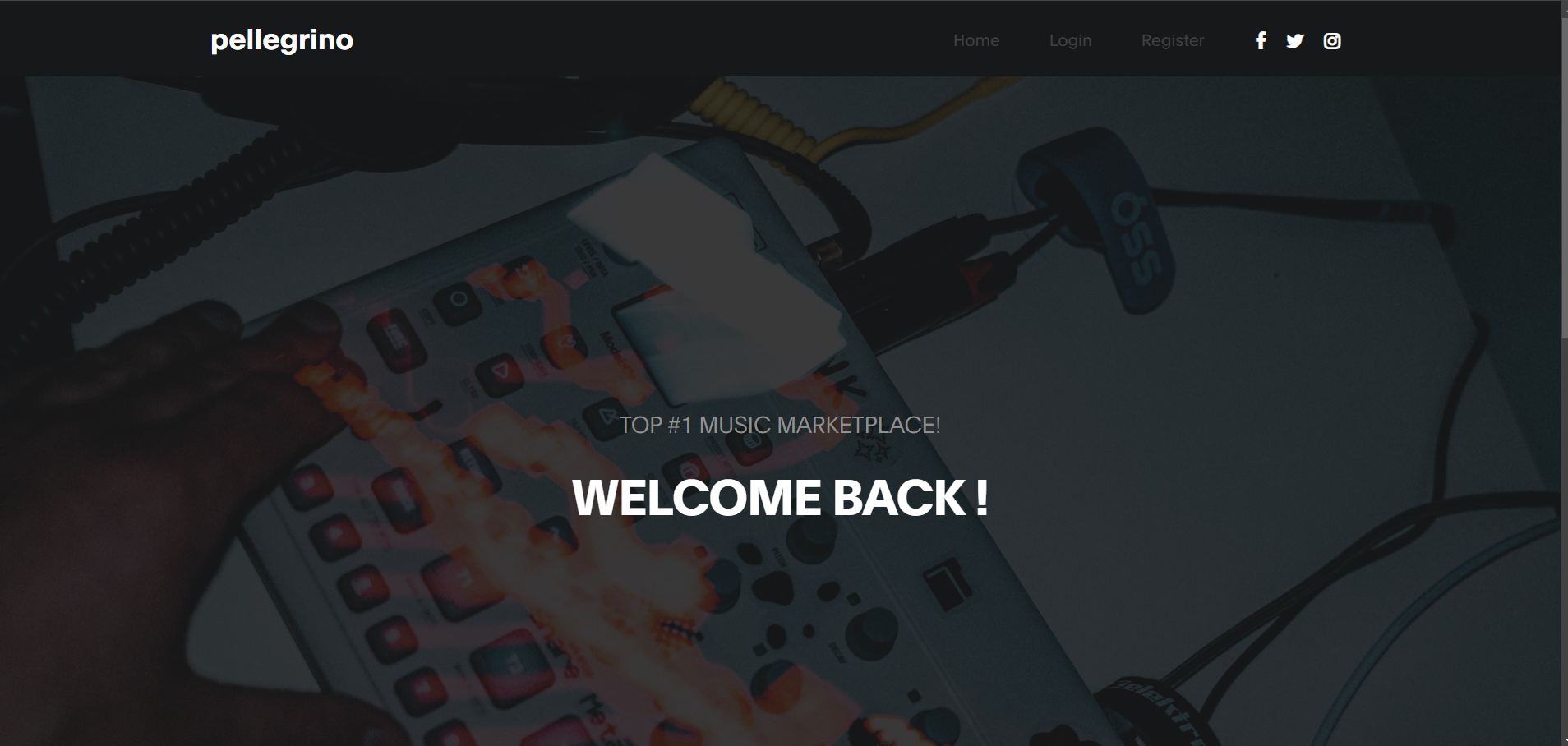
* Dinamički meni I futer
* Dinamički ispisane pesme
* Login
* Registracija
* Artist panel
* Unos, brisanje I editovanje pesama
* Mogucnost lajkovanja pesme
* Pretraga
* Filtriranje sadržaja delova sajta (User, Artist)

# Template

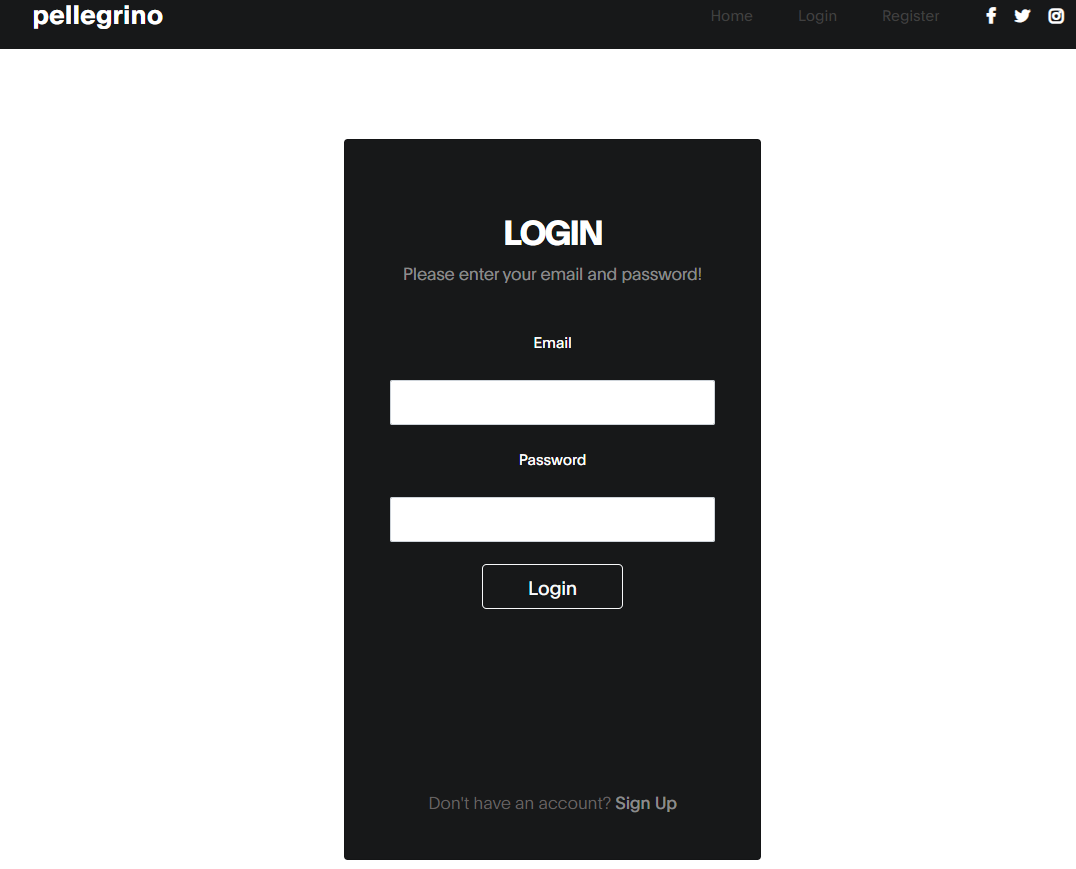
Za izradu sajta je korišćen template(<https://www.tooplate.com/view/2119-gymso-fitness>).

Organizacija

# /home

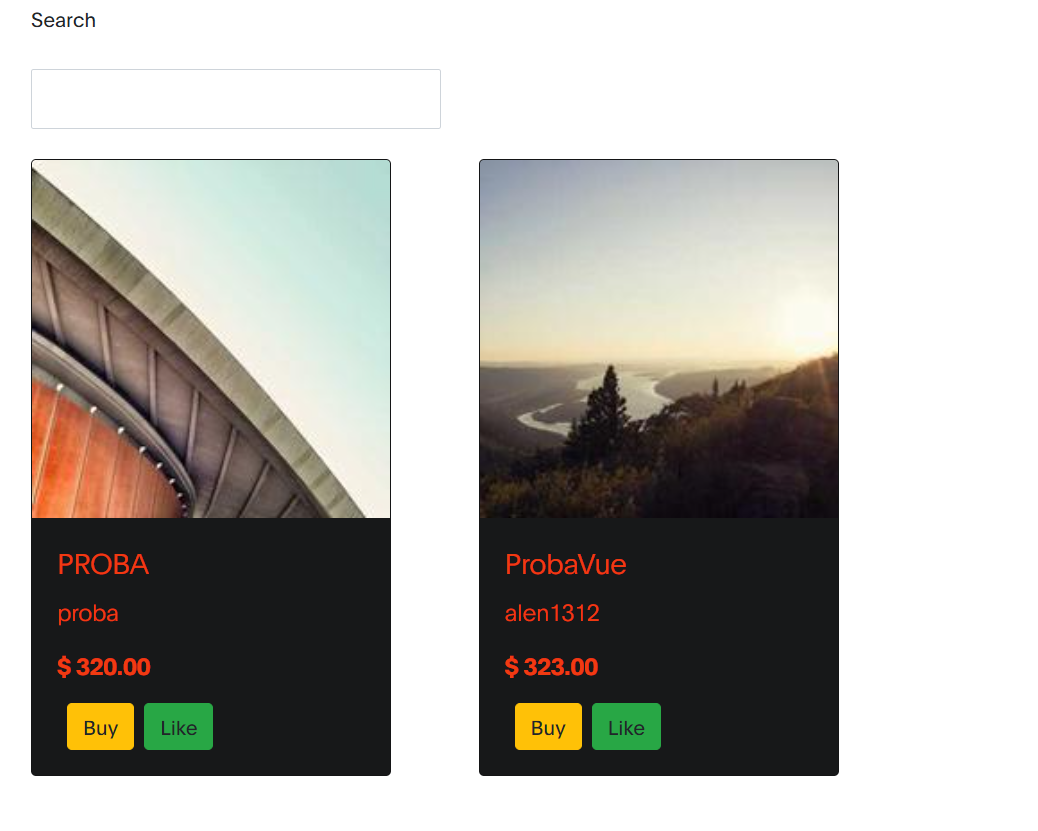


# /login

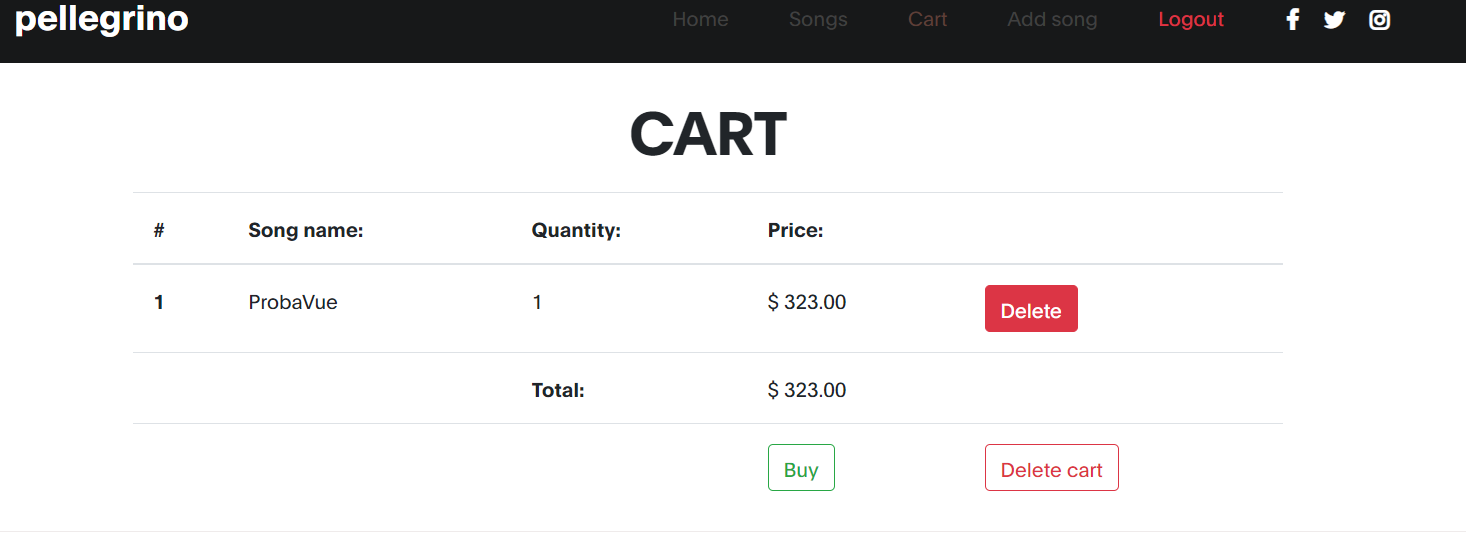


# /registerC:\Users\Alen\AppData\Local\Microsoft\Windows\INetCache\Content.Word\register.png

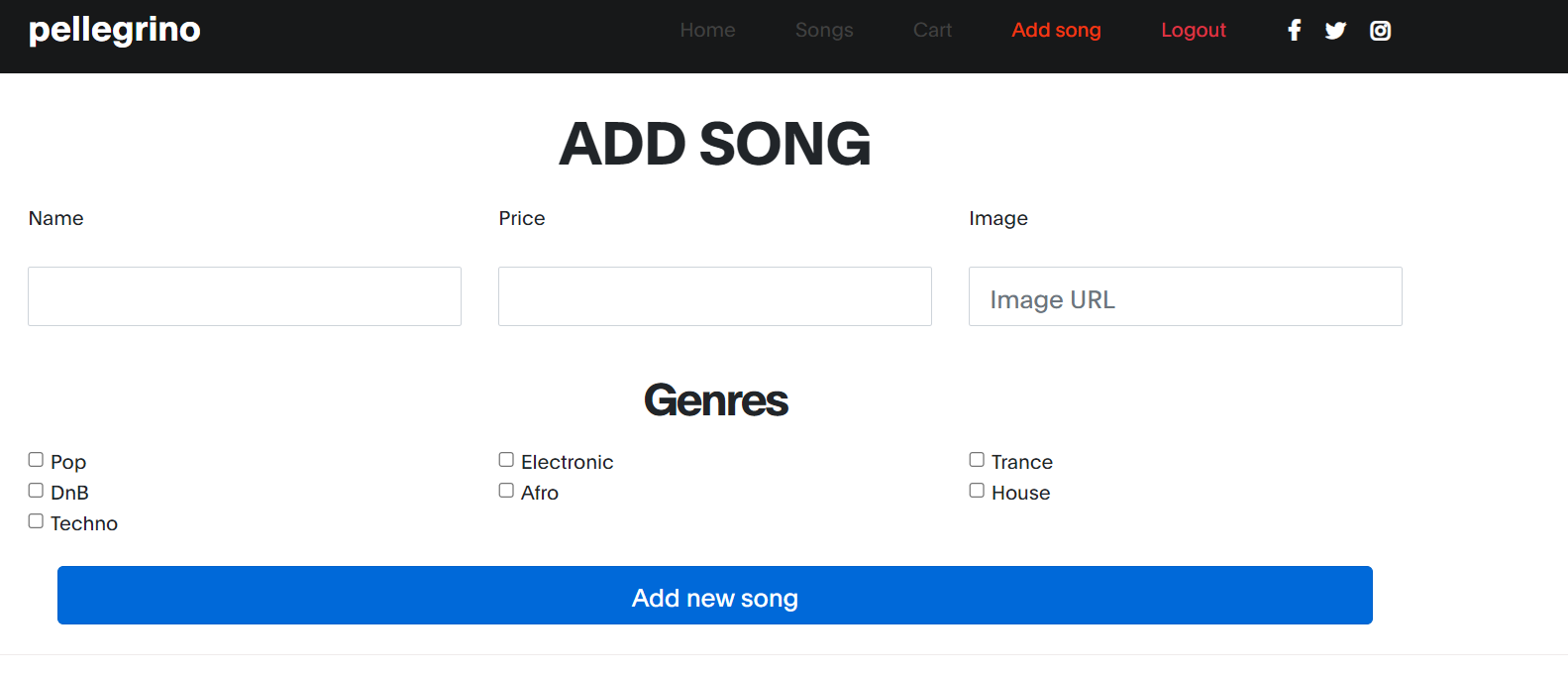
# /songs



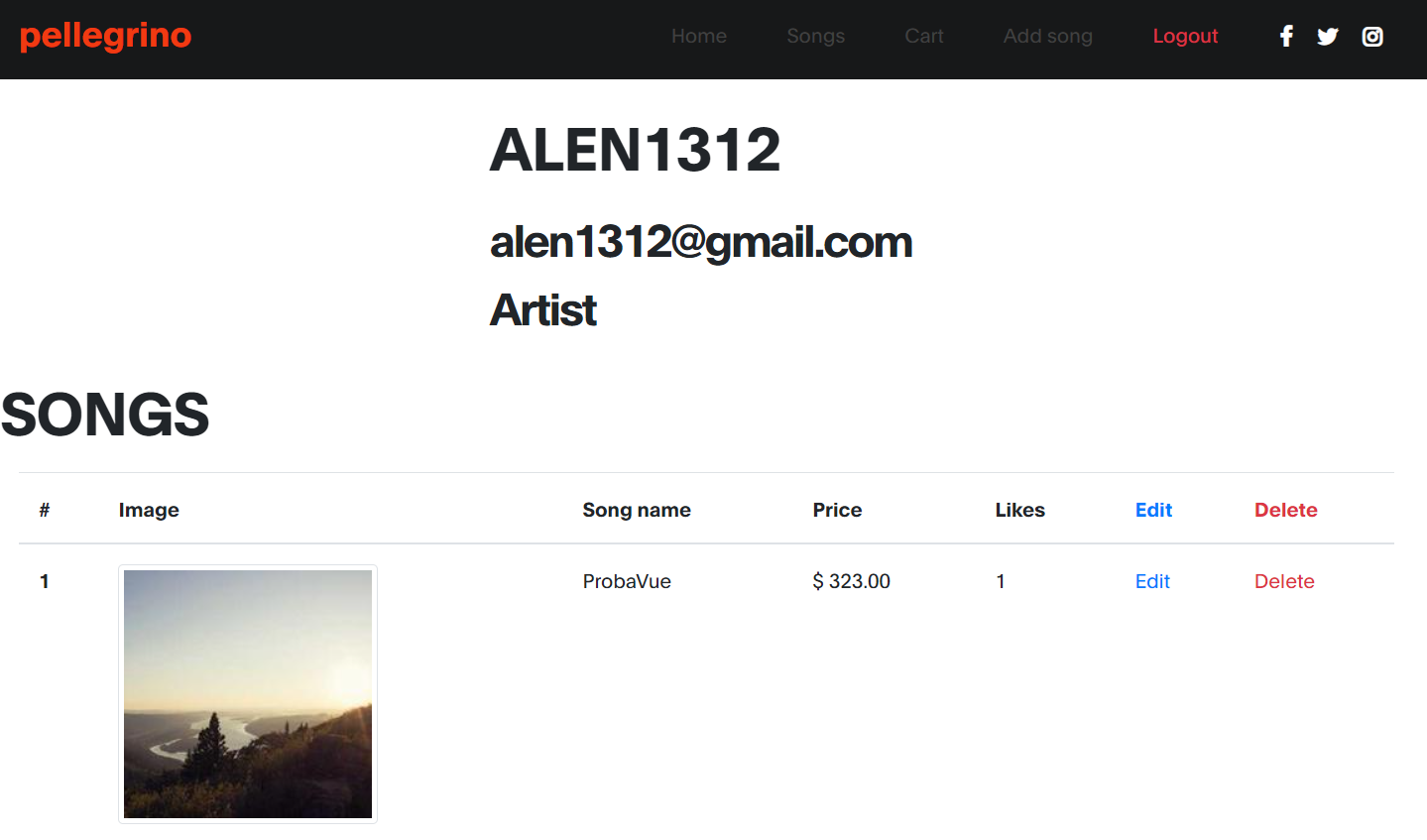
# /cart



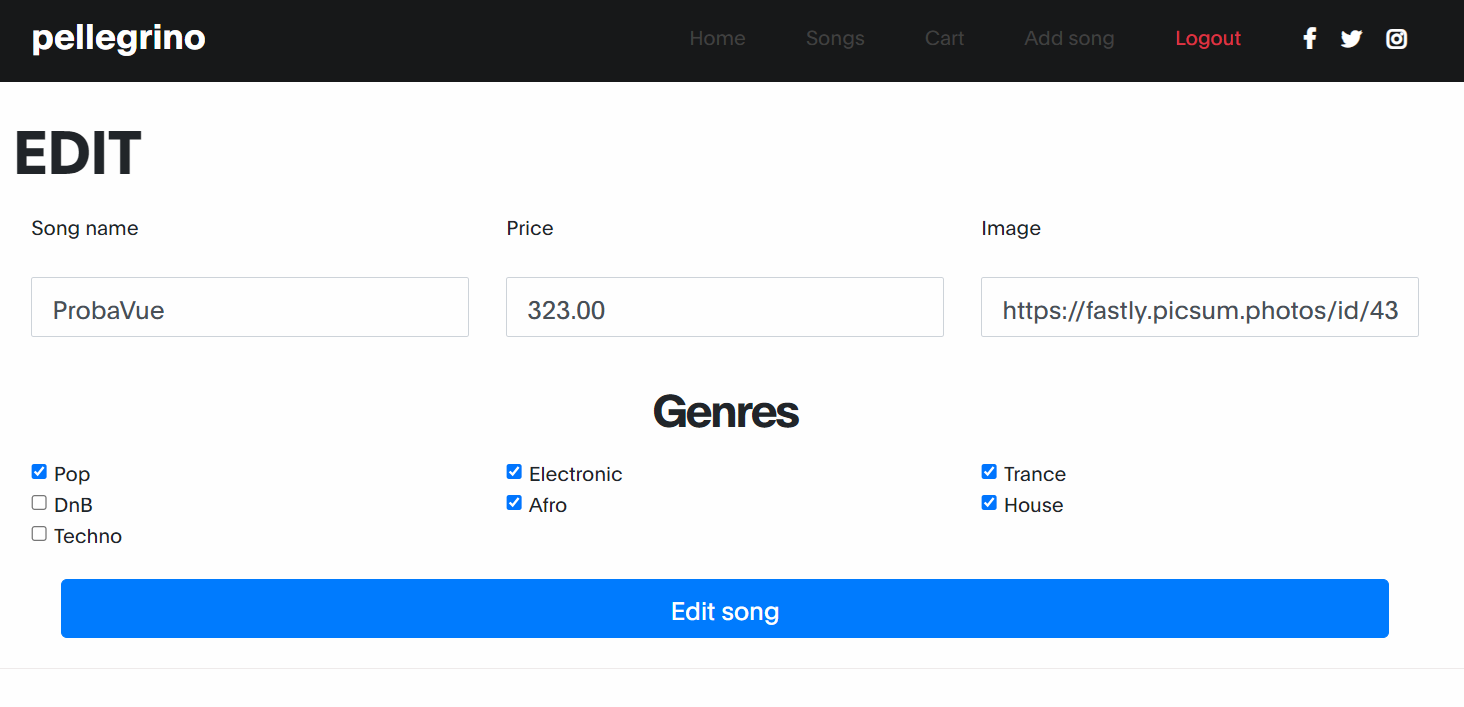
# /addSong



# /profile



# /edit



Kod

# Vue

## App.vue

<template>

  <the-header></the-header>

  <router-view></router-view>

  <the-footer></the-footer>

</template>

<script>

import TheHeader from './components/layout/TheHeader.vue';

import TheFooter from './components/layout/TheFooter.vue';

export default {

  components:{TheFooter,TheHeader},

  async created(){

    await this.$store.dispatch("songs/allGenres");

    await this.$store.dispatch("songs/allSongs");

  }

}

</script>

## Index.js-router

import { createRouter, createWebHistory } from 'vue-router'

import HomeView from '../views/TheHome.vue'

import TheContact from '../views/TheContact.vue'

import TheSchedule from '@/views/TheSchedule.vue'

import TheLogin from '@/views/TheLogin.vue'

import TheRegister from '@/views/TheRegister.vue'

import TheSongs from '@/views/TheSongs.vue'

import InsertSong from '@/views/InsertSong.vue'

import ProbaValidacija from '@/views/ProbaValidacija.vue'

import EditSong from '@/views/EditSong.vue'

import TheProfile from '@/views/TheProfile.vue'

import TheCart from '@/views/TheCart.vue'

import store from '../store/index';

const routes = [

  {

    path: '/',

    name: 'home',

    component: HomeView

  },

  {

    path: '/validacija',

    name: 'validacija',

    component: ProbaValidacija

  },

  {

    path: '/contact',

    name: 'contact',

    component: TheContact

  },

  {

    path: '/schedule',

    name: 'schedule',

    component: TheSchedule

  },

  {

    path: '/login',

    name: 'login',

    component: TheLogin,

    meta:{

      isUnauth:true

    }

  },

  {

    path: '/register',

    name: 'register',

    component: TheRegister,

    meta:{

      isUnauth:true

    }

  },

  {

    path: '/songs',

    name: 'songs',

    component: TheSongs,

    meta:{

      isAuth:true

    }

  },

  {

    path: '/addSong',

    name: 'addsong',

    component: InsertSong,

    beforeEnter: (to, from, next) => {

      const userRole = store.getters['auth/getRole'];

      console.log(userRole);

      if (userRole == 'Artist') {

        next();

      } else {

        next({ name: 'home' });

      }

    },

    meta:{

      isAuth:true

    }

  },

  {

    path: '/profile/:id',

    name: 'profile',

    component: TheProfile,

    meta:{

      isAuth:true

    }

  },

  {

    path: '/edit/:id',

    name: 'edit',

    component: EditSong,

    meta:{

      isAuth:true

    }

  },

  {

    path: '/cart',

    name: 'cart',

    component: TheCart,

    meta:{

      isAuth:true

    }

  }

]

const router = createRouter({

  history: createWebHistory(process.env.BASE\_URL),

  routes

})

// router.beforeEach(function(to,\_,next){

//   if(to.meta.isAuth && !store.getters['auth/getToken']){

//     return next('/login')

//   }else if(to.meta.isUnauth && store.getters['auth/getToken']){

//     return next('/')

//   }

//   return next();

// })

router.beforeEach((to,\_,next)=>{

  const parseToken = (token) => {

    try{

      return JSON.parse(atob(token.split('.')[1]));

    }catch(er){

      return null;

    }

  }

  let token = parseToken(store.getters["auth/getToken"]);

  if(token != null){

    token = token.exp;

  }

  const date = new Date().getTime() / 1000;

  if(token && date > token){

    localStorage.removeItem("token");

    return next({name:"login"});

  }else{

    if(to.meta.isAuth && store.getters['auth/getToken']){

      return next();

    }else if(to.meta.isAuth && !store.getters['auth/getToken']){

      return next({name:"login"});

    }else{

      return next();

    }

  }

})

export default router

## TheFooter.vue

<template>

 <!-- FOOTER -->

 <footer class="site-footer">

          <div class="container">

               <div class="row">

                    <div class="ml-auto col-lg-4 col-md-5">

                        <p class="copyright-text">Copyright &copy; 2020 Gymso Fitness Co.

                        </p>

                    </div>

                    <div class="d-flex justify-content-center mx-auto col-lg-5 col-md-7 col-12">

                        <p class="mr-4">

                            <i class="fa fa-envelope-o mr-1"></i>

                            <a href="#">hello@company.co</a>

                        </p>

                        <p><i class="fa fa-phone mr-1"></i> 010-020-0840</p>

                    </div>

               </div>

          </div>

     </footer>

</template>

## TheHeader.vue

<template>

     <!-- MENU BAR -->

     <nav class="navbar navbar-expand-lg">

        <div class="container">

            <router-link class="navbar-brand" to="/">pellegrino</router-link>

            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false"

                aria-label="Toggle navigation">

                <span class="navbar-toggler-icon"></span>

            </button>

            <div class="collapse navbar-collapse" id="navbarNav">

              <the-nav></the-nav>

                <!-- <ul class="navbar-nav ml-lg-auto">

                    <li class="nav-item">

                        <a href="#home" class="nav-link smoothScroll">Home</a>

                    </li>

                    <li class="nav-item">

                        <a href="#about" class="nav-link smoothScroll">About Us</a>

                    </li>

                    <li class="nav-item">

                        <a href="#class" class="nav-link smoothScroll">Classes</a>

                    </li>

                    <li class="nav-item">

                        <a href="#schedule" class="nav-link smoothScroll">Schedules</a>

                    </li>

                    <li class="nav-item">

                        <a href="#contact" class="nav-link smoothScroll">Contact</a>

                    </li>

                </ul> -->

                <ul class="social-icon ml-lg-3">

                    <li><a href="https://fb.com/tooplate" class="fa fa-facebook"></a></li>

                    <li><a href="#" class="fa fa-twitter"></a></li>

                    <li><a href="#" class="fa fa-instagram"></a></li>

                </ul>

            </div>

        </div>

    </nav>

</template>

<script>

import TheNav from './TheNav.vue';

export default {

    components:{TheNav}

}

</script>

## TheNav.vue

<template>

  <ul class="navbar-nav ml-lg-auto">

                    <router-link class="nav-link" to="/">Home</router-link>

                    <router-link class="nav-link" v-if="!islog" to="/login">Login</router-link>

                    <router-link class="nav-link" v-if="!islog" to="/register">Register</router-link>

                    <router-link class="nav-link" v-if="islog" to="/songs">Songs</router-link>

                    <router-link class="nav-link" v-if="islog" to="/cart">Cart</router-link>

                    <router-link class="nav-link" v-if="isArtist" to="/addSong">Add song</router-link>

                    <a @click="logout" class="nav-link text-danger" v-if="islog">Logout</a>

  </ul>

</template>

<script>

import { mapGetters } from "vuex";

export default {

  computed:{

    islog(){

      var e = this['auth/getAa'];

    return e;

    },

    isArtist() {

      // console.log(this['auth/getRole'])

        return this['auth/getRole'] == 'Artist';

    },

    ...mapGetters(

      [

        'auth/getAa',

        'auth/getRole'

      ]

    )

  },

  methods:{

    logout(){

      this.$store.dispatch("auth/logout")

      this.$router.push("/login")

    }

  },

}

</script>

## TextInput.vue

<template>

    <div data-mdb-input-init class="form-outline form-white mb-4">

        <label class="form-label" :for="id">{{ label }}</label>

        <input :name="id" :value="modelValue" @input="$emit('update:modelValue',$event.target.value)" :type="type" :id="id" class="form-control form-control-lg" :placeholder="placeholder" />

        <p v-if="error" class="alert p-1 mt-2 alert-danger">{{ error }}</p>

    </div>

</template>

<script>

export default {

    props:{

        modelValue:{

            type:String,

            required:true

        },

        label:{

            type:String,

            default:'Label'

        },

        type:{

            type:String,

            default:'text'

        },

        id:{

            type:String

        },

        error:{

            type:String,

            default:""

        },

        placeholder:{

            type:String,

            default:""

        }

    },

    emits:['update:modelValue'],

    computed: {

            model: {

              get() {

                return this.modelValue;

              },

              set(value) {

                this.$emit('update:modelValue', value);

              }

            }

    }

}

</script>

## TheHome.vue

<template>

   <!-- HERO -->

   <section class="hero d-flex flex-column justify-content-center align-items-center" id="home">

<div class="bg-overlay"></div>

   <div class="container">

        <div class="row">

             <div class="col-lg-8 col-md-10 mx-auto col-12">

                  <div class="hero-text mt-5 text-center">

                        <h6>TOP #1 MUSIC MARKETPLACE!</h6>

                        <h1 class="text-white" >Welcome back <router-link v-if="username" :to="'/profile/' + id">{{ username }}</router-link> !</h1>

                  </div>

             </div>

        </div>

   </div>

</section>

<!-- CONTACT -->

<section class="contact section" id="contact">

          <div class="container">

               <div class="row">

                    <div class="ml-auto col-lg-5 col-md-6 col-12">

                        <h2 class="mb-4 pb-2">Feel free to ask anything</h2>

                        <form action="#" method="post" class="contact-form webform"  role="form">

                            <input type="text" class="form-control" name="cf-name" placeholder="Name">

                            <input type="email" class="form-control" name="cf-email" placeholder="Email">

                            <textarea class="form-control" rows="5" name="cf-message" placeholder="Message"></textarea>

                            <button type="submit" class="form-control" id="submit-button" name="submit">Send Message</button>

                        </form>

                    </div>

                    <div class="mx-auto mt-4 mt-lg-0 mt-md-0 col-lg-5 col-md-6 col-12">

                        <h2 class="mb-4" >Where you can <span>find us</span></h2>

                        <p ><i class="fa fa-map-marker mr-1"></i> 120-240 Rio de Janeiro - State of Rio de Janeiro, Brazil</p>

                        <div class="google-map">

                            <iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2830.358862542557!2d20.479543886239195!3d44.81425329647885!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x475a7a9609031735%3A0xc17ed71f59ac4725!2z0JLQuNGB0L7QutCwIElDVCDRiNC60L7Qu9Cw!5e0!3m2!1ssr!2srs!4v1724795939148!5m2!1ssr!2srs" width="1920" height="250" frameborder="0" style="border:0;"></iframe>

                        </div>

                    </div>

               </div>

          </div>

     </section>

</template>

<script>

import { mapGetters } from 'vuex';

export default {

    name:"TheHome",

    data(){

     return{

          username:"",

          id:null

     }

    },

    computed:{

     ...mapGetters(['auth/getTokenInfo'])

    },

    created(){

     if(this['auth/getTokenInfo']){

          this.username = this['auth/getTokenInfo'].username;

          this.id = this['auth/getTokenInfo'].id;

     }

    }

}

</script>

## TheRegister.vue

<template>

    <section class="rounded mt-5">

  <div class="container py-5 h-100">

    <div class="row d-flex justify-content-center align-items-center h-100">

      <div class="col-12 col-md-8 col-lg-6 col-xl-5">

        <div class="card text-white">

          <div class="card-body p-5 text-center">

            <!-- Pocetak Forma -->

            <div class="mb-md-5 mt-md-4 pb-5">

              <h2 class="fw-bold mb-2 text-uppercase">Register</h2>

              <text-input @keyup="validate('username')" v-model="regadata.username.value" label="Username" id="username"/>

              <p class="text-danger" v-if="regadata.username.errormsg">{{ regadata.username.errormsg }}</p>

              <text-input @keyup="validate('email')" v-model="regadata.email.value" label="Email" id="email"/>

              <p class="text-danger" v-if="regadata.email.errormsg">{{ regadata.email.errormsg }}</p>

              <text-input @keyup="validate('password')" v-model="regadata.password.value" label="Password" id="password" type="password"/>

              <p class="text-danger" v-if="regadata.password.errormsg">{{ regadata.password.errormsg }}</p>

              <text-input @keyup="validate('confpassword')" v-model="regadata.confpassword.value" label="Confirm password" id="confpassword" type="password"/>

              <p class="text-danger" v-if="regadata.confpassword.errormsg">{{ regadata.confpassword.errormsg }}</p>

              <button @click.prevent="register" data-mdb-button-init data-mdb-ripple-init class="btn btn-outline-light btn-lg px-5" type="submit">Login</button>

            </div>

            <!-- Kraj Forma -->

            <p v-if="infomsg" :class=" msg ? 'text-danger': 'text-success' ">{{ infomsg }}</p>

            <div>

              <p class="mb-0">You already have an account? <router-link to="/login" class="text-white-50 fw-bold">Login</router-link>

              </p>

            </div>

          </div>

        </div>

      </div>

    </div>

  </div>

</section>

</template>

<script>

import TextInput from '@/components/inputfields/TextInput.vue'

export default {

  components: { TextInput },

  name:"TheRegister",

  data(){

    return{

      regadata:{

        username:{

          value:"",

          errormsg:""

        },

        email:{

          value:"",

          errormsg:""

        },

        password:{

          value:"",

          errormsg:""

        },

        confpassword:{

          value:"",

          errormsg:""

        },

      },

      error:false,

      msg:false,

      infomsg:""

    }

  },

  methods:{

    async register(){

      if(!this.error){

        const dataforsend = {

        email:this.regadata.email.value,

        username:this.regadata.username.value,

        password:this.regadata.password.value

        }

        try{

          await this.$store.dispatch("auth/register",dataforsend);

          setTimeout(() => {

                    this.$router.push({ name: 'login' });

                }, 2000);

          this.msg = false;

          this.infomsg = "Registration successful, welcome!";

        }catch(error){

          if(error.status == 400){

            this.regadata.email.errormsg = error.response.data.errors['email'];

            this.regadata.username.errormsg = error.response.data.errors['username'];

          }else{

            this.msg = true;

            this.infomsg = "Please try later."

          }

        }

      }

    },

    validate(type){

      switch(type){

        case "username":

          if(this.regadata.username.value.length < 2 || this.regadata.username.value.length > 50){

            this.error = true;

            this.regadata.username.errormsg = "The username must be between 2 and 50 characters long."

          }else{

            this.error = false;

            this.regadata.username.errormsg = ""

          }

          break;

        case "email":

          //eslint-disable-next-line

          if(!/^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$/.test(this.regadata.email.value)){

            this.error = true;

            this.regadata.email.errormsg = "The email address entered is not in the correct format!"

          }else{

            this.error = false;

            this.regadata.email.errormsg = ""

          }

          break;

        case "password":

          //eslint-disable-next-line

          if(!/^(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[a-zA-Z]).{8,}$/.test(this.regadata.password.value)){

            this.error = true;

            this.regadata.password.errormsg = "The password must contain at least one uppercase letter, one lowercase letter, one number, and be more than 8 characters long.";

          }else{

            this.error = false;

            this.regadata.password.errormsg = "";

          }

          break;

          case "confpassword":

            if(this.regadata.confpassword.value != this.regadata.password.value){

              this.error = true;

              this.regadata.confpassword.errormsg = "The passwords do not match.";

            }else{

              this.error = false;

              this.regadata.confpassword.errormsg = "";

            }

      }

    }

  }

}

</script>

<style scoped>

.card{

    background-color: #171819;

}

</style>

## TheLogin.vue

<template>

<section class="rounded mt-5">

  <div class="container py-5 h-100">

    <div class="row d-flex justify-content-center align-items-center h-100">

      <div class="col-12 col-md-8 col-lg-6 col-xl-5">

        <div class="card text-white">

          <div class="card-body p-5 text-center">

            <!-- Pocetak Forma -->

              <div class="mb-md-5 mt-md-4 pb-5">

              <h2 class="fw-bold mb-2 text-uppercase">Login</h2>

              <p class="text-white-50 mb-5">Please enter your email and password!</p>

              <text-input @keyup="validateEmail"  v-model="loginData.email.value" label="Email" id="email" type="email"/>

              <p class="text-danger" v-if="loginData.email.errormsg">{{ loginData.email.errormsg }}</p>

              <text-input v-model="loginData.password.value" label="Password" id="password" type="password"/>

              <button @click.prevent="login" class="btn btn-outline-light btn-lg px-5" type="submit">Login</button>

                <div class="p-5">

                  <p v-if="infomessage" :class=" msg ? 'text-danger': 'text-success' ">{{ infomessage }}</p>

                </div>

              </div>

            <!-- Kraj Forma -->

            <div>

              <p class="mb-0">Don't have an account? <router-link to="/register" class="text-white-50 fw-bold">Sign Up</router-link>

              </p>

            </div>

          </div>

        </div>

      </div>

    </div>

  </div>

</section>

</template>

<script>

import TextInput from '../components/inputfields/TextInput.vue'

export default {

    components:{TextInput},

    name:"TheLogin",

    data(){

      return{

        loginData:{

          email:{

            value:"",

            errormsg:""

          },

          password:{

            value:"",

            errormsg:""

          }

        },

        msg:false,

      infomessage:"",

      error:false

      }

    },

    methods:{

      async login(){

        if(!this.error){

          const dataforsend = {

          email : this.loginData.email.value,

          password : this.loginData.password.value

        };

          try{

          await this.$store.dispatch("auth/login", dataforsend);

          this.msg = false;

          this.infomessage = "Login successful, welcome back!"

          setTimeout(()=> this.$router.push("/"), 2000);

          }catch(err){

              this.msg = true;

              this.infomessage = "The email or password you entered is incorrect, please try again.";

          }

        }

      },

      validateEmail(){

         //eslint-disable-next-line

         if(!/^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$/.test(this.loginData.email.value)){

          this.error = true;

          this.loginData.email.errormsg = "The email address entered is not in the correct format!"

        }else{

          this.error = false;

          this.loginData.email.errormsg = ""

        }

      }

    }

}

</script>

<style scoped>

.card{

    background-color: #171819;

}

</style>

## TheSongs.vue

<template>

    <div class="container p-5">

        <h1 class="text-center">Subscriptions</h1>

        <div class="row">

            <div class="col-lg-4">

                <text-input v-model="keyword" label="Search" id="search"/>

            </div>

        </div>

        <div class="row">

            <the-sub-info

            v-bind="song"

            v-for="song in filterData"

            :key="song.id"

            :img="song.image"

            :name="song.name"

            :author="song.username"

            :price="song.price"

            :id="song.id"

            ></the-sub-info>

        </div>

    </div>

</template>

<script>

import TheSubInfo from '@/components/TheSubInfo.vue';

import TextInput from '@/components/inputfields/TextInput.vue';

export default {

    name:"TheSubscriptions",

    components:{TheSubInfo,TextInput},

    data(){

        return{

            songs:[],

            genres:[],

            keyword:"",

            filterData:[],

            user:[]

        }

    },

    created(){

        // await this.$store.dispatch("songs/allGenres");

        this.genres = this.$store.state.songs.genres;

        // await this.$store.dispatch("songs/allSongs");

        this.songs = this.$store.state.songs.songs;

        this.filterData = this.songs;

    },

    watch:{

        keyword(){

            this.filterData = this.songs.filter((x)=>{

                if(x.name.toLowerCase().includes(this.keyword.toLowerCase())){

                    return true;

                }

                return false;

            });

        }

    }

}

</script>

## TheProfile.vue

<template>

    <div class="container">

        <div class="row d-flex justify-content-center">

            <div class="col-lg-4">

                <h1>{{ data.username }}</h1>

                <h2>{{ data.email }}</h2>

                <h2>{{ data.role }}</h2>

            </div>

        </div>

        <div v-if="isArtist" class="row">

            <h1>Songs</h1>

            <div class="col-lg-12">

                <table class="table">

                <thead>

                    <tr>

                    <th scope="col">#</th>

                    <th scope="col">Image</th>

                    <th scope="col">Song name</th>

                    <th scope="col">Price</th>

                    <th scope="col">Likes</th>

                    <th scope="col" class="text-primary">Edit</th>

                    <th scope="col" class="text-danger">Delete</th>

                    </tr>

                </thead>

                <tbody>

                    <tr v-bind="song" v-for="(song, i) in data.songs" :key="i">

                    <th scope="row">{{ i+1 }}</th>

                    <td><img :src="song.image" alt="nesto" class="img-thumbnail"></td>

                    <td>{{ song.name }}</td>

                    <td>$ {{ song.latest\_price.price }}</td>

                    <td>{{ song.liked\_by\_users\_count }}</td>

                    <td><router-link :to="'/edit/' + song.id" class="text-primary">Edit</router-link></td>

                    <td><a @click="deleteSong(song.id)" class="text-danger">Delete</a></td>

                    </tr>

                </tbody>

                </table>

            </div>

        </div>

    </div>

</template>

<script>

import { mapGetters } from 'vuex';

export default {

    name:"TheProfile",

    data(){

        return{

            data:[]

        }

    },

    async created(){

        await this.$store.dispatch("auth/User",this.$store.getters['auth/getUserId']);

        this.data = this.$store.state.auth.userInfo;

    },

    computed:{

        isArtist(){

            return this['auth/getRole'] == 'Artist'

        },

        ...mapGetters([

            'auth/getRole'

        ])

    },

    methods:{

        async deleteSong(id){

            try{

                await this.$store.dispatch('songs/deleteSong', id)

                console.log("bravo");

            }catch(error){

                console.log(error);

            }

        }

    }

}

</script>

## TheCart.vue

<template>

    <div class="container">

        <h1 class="text-center">Cart</h1>

        <div class="row d-flex justify-content-center">

            <div class="col-lg-10">

                <table class="table">

                <thead>

                    <tr>

                    <th scope="col">#</th>

                    <th scope="col">Song name:</th>

                    <th scope="col">Quantity:</th>

                    <th scope="col">Price:</th>

                    <th scope="col"></th>

                    </tr>

                </thead>

                <tbody v-if="cartItems.length != 0">

                    <tr v-for="(item,i) in cartItems" :key="item.song.id">

                    <th scope="row"> {{ i+1 }} </th>

                    <td>{{ item.song.name }}</td>

                    <td>{{ item.qty }}</td>

                    <td>$ {{ (item.qty \* item.song.price).toFixed(2) }}</td>

                    <td><button @click="removeItem(item.song.id)" type="button" class="btn btn-danger">Delete</button></td>

                    </tr>

                    <tr>

                        <td></td>

                        <td></td>

                        <th>Total:</th>

                        <td>$ {{ (totalPrice()).toFixed(2) }}</td>

                        <td></td>

                    </tr>

                    <tr>

                        <td></td>

                        <td></td>

                        <th></th>

                        <td><button @click="buy" type="button" class="btn btn-outline-success">Buy</button></td>

                        <td><button @click="clearCart" type="button" class="btn btn-outline-danger">Delete cart</button></td>

                    </tr>

                </tbody>

                <div v-else class="div">

                    <h1 class="text-center">Your cart is empty.</h1>

                </div>

                </table>

                <div v-if="msg" class="alert alert-success" role="alert">

                {{ msg }}

                </div>

            </div>

        </div>

    </div>

</template>

<script>

import { mapGetters } from 'vuex';

export default {

    name:"TheCart",

    data(){

        return{

            cartItems:[],

            msg:""

        }

    },

    created(){

        this.loadCart();

    },

    computed:{

        ...mapGetters(['auth/getUserId'])

    },

    methods:{

        loadCart(){

            const cart = JSON.parse(localStorage.getItem("cart")) || [];

            const songss = this.$store.getters["songs/getSongs"];

            console.log(cart);

            console.log(songss);

            this.cartItems = cart.map(x => {

                const song = songss.find(song => song.id == x.id);

                return {

                    song: song,

                    qty: x.qty

                }

            }).filter(x => x.song);

        },

        totalPrice() {

        return this.cartItems.reduce((total, item) => {

        return total + (item.qty \* item.song.price);

        }, 0)},

        clearCart() {

            localStorage.removeItem('cart');

            this.cartItems = [];

        },

        updateLocalStorage(cart) {

            localStorage.setItem('cart', JSON.stringify(cart));

            this.loadCart();

        },

        removeItem(id){

            var cart = JSON.parse(localStorage.getItem("cart"));

            const item = cart.find(x => x.id == id);

            console.log(item)

            if(item){

                if(item.qty > 1){

                    console.log(item.qty);

                    item.qty--;

                }else{

                    cart = cart.filter(x => x.id != id);

                }

                this.updateLocalStorage(cart)

            }

        },

        async buy(){

            try{

                var cart = JSON.parse(localStorage.getItem("cart"));

                console.log(cart);

                var total\_price = this.totalPrice();

                console.log(total\_price);

                const dataforsend = {

                    items: cart.map(item => ({

                        id: item.id,

                        qty: item.qty

                    })),

                    total\_price: total\_price,

                    user:this['auth/getUserId']

                };

                console.log(dataforsend);

                await this.$store.dispatch("songs/buySongs",dataforsend);

                this.msg = "Thank you for your purchase! Your order has been successfully processed.";

                localStorage.removeItem("cart");

                setTimeout(() => {

                    this.$router.push({ name: 'home' });

                }, 2000);

            }catch(error){

                console.log(error);

            }

        }

    }

}

</script>

## InsertSong.vue

<template>

    <div class="container">

        <h1 class="text-center">Add song</h1>

        <div class="row">

            <div class="col-lg-4">

                <text-input @keyup="validate('name')" v-model="insertData.name.value" id="name" label="Name"/>

                <p class="text-danger" v-if="insertData.name.errormsg">{{ insertData.name.errormsg }}</p>

            </div>

            <div class="col-lg-4">

                <text-input @keyup="validate('price')" v-model="insertData.price.value" id="price" label="Price"/>

                <p class="text-danger" v-if="insertData.price.errormsg">{{ insertData.price.errormsg }}</p>

            </div>

            <div class="col-lg-4">

                <text-input @keyup="validate('image')" v-model="insertData.image.value" id="image" label="Image" placeholder="Image URL"/>

                <p class="text-danger" v-if="insertData.image.errormsg">{{ insertData.image.errormsg }}</p>

            </div>

        </div>

        <h2 class="text-center p-2">Genres</h2>

        <div class="row">

            <div v-for="g in genres" :key="g.id" class="col-lg-4">

                <div class="form-check form-check-inline">

                <input v-model="insertData.selectedGenres.value" class="form-check-input" type="checkbox" :id="g.id" :value="g.id">

                <label class="form-check-label" :for="g.id">{{ g.genre\_name }}</label>

                </div>

            </div>

            <p class="text-danger" v-if="insertData.selectedGenres.errormsg">{{ insertData.selectedGenres.errormsg }}</p>

        </div>

        <div class="row m-4">

            <button @click.prevent="insert" type="submit" class="btn btn-primary btn-lg btn-block">Add new song</button>

        </div>

        <p v-if="infomsg" :class=" msg ? 'text-danger': 'text-success' ">{{ infomsg }}</p>

    </div>

</template>

<script>

import TextInput from '@/components/inputfields/TextInput.vue';

import { mapGetters } from 'vuex';

export default {

    components:{TextInput},

    data(){

        return{

            insertData:{

                name:{

                    value:"",

                    errormsg:""

                },

                price:{

                    value:"",

                    errormsg:""

                },

                image:{

                    value:"",

                    errormsg:""

                },

                selectedGenres:{

                    value:[],

                    errormsg:""

                }

            },

            genres:[],

            error:false,

            msg:false,

            infomsg:""

        }

    },

    async created(){

        await this.$store.dispatch("songs/allGenres");

        this.genres = this.$store.state.songs.genres;

    },

    computed:{

        ...mapGetters(['auth/getUserId'])

    },

    methods:{

         async insert(){

            if(!this.error){

                const dataforsend = {

                name:this.insertData.name.value,

                artist:this['auth/getUserId'],

                genres:this.insertData.selectedGenres.value,

                price:this.insertData.price.value,

                image:this.insertData.image.value

                }

                console.log(dataforsend);

                try{

                    await this.$store.dispatch("songs/insertSong",dataforsend);

                    this.msg = false;

                    this.infomsg = "Song added successfully!";

                }catch(error){

                    console.log(error)

                    if(error.status == 403){

                        this.insertData.name.errormsg = error.response.data.errors['name'];

                        this.insertData.price.errormsg = error.response.data.errors['price'];

                        this.insertData.image.errormsg = error.response.data.errors['image'];

                        this.insertData.selectedGenres.errormsg = error.response.data.errors['genres'];

                    }else{

                        this.msg = true;

                        this.infomsg = "Please try later."

                    }

                }

            }

        },

        validate(type){

            switch(type){

                case "name":

                    if(this.insertData.name.value.length < 3){

                        this.error = true;

                        this.insertData.name.errormsg = "The name must be longer than 3 characters.";

                    }else{

                        this.error = false;

                        this.insertData.name.errormsg = "";

                    }

                    break;

                case "price":

                    if(this.insertData.price.value < 0){

                        this.error = true;

                        this.insertData.price.errormsg = "The price must be greater than 0.";

                    }else{

                        this.error = false;

                        this.insertData.price.errormsg = "";

                    }

                    break;

                case "image":

                    //eslint-disable-next-line

                    if(!/^(http(s):\/\/.)[-a-zA-Z0-9@:%.\_\+~#=]{2,256}\.[a-z]{2,6}\b([-a-zA-Z0-9@:%\_\+.~#?&//=]\*)$/.test(this.insertData.image.value)){

                        this.error = true;

                        this.insertData.image.errormsg = "The image must be URL.";

                    }else{

                        this.error = false;

                        this.insertData.image.errormsg = "";

                    }

            }

        }

    }

}

</script>

## EditSong.vue

<template>

    <div class="container">

        <div class="row">

            <h1 class="text-center">Edit</h1>

        </div>

        <div class="row">

            <div class="col-lg-4">

                <text-input @keyup="validate('name')" v-model="editData.name.value" id="name" label="Song name" :error="editData.name.errormsg"/>

            </div>

            <div class="col-lg-4">

                <text-input @keyup="validate('price')" v-model="editData.price.value" id="price" label="Price" :error="editData.price.errormsg"/>

            </div>

            <div class="col-lg-4">

                <text-input @keyup="validate('image')" v-model="editData.image.value" id="img" label="Image" placeholder="Image URL" :error="editData.image.errormsg"/>

            </div>

        </div>

        <h2 class="text-center p-2">Genres</h2>

        <div class="row">

            <div v-for="g in genres" :key="g.id" class="col-lg-4">

                <div class="form-check form-check-inline">

                <input

                 v-model="editData.selectedGenres.value"

                 class="form-check-input"

                 type="checkbox"

                 :id="g.id"

                 :value="g.id"/>

                <label class="form-check-label" :for="g.id">{{ g.genre\_name }}</label>

                </div>

            </div>

            <p class="text-danger" v-if="editData.selectedGenres.errormsg">{{ editData.selectedGenres.errormsg }}</p>

        </div>

        <div class="row m-4">

            <button @click.prevent="edit" type="submit" class="btn btn-primary btn-lg btn-block">Edit song</button>

            <p v-if="infomsg" :class=" msg ? 'text-danger': 'text-success' ">{{ infomsg }}</p>

        </div>

    </div>

</template>

<script>

import TextInput from '@/components/inputfields/TextInput.vue';

export default {

    name:"EditSong",

    components:{TextInput},

    data(){

        return{

            song:[],

            editData:{

                name:{

                    value:"",

                    errormsg:""

                },

                price:{

                    value:"",

                    errormsg:""

                },

                image:{

                    value:"",

                    errormsg:""

                },

                selectedGenres:{

                    value:[],

                    errormsg:""

                }

            },

            genres:[],

            error:false,

            msg:false,

            infomsg:""

        }

    },

    created(){

        this.one();

        this.genres = this.$store.state.songs.genres;

    },

    methods:{

        one(){

            this.song = this.$store.getters["songs/getOneSong"](this.$route.params.id)

            this.editData.name.value = this.song.name;

            this.editData.price.value = this.song.price;

            this.editData.image.value = this.song.image;

            this.editData.selectedGenres.value = this.song.genre\_ids.split(',').map(Number);

        },

        async edit(){

            if(!this.error){

                const dataforsend = {

                id:this.$route.params.id,

                name:this.editData.name.value,

                price:this.editData.price.value,

                image:this.editData.image.value,

                genres:this.editData.selectedGenres.value

            }

            // console.log(dataforsend);

                try{

                    await this.$store.dispatch("songs/edit",dataforsend);

                    this.msg = false;

                    this.infomsg = "Song edited successfully!";

                    console.log("bravo");

                }catch(error){

                    console.log(error);

                    if(error.status == 403){

                            this.editData.name.errormsg = error.response.data.errors['name'];

                            this.editData.price.errormsg = error.response.data.errors['price'];

                            this.editData.image.errormsg = error.response.data.errors['image'];

                            this.editData.selectedGenres.errormsg = error.response.data.errors['genres'];

                        }else{

                            this.msg = true;

                            this.infomsg = "Please try later."

                        }

                }

            }

        },

        validate(type){

            switch(type){

                case "name":

                    if(this.editData.name.value.length < 3){

                        this.error = true;

                        this.editData.name.errormsg = "The name must be longer than 3 characters.";

                    }else{

                        this.error = false;

                        this.editData.name.errormsg = "";

                    }

                    break;

                case "price":

                    if(this.editData.price.value < 0){

                        this.error = true;

                        this.editData.price.errormsg = "The price must be greater than 0.";

                    }else{

                        this.error = false;

                        this.editData.price.errormsg = "";

                    }

                    break;

                case "image":

                    //eslint-disable-next-line

                    if(!/^(http(s):\/\/.)[-a-zA-Z0-9@:%.\_\+~#=]{2,256}\.[a-z]{2,6}\b([-a-zA-Z0-9@:%\_\+.~#?&//=]\*)$/.test(this.insertData.image.value)){

                        this.error = true;

                        this.editData.image.errormsg = "The image must be URL.";

                    }else{

                        this.error = false;

                        this.editData.image.errormsg = "";

                    }

            }

        }

    }

}

</script>

# Store-Auth

## Index.js

import actions from "./actions"

import getters from "./getters"

import mutations from "./mutations"

export default{

    namespaced:true,

    state(){

        return{

            token:null,

            userId:null,

            role:null,

            userInfo:null,

        }

    },

    mutations,

    getters,

    actions

}

## Mutations.js

export default{

    setToken(state,token){

        localStorage.setItem("token", token)

        state.token = token;

    },

    clearToken(state){

        localStorage.removeItem("token")

        state.token = null;

    },

    clearRole(state){

        localStorage.removeItem("role");

        state.role = null;

    },

    clearUserId(state){

        localStorage.removeItem("userId")

        state.userId = null;

    },

    clearUserInfo(state){

        localStorage.removeItem("userInfo")

        state.userInfo = null;

    },

    setSub(state,id){

        localStorage.setItem("subs", id)

        state.sub = id;

    },

    setUserId(state,userId){

        if(userId == null){

            localStorage.removeItem("userId");

        }

        else{

            localStorage.setItem("userId",userId);

        }

        state.userId = userId;

    },

    setRole(state,role){

        if(role == null){

            localStorage.removeItem("role");

        }else{

            localStorage.setItem("role",role);

        }

        state.role = role;

    },

    setUserInfo(state,userinfo){

        if(userinfo == null){

            localStorage.removeItem("userInfo");

        }

        else{

            localStorage.setItem("userInfo",userinfo);

        }

        state.userInfo = userinfo;

    },

    clearCart(state){

        localStorage.removeItem("cart");

        state.cart = [];

    }

}

## Getters.js

export default{

    getToken(state){

        if(localStorage.getItem("token")){

            state.token = localStorage.getItem("token");

        }

        return state.token;

    },

    getAa(state){

        if(localStorage.getItem("token")){

            state.token = localStorage.getItem("token");

        }

        return !!state.token;

    },

    getTokenInfo(state){

        let parseJwt= (token) => {

            try {

              return JSON.parse(atob(token.split('.')[1]));

            } catch (e) {

              return null;

            }

        }

        const tokenInfo = parseJwt(state.token);

        return tokenInfo;

    },

    getUserId(state){

        if(localStorage.getItem("userId")){

            state.userId = localStorage.getItem("userId");

        }

        return state.userId;

    },

    getRole(state){

        if(localStorage.getItem("role")){

            state.role = localStorage.getItem("role");

        }

        return state.role;

    }

}

## Actions.js

import axios from "@/axios"

export default{

    async login(context,payload){

        const response = await axios.post("/login",payload);

        context.commit("setToken", response.data.authorisation.token);

        context.commit("setUserId",response.data.user.id);

        context.commit("setUserInfo", response.data.user);

        context.commit("setRole",response.data.user.role.role\_name);

    },

    logout(context){

        context.commit("clearToken");

        context.commit("clearUserId");

        context.commit("clearRole");

        context.commit("clearUserInfo");

        context.commit("clearCart");

    },

    sub(context,payload){

        if(!localStorage.getItem("subs"))

        context.commit("setSub",payload);

    },

    async register(\_,payload){

         await axios.post("/register",payload);

    },

    async User(context,id){

        const response = await axios.get("/user/" + id);

        context.commit("setUserInfo",response.data);

    }

}

# Auth-songs

## Index.js

import getters from "./getters"

import actions from "./actions"

import mutations from "./mutations"

export default{

    namespaced:true,

    state:{

        genres:[],

        songs:[],

        song:[]

    },

    getters,

    mutations,

    actions

}

## Mutations.js

export default{

    setGenres(state,genres){

        state.genres = genres;

    },

    setSongs(state,songs){

        state.songs = songs;

    }

}

## Getters.js

export default{

   getGenres(state){

    return state.genres;

   },

   getSongs(state){

      return state.songs;

   },

   getOneSong: (state) => (payload)=>{

      if(state.songs.find(x=>x.id == payload)){

          return state.songs.find(x=>x.id == payload);

      }

      else{

          return null;

      }

    },

}

## Actions.js

import axios from "../../axios";

export default{

   async allGenres(context){

    const response = await axios.get("/genres");

    context.commit("setGenres",response.data);

   },

   async insertSong(\_,payload){

      await axios.post("/songs",payload);

   },

   async allSongs(context){

      const response = await axios.get("/songs");

      context.commit("setSongs", response.data.songs);

   },

   async edit(\_,payload){

      await axios.post("/edit",payload);

   },

   async deleteSong(\_,payload){

      await axios.post("/delete/" + payload);

   },

   async buySongs(\_,payload){

      await axios.post("/buy",payload);

   },

   async likeSong(\_,payload){

      await axios.post("/like",payload);

   }

}