

## Festivian

Stefan Kuperšak 68/16

<https://festivian.bluegrid.io/>

<https://github.com/dev-cyprium/php-festivals>

<b>Uvod</b>	<b>5</b>
Korišćene tehnologije	5
Opis funkcionalnosti	6
Templates	7
<b>Organizacija</b>	<b>8</b>
Organizaciona šema	8
Korišćena baza podataka	9
Detalji o stranicama	10
Početna	10
Festivali	11
Autor	12
Admin	13
Novi festival	13
Edituj festival	13
Statistika	14
Registracija	14
Prijava	15
<b>Kodovi</b>	<b>16</b>
Abstrakcije	16
Korišćeni kod	17
Festival.php	17
Glasaj.php	18
Message.php	19
Message.php	20
_accounts.scss	21
_autor.scss	23
_autor.scss	24
_contact.scss	26
_festivals.scss	29
_footer.scss	32

_header.scss	33
_hero.scss	35
_login.scss	37
_thoughts.scss	40
_utils.scss	41
_variables.scss	45
_main.scss	45
Admin_edit.js	46
Alerts.js	49
App.js	49
Contact.js	51
Dates.js	54
Map.js	54
Search.js	58
Tabs.js	61
Validaors.js	62
Vote.js	67
Brunch-conifg.js	68
Package.json	70
Constants.php	71
Database.php	72
Router.php	72
Func.php	74
Festivals.php	81
Links.php	82
Users.php	83
.htaccess	85
Index.php	86
Admin_nav.php	87
Admin.php	88
Admin_anketa.php	92
Admin_edit.php	93
Admin_korisnici.php	98

Autor.php	101
Festivali.php	103
Footer.php	105
Header.php	106
Includes.php	107
Logout.php	108
Pocetna.php	108
Prijava.php	112
Registracija.php	114
.gitignore	117
Env.json	118

## Uvod

### Korišćene tehnologije

Za izradu platforme korišćene su sledeće tehnologije:

- HTML 5
- Node JS
- Babel Transpiler
- Git
- Apache
- AWS
- Mozilla Quantum
- PHP 7.2
- MySQL
- PHPmyAdmin
- SASS
- Brunch ( asset manager )
- PHPStorm
- Ubuntu
- jQuery
- Select2

Hosting realizujem lično, na jednom od mojih servera na kome se nalazi produkcijska verzija kodova. Napravljen je **build & deploy** sistem koji automatski sinhronizuje stanje projekta sa git repositorijumom da omogći brz i lak deployment bez downtime.

## Opis funkcionalnosti

1. **Animirana mapa:** na početnoj stranici koristi ručno napisanu SVG animaciju gde se zvezdice nasumično smenjuju
2. **Sistem za logovanje:** omogućava adminima i korisnicima da se prijave i koriste platformu
3. **Sistem za registraciju:** omogućava korisnicima da se registruju i koriste platformu
4. **Sistem za pretragu festivala:** omogućava jednostavan način za pronalazak festivala.
5. **Paginacija:** omogućava pregledan način za listanje festivala
6. **Admin panel za nove festivale i menjanje starih:** Obriši, izmeni, dodaj festivale preko admin panela
7. **Anketa za izbor:** omiljenog festivala, statistika na admin panelu
8. **Kontakt forma:** koja čuva i prikazuje poruke na admin panelu kao i šalje mail administratoru sajta

## Templates

Header

POČETNA

FESTIVALI

AUTOR

DOKUMENTACIJA

PRIJAVA

REGISTRACIJA

SADRZAJ

Mail

Lozinka

Prijavi se

Nemas nalog? Registruj se

FOOTER

O Festivian

Festivian platforma je moderna, skalabilna platforma koja omogućava brz, lak, i ekonomičan, jednostavan pristup festivalima i korisnicima i organizatorima

Konektujte se


f

in

tw

ig

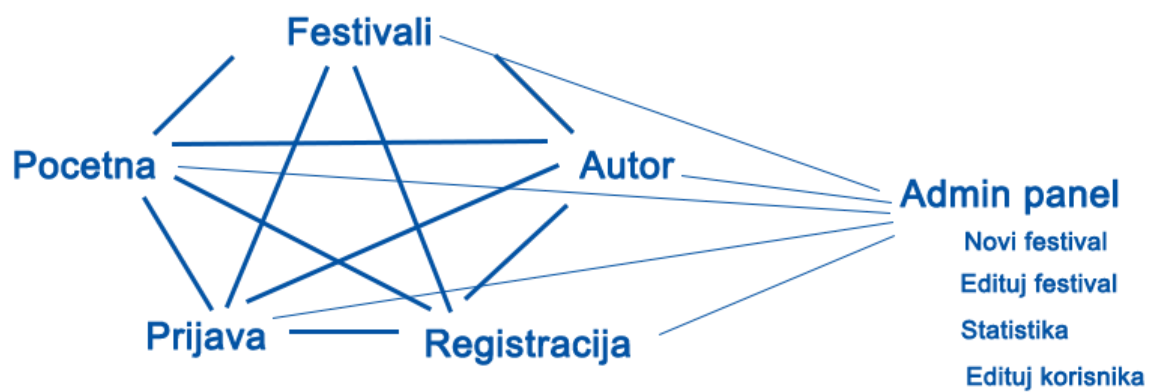
Domen



Glavni template koji se koristi na svakoj stranici na festivian platormi. Sav kod je napisan u SASS-u i ni jedan eksterni framework nije korišćen za kreiranje ove teme.

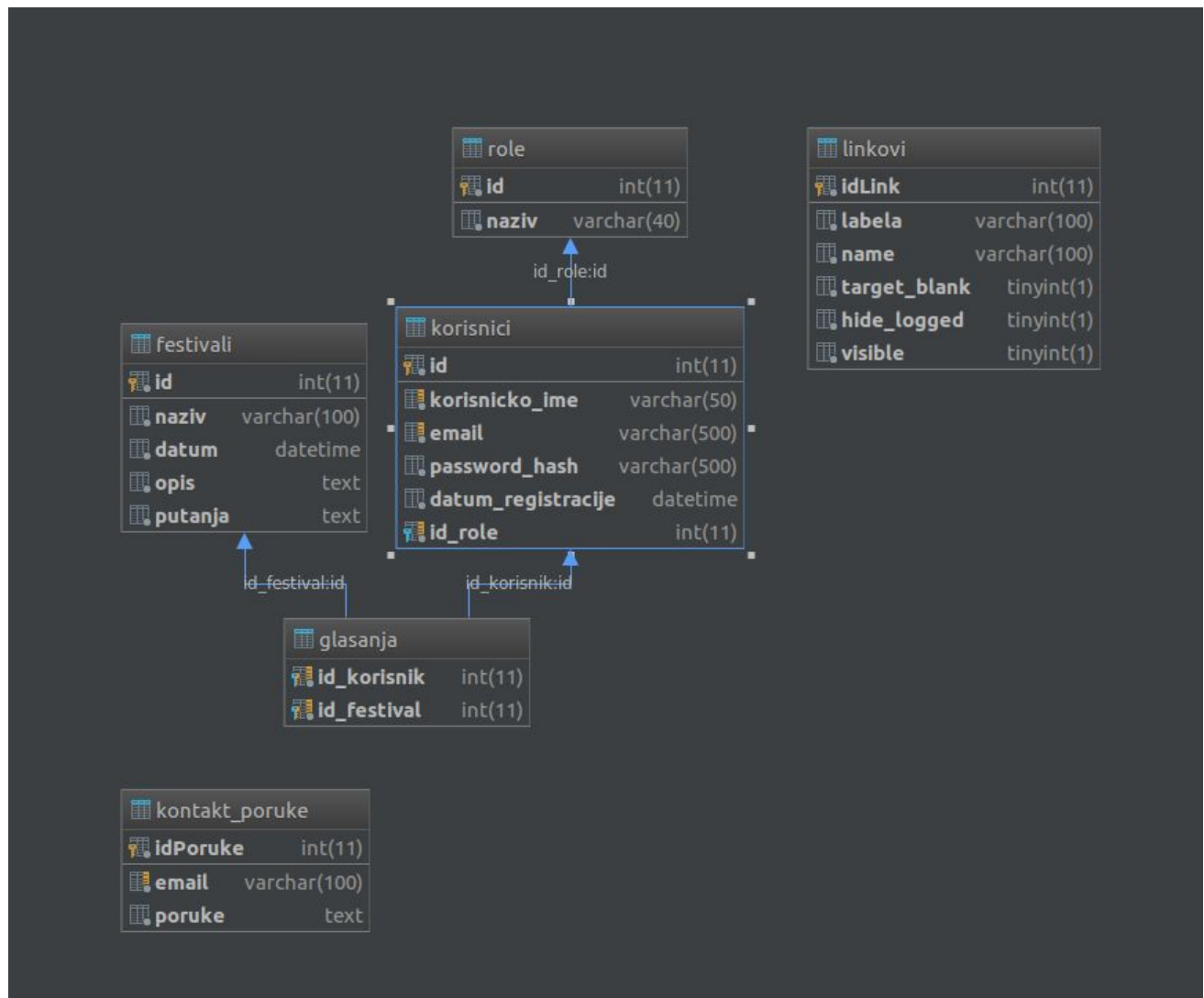
## Organizacija

Organizaciona šema



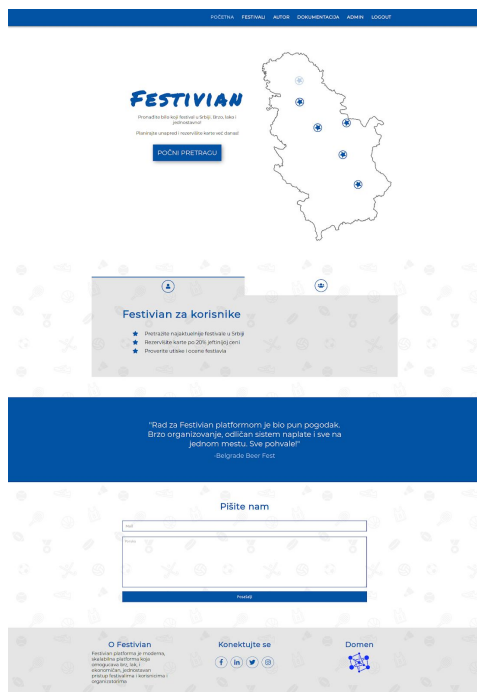


## Korišćena baza podataka



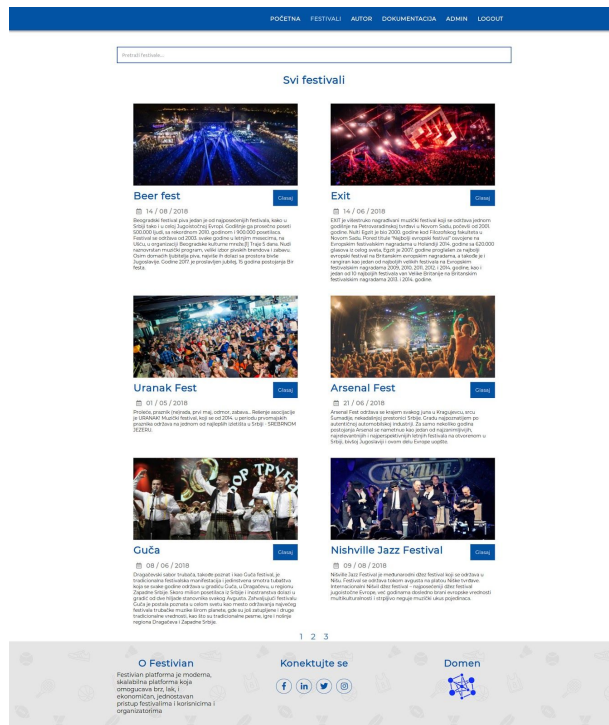
## Detalji o stranicama

### Početna



Početna stranica predstavlja reklamnu stranicu sajta. Ona sadrži moju **animiranu SVG** mapu Srbije, dugme za **početak pretrage**, **formu za kontakt** kao i **tabove** koji opisuju kako se festivian koristi za korisnike i organizatore.

## Festivali



Stranica festivali sadrži **galeriju slika** festivala, kao i dugme za **anketu** ukoliko su korisnici ulogovani.

Takođe, ova stranica sadrži takozvanu **debounce** funkcionalnost koja omogućava pretragu festivala čim korisnik ne dira tastaturu 300ms.

Na dnu stranice se nalazi **paginacija** koja ovu galeriju prikazuje na određen broj stranica u zavisnosti od broja festivala. Uvek se prikazuje 6 festivala po stranici, mada, ovo je lako moguće promeniti.

[POČETNA](#)
[FESTIVALI](#)
[AUTOR](#)
[DOKUMENTACIJA](#)
[ADMIN](#)
[LOGOUT](#)

## Stefan Kuprešak



Big Data // Full Stack Web // Open Source Software

Zdravo! S obzirom da smo svi mi "ambiciozni programeri", na ovaj ili onaj način ovog puta ću preskočiti formalnosti i preći na stvar. Bavim se Big Data developmentom u programskom jeziku Elktir kao i interesovanju za moderne web frameworke za full stack development poput React-a, Laravel-a i eliksirovog Feniksa.

Poseđujem gomilu iskustva na velikom spektru polja u IT sektoru, uključujući rad na mnogim projektima i to na izgradnji i skaliranju istih od nule. Trenutno sam zaposlen u firmi BlueGrid, gde učestvujem na mnogim internim i spoljnim projektima. Takođe, imam veliko interesovanje za eksperimentalne jezike poput GoLang-a i njemu sličnim pa povremeno poklanjam deo svog vremena istraživanju takvih tehnologija.

**Festivian** je platforma koju sam inicijalno zamislio da sadrži mnogo više zanimljivih stvari, međutim na žalost zbog vremenskih ograničenja nisam uspeo da sprovedem krajnju zamisao već samo sastavih ovaj kratak prototip nekih funkcionalnosti sa kranje platforme.

### O Festivian

Festivian platforma je moderna, skalabilna platforma koja omogućava brz, lak i ekonomičan pristup festivalima i korisnicima i organizatorima

### Konektujte se

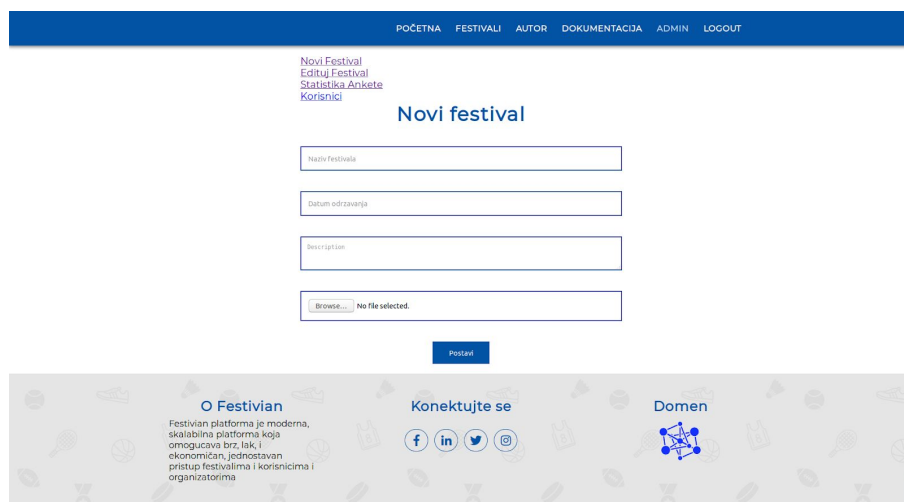


### Domen



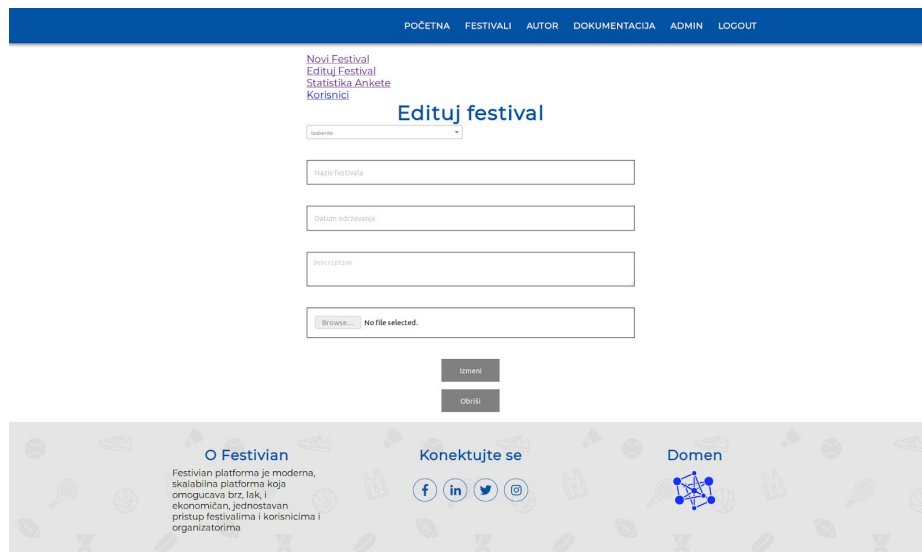
Stranica o autoru je stranica gde sam napisao par pasusa o meni.

## Admin Novi festival



Stranica na **admin panelu** koja predstavlja interfejs za unos novih festivala omogućava unos novih festivala u sistem

## Edituj festival



Stranica na **admin panelu** koja omogućava editovanje postojećih festivala.

POČETNA FESTIVALI AUTOR DOKUMENTACIJA ADMIN LOGOUT


[Novi Festival](#)  
[Edituj Festival](#)  
[Statistika Ankete](#)  
[Korisnici](#)

### Statistika ankete

Ime festivala	Broj glasova
Beer fest	3
Uranak Fest	1

**O Festivian**  
 Festivian platforma je moderna, skalabilna platforma koja omogućava brz, lak i ekonomičan, jednostavan pristup festivalima i korisnicima i organizatorima

**Konektujte se**  
[f](#) [in](#) [t](#) [@](#)

**Domen**  


Stranica na **admin panelu** na kojoj se vide **rezultati ankete** na sajtu.

## Registracija

POČETNA FESTIVALI AUTOR DOKUMENTACIJA PRIJAVA REGISTRACIJA

### Registracija


Email  
 Ime i prezime  
 Lozinka  
 Lozinka ponovo

[Registruj se](#)

Imaš nalog? [Prijavi se](#)

**O Festivian**  
 Festivian platforma je moderna, skalabilna platforma koja omogućava brz, lak i ekonomičan, jednostavan pristup festivalima i korisnicima i organizatorima

**Konektujte se**  
[f](#) [in](#) [t](#) [@](#)

**Domen**  


Stranica koja omogućava **registraciju** korisnika u sistem.

## Prijava

[POČETNA](#) [FESTIVALI](#) [AUTOR](#) [DOKUMENTACIJA](#) [PRIJAVA](#) [REGISTRACIJA](#)

### Prijava

[Nemas nalog? Registruj se](#)


#### O Festivian

Festivian platforma je moderna, skalabilna platforma koja omogućava brz, lak, i ekonomičan, jednostavan pristup festivalima i korisnicima i organizatorima

#### Konektujte se

[f](#) [in](#) [t](#) [@](#)

#### Domen



Stranica koja omogućava **prijavu** korisnika na sistem.

## Kodovi

### Abstrakcije

Sajt **Festivian** koristi moju napisanu biblioteku za validaciju svih formulara na front-endu pomoću data-\* atributa koja omogućava brzu i laku implementaciju na bilo kom formularu.

Takođe, **Festivian** koristi funkcionalni backend model za validaciju, insertovanje i update koji omogućava laku i skalabilnu integraciju novih modela u sistem.



Korišćeni kod

### Festival.php

```
<?php
$status = 403;

if(adminLogged()) {
    $status = 400;
    if(isset($_POST['id'])) {
        $id = $_POST['id'];

        $status = 200;
        $query = "select * from festivali where
                    id=:id";
        $result = safeQuery($conn, $query, ["id" => $id], true);
        echo json_encode($result);
    }
}

http_response_code($status);
```

## Glasaj.php

```
<?php

$status = 403;

if(userLogged()) {
    if($_SERVER['REQUEST_METHOD'] == 'POST') {
        $status = 400;
        if (isset($_POST['id'])) {
            $idFest = $_POST['id'];
            $idKor = $_SESSION['user']->id;
            $upit = "INSERT INTO glasanja
                    VALUES(:korisnik, :festival)";
            try {
                $status = 200;
                runSafeQuery($conn, $upit, [
                    "korisnik" => $idKor,
                    "festival" => $idFest
                ]);
                echo json_encode(["message" => "Uspesno ste
glasali"]);
            } catch (PDOException $e) {
                $status = 403;
                echo json_encode(["message" => "Vec ste glasali"]);
            }
        }
    }
    else if($_SERVER['REQUEST_METHOD'] == 'DELETE') {
        $status = 204;
    }
}
```

```
$idKor = $_SESSION['user']->id;
$upit = "DELETE FROM glasanja where id_korisnik=:id";
runSafeQuery($conn, $upit, ["id" => $idKor]);
}
}

http_response_code($status);
```

### Message.php

```
<?php
$message = $_POST['message'];
$email    = $_POST['email'];

$msgReg = '/^{10,100}$/';
$status = 200;
$errors = [];

if(!preg_match($msgReg, $message)) {
    $errors['message'] = "Mora biti između 10 i 100
karaktera";
    $status = 400;
}

if(!filter_var($email, FILTER_VALIDATE_EMAIL)) {
    $errors['email'] = "Mora biti u dobrom formatu";
    $status = 400;
}
```

```
http_response_code($status);

if($status == 400) {
    echo json_encode($errors);
} else {
    $headers = "From: administrators@festivian.bluegrid.io
\r\n";
    mail("stefan_vg@hotmail.com", "Kontakt sa platforme
festivian", $message . " Email: " . $email, $headers);
}
```

### Message.php

```
<?php
$term = $_GET['term'];

$query = "select * from festivali
        where lower(naziv) like lower(:naziv)";

$result = safeQuery($conn, $query, ["naziv" => "%$term%"]);

echo json_encode($result);
```

\_accounts.scss

```
#acc-tabs {  
  display: flex;  
  @include center-width();  
  .tab {  
    flex: 1;  
    text-align: center;  
    padding: 1em 0;  
    cursor: pointer;  
    &.active {  
      border-top: 3px solid $primary;  
      background: rgba(0, 0, 0, 0.1);  
    }  
    i {  
      font-size: 2rem;  
      line-height: 46px;  
      width: 5rem;  
      height: 5rem;  
      border: 2px solid $primary;  
      border-radius: 50%;  
      color: $primary;  
    }  
  }  
}  
  
.tab-content {  
  background: rgba(0, 0, 0, 0.1);  
  @include center-width();
```

```
padding: 2em 0;
padding-bottom: 8em;
h1 {
  color: $primary;
  font-size: 4rem;
  padding-top: .75em;
  padding-left: 3em;
}
ul {
  margin-top: 2em;
  font-size: 2rem;
  list-style-type: none;
  margin-left: 6em;
  li {
    padding-left: 2em;
    margin-bottom: .5em;
    i {
      color: $primary;
      margin-right: 1em;
    }
  }
}
}
```

\_autor.scss

```
.autor {  
  @include center-width();  
  h1 {  
    color: $primary;  
    font-size: 4rem;  
    margin: .76em 0;  
  }  
  &__image {  
    text-align: center;  
    margin-bottom: 4em;  
    img {  
      width: 150px;  
      height: 150px;  
      border-radius: 50%;  
    }  
    h2 {  
      margin-top: 1.5em;  
      color: lighten($primary, 10%);  
    }  
  }  
  &__bio {  
    width: 50%;  
    min-width: 10rem;  
    text-align: justify;  
    font-size: 1.8rem;  
    margin: 0 auto;  
    p {
```

```

    margin-bottom: .7em;
  }
}
}

```

### \_autor.scss

```

.contact-form {
  @include center-width();
  padding-bottom: 10rem;
  padding-top: 7.5rem;
  h1 {
    text-align: center;
    color: $primary;
    font-size: 3.4rem;
    margin-bottom: 1em;
  }
  input[type='text'] {
    display: block;
    padding: .75em;
    width: 100%;
    border: 2px solid $primary;
    transition: border ease-out 800ms;
    background: transparent;
    &:focus,
    &:active {
      border: 2px solid lighten($primary, 20%);
    }
  }
  .form-wrap {
    display: block;
    margin: 0 auto;
    max-width: 80%;
    textarea {
      border: 2px solid $primary;
    }
  }
}

```



```
background: transparent;
width: 100%;
min-height: 20rem;
max-width: 100%;
min-width: 50%;
max-height: 40rem;
margin-top: 2rem;
padding: .75em;
transition: border ease-out 800ms;
&:focus,
&:active {
  border: 2px solid lighten($primary, 20%);
}
}
.counter {
  font-size: 2rem;
  margin-top: .5em;
  content: "50";
  display: block;
  color: $primary;
  &.danger {
    color: crimson;
  }
}
}
button {
  display: block;
  padding: .75em;
  width: 80%;
  margin: 0 auto;
  border: 2px solid $primary;
  margin-top: 1em;
  transition: border ease-out 800ms;
  background: $primary;
  color: white;
```

```
    transition: all ease 800ms;
    cursor: pointer;
    &:hover {
      background: lighten($primary, 20%);
      border: 2px solid lighten($primary, 20%);
    }
  }
}
```

### \_contact.scss

```
.contact-form {
  @include center-width();
  padding-bottom: 10rem;
  padding-top: 7.5rem;
  h1 {
    text-align: center;
    color: $primary;
    font-size: 3.4rem;
    margin-bottom: 1em;
  }
  input[type='text'] {
    display: block;
    padding: .75em;
    width: 100%;
    border: 2px solid $primary;
    transition: border ease-out 800ms;
    background: transparent;
    &:focus,
    &:active {
```

```
        border: 2px solid lighten($primary, 20%);
    }
}
.form-wrap {
    display: block;
    margin: 0 auto;
    max-width: 80%;
    textarea {
        border: 2px solid $primary;
        background: transparent;
        width: 100%;
        min-height: 20rem;
        max-width: 100%;
        min-width: 50%;
        max-height: 40rem;
        margin-top: 2rem;
        padding: .75em;
        transition: border ease-out 800ms;
        &:focus,
        &:active {
            border: 2px solid lighten($primary, 20%);
        }
    }
}
.counter {
    font-size: 2rem;
    margin-top: .5em;
    content: "50";
    display: block;
```

```
        color: $primary;
        &.danger {
            color: crimson;
        }
    }
}
button {
    display: block;
    padding: .75em;
    width: 80%;
    margin: 0 auto;
    border: 2px solid $primary;
    margin-top: 1em;
    transition: border ease-out 800ms;
    background: $primary;
    color: white;
    transition: all ease 800ms;
    cursor: pointer;
    &:hover {
        background: lighten($primary, 20%);
        border: 2px solid lighten($primary, 20%);
    }
}
}
```

\_festivals.scss

```
.festivali {  
  @include center-width();  
  margin-top: 5rem;  
  &__list {  
    display: flex;  
    flex-wrap: wrap;  
    justify-content: space-around;  
  }  
  &__wrap {  
    h1 {  
      margin: 1em 0;  
      font-size: 3.2rem;  
      color: $primary;  
      text-align: center;  
    }  
  }  
}  
  
.festival {  
  margin: 2em;  
  width: 50rem;  
  &__slika {  
    display: block;  
  
    img {  
      height: 250px;  
      width: 100%;  
    }  
  }  
}
```

```
    }  
  }  
  &__title {  
    display: flex;  
    margin-top: 1em;  
    h2 {  
      padding-right: .5em;  
    }  
    a {  
      display: block;  
      padding: 1em;  
      background: $primary;  
      color: white;  
      text-decoration: none;  
      font-size: 1.5em;  
      margin-left: auto;  
      align-self: flex-start;  
    }  
  }  
  h2 {  
    font-size: 3rem;  
    color: $primary;  
  }  
  h3 {  
    font-size: 1.8rem;  
    color: gray;  
    margin-top: .3em;  
  }  
}
```

```
i {
  margin: 0 .5em;
}
}
p {
  margin-top: .5em;
  font-size: 1.4rem;
}
}

.pagination {
  display: flex;
  justify-content: center;
  font-size: 2.5rem;
  a {
    text-decoration: none;
    color: $primary;
    transition: color linear 500ms;
    margin: 0 .5em;
    &:hover,
    &:active {
      color: lighten($primary, 20%);
    }
  }
}
```

\_footer.scss

```
footer {
  background-image: url("/assets/images/sports.png");
  .footer-wrap {
    background: rgba(0, 0, 0, .1);
    padding: 5rem 3rem;
    .footer-center {
      @include center-width();
      display: flex;
      color: $primary;
      h1 {
        font-size: 2.8rem;
        margin-bottom: .75rem;
        text-align: center;
      }
      p {
        color: black;
        font-size: 1.8rem;
        text-align: left;
      }
    }
  }
}

.icons-connected {
  margin-top: 3rem;
  text-align: center;
  i {
```



```

    font-size: 2.4rem;
    margin: 0 .5rem;
    width: 2em;
    text-align: center;
    display: inline-block;
    border-radius: 60px;
    border: 1.5px solid $primary;
    padding: .4em 0;
  }
}

.logo-foot {
  text-align: center;
  img {
    width: 10rem;
  }
}

```

### \_header.scss

```

header {
  background: $primary;
  box-shadow: 0px 2px 6px 1px gray;
  nav {
    @include center-width();
  }
  ul {
    list-style-type: none;
    display: flex;

```

```
justify-content: flex-end;
margin-right: 1rem;
padding: 2em 0;
li {
  padding: .5em;
  margin: 0 1em;
  a {
    text-decoration: none;
    font-size: 1.7rem;
    text-transform: uppercase;
    color: white;
    font-weight: 700;
    &:hover, &:focus, &.active {
      color: #b9d6f2;
    }
  }
}
}
```

\_hero.scss

```
/*
  Map
*/
#center {
  display: flex;
  @include center-width();
  margin-top: 5rem;
  min-height: 90vh;
}

#map {
  display: flex;
  flex: 1 0;
  svg {
    width: 100%;
    height: auto;
  }
}

#hero-title {
  display: flex;
  flex: 1 2;
  margin-left: 10rem;
  align-items: center;
  justify-content: center;
  margin-bottom: 20rem;
  flex-direction: column;
```

```
a {
  background: $primary;
  color: #fafafa;
  text-decoration: none;
  padding: .7em;
  font-size: 2.5rem;
  margin-top: .7em;
  text-transform: uppercase;
  box-shadow: 4px 4px 16px -2px gray;
}
h1 {
  font-size: 8.2rem;
  color: $primary;
  letter-spacing: .7rem;
  font-family: 'Permanent Marker', sans-serif;
}
p {
  font-size: 1.7rem;
  max-width: 50rem;
  text-align: center;
  margin: .3em 0;
  line-height: 1.2em;
  margin-bottom: 1em;
}
}

.circle-star {
  opacity: 0;
```

```
    transition: opacity ease-in 500ms;
  }

  .circle-star.active {
    opacity: 1;
  }
```

### \_login.scss

```
.site-form {
  display: flex;
  flex-direction: column;
  justify-content: center;
  padding: 2em;
  min-height: 60em;
  @include center-width($width: 70rem);
  &__title {
    font-size: 4rem;
    color: $primary;
    text-align: center;
  }
  &__wrap {
    p {
      font-size: 2rem;
      margin-top: 2em;
      a {
        text-decoration: none;
        color: $primary;
        &:hover,
```

```
        &:active {
            color: lighten($primary, 20%);
        }
    }
}
}
}

.form {
    font-size: 1.4rem;
    .form__group {
        margin: 3em 0;
    }
    .form__errors {
        color: crimson;
    }
    .form__control {
        display: block;
        padding: 1em;
        width: 100%;
        border: 2px solid $primary;
        transition: border ease-out 800ms;
        background: transparent;
        &:focus,
        &:active {
            border: 2px solid lighten($primary, 20%);
        }
    }
    margin-bottom: .75em;
}
```

```
&.disabled {
  border: 2px solid gray;
  cursor: not-allowed;
}
}
.form__submit {
  display: block;
  width: 8em;
  margin: 0 auto;
  padding: 1em;
  border: none;
  cursor: pointer;
  transition: background ease 250ms;

  &.disabled {
    background: gray;
    cursor: not-allowed;
  }

  &--primary {
    background: $primary;
    color: white;
    &:hover:not(.disabled) {
      background: lighten($primary, 20%);
    }
  }
  &--delete {
    background: crimson;
  }
}
```

```
margin-top: 1em;
color: white;
&:hover:not(.disabled) {
  background: lighten(crimson, 20%);
}
}
}
}
```

### \_thoughts.scss

```
.thoughts-slider {
  background: $primary;
  padding: 9rem 0;
  blockquote {
    @include center-width($width: 80rem);
    font-size: 3rem;
    color: white;
    text-align: center;
    span {
      display: block;
      text-align: center;
      color: #b9d6f2;
      font-size: 2.4rem;
      margin-top: 1.4rem;
    }
  }
}
```



\_utils.scss

```
::selection {
  background: #b9d6f2;
}

::-moz-selection {
  background: #b9d6f2;
}

header ul a::-moz-selection {
  background: black !important;
}

header ul a::selection {
  background: black !important;
}

.bg-wrap {
  background-image: url("/assets/images/sports.png");
}

.p-8rem {
  padding: 8rem;
}

.alert {
  display: block;
  position: fixed;
```

```
width: 40%;
padding: 2em;
background: #f9fbb2;
margin: 0 auto;
top: 5vh;
left: 50%;
transform: translateX(-50%);
font-size: 1.5rem;
color: rgb(51, 51, 51);
box-shadow: 2px 2px 10px 1px gray;
.closebtn {
  position: absolute;
  right: 2rem;
  top: 50%;
  transform: translateY(-50%);
  font-size: 3rem;
  color: rgb(51, 51, 51);
  cursor: pointer;
  &:hover {
    color: crimson;
  }
}

.form-error {
  border: 2px solid crimson !important;
}

.err-tooltip {
```

```
        display: block;
        color: crimson;
        font-size: 1.4rem;
    }
    .col {
        flex: 0 0 25%;
    }

    .col-2 {
        flex: 1 0 50%;
    }

    .text-center {
        text-align: center;
    }

    .big-link {
        font-size: 2rem;
    }

    .vote-table {
        border-collapse: collapse;
        font-size: 2rem;
        margin-top: 1.5em;
        width: 100%;
        tr, th, td {
            border: 1px solid $primary;
            padding: 2em;
        }
    }
}
```

```
    }  
  }  
  
  $breakpoints: (  
    'small': 767px,  
    'medium': 992px,  
    'large': 1200px  
  ) !default;  
  
  @mixin respond-to($breakpoint) {  
    @if map-has-key($breakpoints, $breakpoint) {  
      @media (min-width: map-get($breakpoints, $breakpoint))  
    {  
      @content;  
    }  
  }  
  
  @else {  
    @warn "Unfortunately, no value could be retrieved from  
    '#{ $breakpoint }'."  
    + "Available breakpoints are:  
    #{map-keys($breakpoints)}.";  
  }  
}
```

### \_variables.scss

```
// Variables
$primary: #0353a4;
$secondary: #b9d6f2;

@mixin center-width($margin:0, $width:120rem) {
  max-width: $width;
  margin: $margin auto;
}
```

### \_main.scss

```
* {
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}

html {
  font-size: 62.5%;
  font-family: 'Montserrat', sans-serif;
}

@import 'variables';

@import 'utils';

@import 'partials/header';
@import 'partials/hero';
@import 'partials/accounts';
```

```
@import 'partials/thoughts';
@import 'partials/contact';
@import 'partials/footer';
@import 'partials/login';
@import 'partials/festivals';
@import 'partials/autor';
```

### Admin\_edit.js

```
import $ from 'jquery';
import select2 from 'select2';
import 'pickadate/lib/picker';
import 'pickadate/lib/picker.date';

select2($);

class AdminEdit {
  initializeAdminEdit() {
    $("#korisnik-select").select2();
    $("#fetival-select").select2();
    $("#fetival-select").change((ev) => {
      this.handleChange(ev)
    });
    $("#korisnik-select").change((ev) => {
      this.handleKorChange(ev);
    });
    this.form = $(".site-form--admin_edit form");
    this.korForm = $(".site-form--korisnik_edit form");
  }
}
```

```
handleChange(ev) {
  const id = $("#festival-select").val();
  $.ajax({
    url: '/api/festival',
    method: 'POST',
    data: {
      id: id
    },
    success: (data) => {
      const naziv = this.form.find("#naziv");
      const datum = this.form.find("#datum");
      const opis = this.form.find("#opis");
      const slika = this.form.find("#slika");
      const izmeni = this.form.find("#izmeni");
      const obrisi = this.form.find("#obrisi");
      this.form.find("#festID").val(id);
      this.updateInput(naziv, data.naziv);
      this.updatePicker(datum, data.datum);
      this.updateInput(opis, data.opis);
      this.updateInput(slika, null);
      this.updateInput(izmeni, null);
      this.updateInput(obrisi, null);
      $("#preview").attr('src', data.putanja);
    }
  });
}
```

```
updatePicker(input, rawDate) {  
  let picker = input.pickadate('picker');  
  const date = rawDate.split(" ")[0];  
  picker.set('select', date, {format: 'yyyy-mm-dd'});  
  input.removeAttr('disabled');  
  input.removeClass('disabled');  
}  
  
updateInput(input, val) {  
  input.val(val);  
  input.removeAttr('disabled');  
  input.removeClass('disabled');  
}  
}  
  
export default new AdminEdit();
```



## Alerts.js

```
import $ from 'jquery'

function template(message) {
  return `
    <div class='alert'>
      <span class='message'>${message}</span>
      <span class='closebtn'
onclick="this.parentElement.style.display='none';">
        &times;</span>
    </div>
  `;
}

let el = null;

const Alerts = {
  createAlert: (message) => {
    if(el) {
      el.css("display", "block");
      el.find('.message').html(message);
    } else {
      el = $(template(message));
      $('body').append(el)
    }
  },
  closeAll: () => {
    $(".alert").css('display', 'none');
  }
}
```

```
    }  
};  
  
export default Alerts;
```

## App.js

```
// Main JS file  
import Map from "./map.js";  
import Tabs from "./tabs";  
import alerts from "./alerts";  
import Contact from "./contact";  
import Validator from './validators';  
import SearchFestivals from './search';  
import transform from './dates';  
import initVoteSystem from './vote';  
import AdminEdit from './admin_edit';  
  
window.App = {};  
App.alerts = alerts;  
  
function boot() {  
    let map = new Map();  
    let tabs = new Tabs();  
    let contact = new Contact();  
  
    Validator.initializeFormValidators();  
    SearchFestivals.initializeSearchBar();  
    AdminEdit.initializeAdminEdit();  
    transform();  
    initVoteSystem();  
}  
  
window.addEventListener('DOMContentLoaded', boot);
```

```
import $ from 'jquery'

export default class Contact {
  constructor() {
    this.form = $('.contact-form form')
    this.form.submit((e) => this.handle(e))
    this.form.find('#message').on('keyup', (e) =>
this.handleKeyUp(e))
    this.firstKey = true
  }

  handleKeyUp(e) {
    if(this.firstKey) {
      this.firstKey = false

this.form.find(".form-wrap:last-of-type").append(`<span
class='counter'></span>`);
    }
    var count = 100 -
this.form.find('#message').val().length;
    if(count < 10) {
      this.form.find('.counter').addClass('danger');
    }

    if(count > 10) {
      this.form.find('.counter').removeClass('danger');
    }
  }
}
```

```
        this.form.find('.counter').text(count);
    }

    handle(e) {
        e.preventDefault()
        this._removeErr()
        $.ajax({
            url: '/api/message',
            method: 'POST',
            data: {
                email: this.form.find('#email').val(),
                message: this.form.find("#message").val()
            },
            success: (data, textStatus, xhr) => {
                this.receiveData(data, xhr.status)
            },
            error: (xhr) => {
                this.receiveData(JSON.parse(xhr.responseText),
xhr.status)
            }
        })
    }

    _removeErr() {
        this.form.find('#err-message').text("")
        this.form.find('#message').removeClass('form-error')
        this.form.find("#err-email").text("")
        this.form.find("#email").removeClass('form-error')
```

```
}

receiveData(data, status) {
    if(status == 400) {
        if (data["message"]) {

this.form.find('#err-message').text(data['message'])

this.form.find('#message').addClass('form-error')
        }

        if (data["email"]) {

this.form.find("#err-email").text(data['email'])

this.form.find("#email").addClass('form-error')
        }
    } else if(status == 200) {
        this.form.find('#message').val('')
        this.form.find('#email').val('')
        App.alerts.createAlert("Hvala Vam! U najkracem
roku ćemo Vam odgovoriti")
    }
}
}
```

### Dates.js

```
import $ from 'jquery';
import 'pickadate/lib/picker';
import 'pickadate/lib/picker.date';

export default function transform() {
  $(".date-input").pickadate({
    container: this,
    formatSubmit: 'yyyy/mm/dd',
    hiddenName: true
  });
}
```

### Map.js

```
const SPEED = 75
const ANIMATE_SPEED = 500
const INIT_SAMPLE = 6

const times = x => f => {
  if(x > 0) {
    f()
    times (x - 1) (f)
  }
}

class Arr {
  constructor(lst=[]) {
```

```
    this.position = 0
    this.list = lst
  }

  next() { return this.list[this.position++] }
  push(item) { this.list.push(item) }
  splice(index, length) { this.list.splice(index, length) }
  sample() {
    let randIndex = Math.floor(Math.random() *
this.list.length)
    let item = this.list[randIndex]
    this.list.splice(randIndex, 1)
    return item
  }
}

export default class Map {
  constructor() {
    if (document.querySelectorAll('.circle-star').length)
    {
      let nodes =
document.querySelectorAll('.circle-star')
      this.markers = new Arr([...nodes])
      this.samples = new Arr()
      this._sample(INIT_SAMPLE)
      this._animateSample(this.samples.next())
    }
  }
}
```

```
next() {
  return this.markers.list[this.position++]
}

animate(marker) {
  if(!marker) return;

  marker.classList.add('active');

  setTimeout(() => this.animate(this.next()), SPEED)
}

_animateRecurse() {
  let vanish = this.samples.sample()
  vanish.classList.remove('active')
  let next = this.markers.sample()
  setTimeout(() => next.classList.add('active'),
ANIMATE_SPEED)
  this.markers.push(vanish)
  this.samples.push(next)
  setTimeout(this._animateRecurse.bind(this), SPEED +
2*ANIMATE_SPEED)
}

_animateSample(sample) {
  if(!sample) {
    setTimeout(() => {
```



```
        this._animateRecurse()  
      }, ANIMATE_SPEED)  
      return  
    }  
    sample.classList.add('active')  
    setTimeout(() =>  
this._animateSample(this.samples.next()), SPEED)  
  }  
  
  _sample(size) {  
    times(size) (() =>  
this.samples.push(this.markers.sample()))  
  }  
}
```

## Search.js

```
import $ from 'jquery';

function debounce(func, wait, immediate) {
  let timeout;
  return function() {
    let context = this;
    let args = arguments;
    let later = function() {
      timeout = null;
      if (!immediate) func.apply(context, args);
    };
    let callNow = immediate && !timeout;
    clearTimeout(timeout);
    timeout = setTimeout(later, wait);
    if (callNow) func.apply(context, args);
  };
}

function template(festival) {
  return `<div class='festival'>
    <figure class='festival__slika'>
      <img src='${festival.putanja}' />
      <figcaption>
        <h2>${festival.naziv}</h2>
        <h3><i class="far fa-calendar-alt"></i>
          ${formatDate(festival.datum)}</h3>
        <h3><i class="fas fa-users"></i> 12 000</h3>
      </figcaption>
    </figure>
  </div>`
}
```

```
        </figcaption>
    </figure>
    <p>${festival.opis}</p>
</div>`;
}

function formatDate(rawDate) {
    var datum = rawDate.split(" ")[0];
    var date = new Date(datum);
    var final =
        date.getUTCDate() + " / " +
        (date.getUTCMonth()+1) + " / " +
        date.getUTCFullYear();
    return final;
}

class SearchFestival {
    initializeSearchBar() {
        this.searchInput = $("#search-festivals");

        this.searchInput.keyup(debounce(this.handleSearch.bind(this),
250))
    }

    handleSearch() {
        let val = this.searchInput.val();
        $.ajax({
            method: 'GET',
```

```
    dataType: 'JSON',
    url: `/api/search?term=${val}`,
    success: (data) => {
      if(val === '') {
        history.pushState('festivali', '', `festivali`);
      } else {
        history.pushState('festivali', '',
`festivali?term=${val}`);
      }
      this.redrawPage(data);
    }
  })
}

redrawPage(data) {
  const timing = 300;
  $('.festivali__list').html('');
  data.forEach((festival, i) => {
    let festDOM = $(template(festival));
    festDOM.hide();
    $('.festivali__list').append(festDOM);
    festDOM.fadeIn(timing + i * timing);
  });
}
}

export default new SearchFestival();
```

## Tabs.js

```
export default class Tabs {
  constructor() {
    if(document.querySelectorAll('.tabs').length) {
      this.tab = document.querySelectorAll('.tabs')[0]
      this.tabs = this.tab.querySelectorAll('.tab')
      this.active = 1
      this._listeners()
      this.matchState()
    }
  }

  matchState() {

    this.tab.parentNode.querySelectorAll('.tab-content').forEach(
      (tab) => {
        let id = tab.dataset.tab
        if(id != this.active) tab.style.display = 'none'
        else tab.style.display = 'block'
      })
  }

  handler(ev, btn) {
    ev.preventDefault()
    this.tabs.forEach((tab) =>
    tab.classList.remove('active'))
    this.active = Array.from(this.tabs).indexOf(btn) + 1
    btn.classList.add('active')
```

```

    this.matchState()
  }

  _listeners() {
    this.tabs.forEach((tab) => {
      let instance = this
      tab.addEventListener('click', function(e) {
instance.handler(e, this) }, false)
      })
    }
  }
}

```

### Validaors.js

```

import $ from 'jquery';

const VALIDATIONS = {
  mail: {
    fn: (value) =>
value.match(/^([a-z0-9\.\_\+\-]+@[a-z0-9\.\_]+\$/),
    msg: "Mail nije u dobrom formatu"
  },
  lozinka: {
    fn: (value) => value.match(/^((?=.*[!@#]).{5,100})$/),
    msg: "Lozinka mora da zadrzi bar jedan od [!, @, #]"
  },
  ime: {
    fn: (value) =>
value.match(/^([A-ZŠĐČĆŽ][a-zšđčćž]+(\s[A-ZŠĐČĆŽ][a-zšđčćž]+)+$/

```

```
),  
  msg: "Ime nije u dobrom formatu"  
},  
"not-empty": {  
  fn: (value) => value !== '',  
  msg: "Mora biti popunjeno"  
}  
};  
  
/**  
 * Generic form validator  
 */  
  
export default class Validator {  
  static validateField(input, validatorName, errField) {  
    let validator = VALIDATIONS[validatorName];  
    let value = input.val();  
    let result = validator.fn(value);  
    if(result) {  
      input.removeClass('form-error');  
      errField.text('')  
    } else {  
      input.addClass('form-error');  
      errField.text(validator.msg)  
    }  
  
    return result  
  }  
}
```

```

}

static validateEqual(inputOne, inputTwo, errField) {
    let valueOne = inputOne.val();
    let valueTwo = inputTwo.val();
    let result = valueOne === valueTwo;
    if(result) {
        inputOne.removeClass('form-error');
        errField.text('')
    } else {
        inputOne.addClass('form-error');
        errField.text("Polje se ne poklapa");
    }
    return result
}

static initializeFormValidators() {
    /**
     * First we find all the forms in the document
     * and throw away the ones without the data validator
     */
    let validatorForms = $("form").filter(function() { return
$(this).data("validator-namespace") } );
    validatorForms.each(this._handle_form);
}

/**
 * We need to check if the data-validator-name or

```



```

data-validator-same-as is present
    * and also make sure it's one of the valid filters.
    *
    * If we don't have a filter, we let the developer know
    * that he made a misatke via an usefull error message.
    */
static _compile_form_input(event, input) {
    let errF = input.parent().find('.form__errors');
    if (input.data('validator-name')) {
        let name = input.data('validator-name');

        if (!VALIDATIONS[name]) {
            throw new Error(`[${name}] doesn't exist as a valid
validator option.
            Try using one of the following:
            ${Object.keys(VALIDATIONS).join(", ")}
            `);
        }

        /**
         * No errors happaned at this point so we continue
         * validating the form
         */
        if (!Validator.validateField(input, name, errF)) {
            event.preventDefault();
        }
    } else if (input.data('validator-same-as')) {
        let asInput = $(input.data('validator-same-as'));
    }
}

```

```
    if (!Validator.validateEqual(input, asInput, errF)) {
      event.preventDefault()
    }
  } else {
    console.error(input);
    throw new Error("Input does not have a data-validator-  
class");
  }
}

static _handle_form() {
  let form = $(this);
  let inputs = form.find('input, textarea');
  form.submit(function(event) {
    inputs.each(function() {
      let input = $(this);
      Validator._compile_form_input(event, input)
    })
  })
}
}

export const validations = VALIDATIONS;
```

## Vote.js

```
import $ from 'jquery';

export default function initVoteSystem() {
  $(".vote-button").click(function(e) {
    e.preventDefault();
    const id = $(this).data('id');
    $(this).attr('disabled', 'true');
    $.ajax({
      dataType: 'json',
      method: 'POST',
      url: '/api/glasaj',
      data: {
        id: id
      },
      success: (data) => {
        App.alerts.createAlert(data.message);
      },
      error: (xhr) => {
        let ovde = `
```



```
    ignore: [/vendor/]
  },
  sass: {
    mode: "native",
  }
},
modules: {
  autoRequire: {
    "js/app.js": ["js/app"]
  }
},
npm: {
  enabled: true,
  styles: {
    pickadate: [
      'lib/themes/default.css',
      'lib/themes/default.date.css'
    ],
    select2: [
      '/dist/css/select2.css'
    ]
  }
}
};
```

## Package.json

```
{
  "repository": {},
  "licence": "MIT",
  "scripts": {
    "deploy": "brunch build --production",
    "watch": "brunch watch"
  },
  "devDependencies": {
    "auto-reload-brunch": "^2.7.1",
    "babel-brunch": "6.1.1",
    "brunch": "^2.10.12",
    "clean-css-brunch": "2.10.0",
    "uglify-js-brunch": "2.10.0"
  },
  "dependencies": {
    "css-brunch": "^2.10.0",
    "jquery": "^3.3.1",
    "pickadate": "^3.5.6",
    "sass-brunch": "^2.10.4",
    "select2": "^4.0.6-rc.1"
  }
}
```

**Constants.php**

```
<?php
/**
 * Load a setting from a configuration env.json file
 */
function env($setting) {
    $settings = json_decode(file_get_contents(PROJECT_ROOT
. "/env.json"), true);
    if(isset($settings[$setting])) {
        return $settings[$setting];
    }
}

/**
 * File defining all constants that will be used
 * in the project
 */
define("PROJECT_ROOT", dirname(dirname(__FILE__)));
define("TEMPLATE_DIR", PROJECT_ROOT . '/views');
define("API_DIR", PROJECT_ROOT . '/api');
define("MODEL_DIR", PROJECT_ROOT . '/models');
define("DB_HOST", env('host'));
define("DB_DATABASE", 'festival_baza');
define("DB_USERNAME", env('username'));
define("DB_PASSWORD", env('password'));
define("PAGINATION", 6);
```

### Database.php

```
<?php
    require_once "constants.php";
    $host = DB_HOST;
    $name = DB_DATABASE;
    $dsn = "mysql:host=$host;dbname=$name;charset=utf8";
    try {
        $conn = new PDO($dsn, DB_USERNAME, DB_PASSWORD);
        $conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
        $conn->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,
PDO::FETCH_OBJ);
    } catch(PDOException $e) {
        echo $e->getMessage();
    }
}
```

### Router.php

```
<?php
    require_once MODEL_DIR . "/links.php";

    /**
     * Resolves the route using the database
     */
    function resolveRoute(PDO $conn) {
        $links = fetchLinks($conn);
        $uri    = $_SERVER['REQUEST_URI'];
        $route = $t = substr($uri, strpos($uri, '/')+1);
        if(strpos($t, "?") !== false) {
```



```

        $route = substr($t, 0, strpos($t, '?'));
    }
    $patt = '/^api.*$/';
    if(preg_match($patt, $route)) {
        return [
            "route" => forApi($route),
            "type"  => "api"
        ];
    }
    foreach($links as $link) {
        if($link->name == strtolower($route)) {
            return [
                "route" => $link->name,
                "type"  => "web"
            ];
        }
    }
    return [
        "route" => "pocetna",
        "type"  => "web"
    ];
}

/**
 * Parser the API route
 */
function forApi($unparsed) {
    $matches = [];

```

```

    $reg = '/^api(\\/[^\?]*)/';
    preg_match_all($reg, $unparsed, $matches);
    $str = $matches[1][0];
    return substr($str, 1);
}

```

### Func.php

```

<?php

/**
 * @param PDO $conn the connection to the database
 * @param string $table the table name
 * @return bool|PDOStatement returns all records for a given
table
 */
function fetchAll(PDO $conn, string $table) {
    return $conn->query("SELECT * FROM $table");
}

function runSafeQuery(PDO $conn, $query, array $bindings) {
    $stmt = $conn->prepare($query);
    foreach($bindings as $key => &$val) {
        $stmt->bindParam(":$key", $val);
    }
    $stmt->execute();
}

function safeQuery(PDO $conn, $query, array $bindings,

```

```
$fetchOne=false) {  
    $stmt = $conn->prepare($query);  
    foreach($bindings as $key => &$val) {  
        $stmt->bindParam(":$key", $val);  
    }  
    $stmt->execute();  
    if($fetchOne) {  
        return $stmt->fetch();  
    }  
    return $stmt->fetchAll();  
}  
  
function insertValidate($insertParams, $validations,  
Callable $transform, $extra=[]) {  
    $state = [];  
    foreach($insertParams as $param) {  
        if(isset($_POST[$param])) {  
            $state[$param] = $_POST[$param];  
        } else {  
            throw new Error("Param $param not provided!");  
        }  
    }  
  
    foreach($validations as $field => $regex) {  
        $val = $state[$field];  
        if(!preg_match($regex, $val)) {  
            throw new Error("Param $field nije dobar");  
        }  
    }  
}
```

```

    }
    return $transform($state, $extra);
}

/**
 * Dynamically maps data from the models
 * into a query
 */
function dataToColumns($data) {
    $names = array_keys($data);
    $placeholders = implode(",", array_map(function($e) {
        return ":$e";
    }, $names));
    return [implode(",", $names), $placeholders];
}

function dataToUpdateColumns($data) {
    $names = array_keys($data);
    $placeholders = implode(", ", array_map(function($e) {
        return "$e=: $e";
    }, $names));
    return $placeholders;
}

function insert(PDO $conn, $insertStructure) {
    $tableName = $insertStructure['tableName'];
    list($names, $placeholders) =

```

```

dataToColumns($insertStructure['data']);

$query = "INSERT INTO $tableName (" . $names . ")" .
        " VALUES(" . $placeholders . ")";

$stmt = $conn->prepare($query);
foreach($insertStructure['data'] as $key => &$value) {
    $stmt->bindParam(":$key", $value);
}
try{
    $stmt->execute();
    return $conn->lastInsertId();
} catch(PDOException $e) {
    return false;
}

}

function update(PDO $conn, $updateStructure, $id) {
    $tableName = $updateStructure['tableName'];
    $placeholders =
dataToUpdateColumns($updateStructure['data']);

    $query = "UPDATE $tableName SET " . $placeholders . "
WHERE id=:id";
    $stmt = $conn->prepare($query);
    $stmt->bindParam(":id", $id);
    foreach($updateStructure['data'] as $key => &$value) {

```

```
        $stmt->bindParam(":$key", $value);
    }
    try {
        $stmt->execute();
    } catch(PDOException $e) {
        var_dump($e->getMessage());
    }
}

function hasError(&$var, $error, $errText) {
    if(isset($var) && isset($var[$error])) {
        return $errText;
    }
    return '';
}

/**
 * @param string $location
 * A simple wrapper which
 */
function redirect(string $location) {
    header("Location: $location");
}

/**
 * @return bool
 * Returns weather or not is a user logged
 * in with session.
```

```

*/
function userLogged() {
    if(isset($_SESSION['user'])) {
        return true;
    }
    return false;
}

function adminLogged() {
    if(userLogged() && $_SESSION['user']->naziv == 'admin') {
        return true;
    }
    return false;
}

/*===== IMAGE MANIPULATION =====*/
function resize_image($file, $target, Callable $callback) {
    list($originalWidth, $originalHeight) =
getimagesize($file);
    $ratio = $originalWidth / $originalHeight;
    list($targetWidth, $targetHeight) = $callback($ratio,
$target);
    $originalImage = imagecreatefromjpeg($file);
    $targetImage = imagecreatetruecolor($targetWidth,
$targetHeight);
    imagecopyresampled($targetImage, $originalImage,
    0, 0, 0, 0,
    $targetWidth, $targetHeight,

```

```
        $originalWidth, $originalHeight);
    imagejpeg($targetImage, $file, 100);
}

function resize_image_by_width($ratio, $targetWidth) {
    if($ratio >= 1) {
        $targetHeight = $targetWidth / $ratio;
    } else {
        throw new Exception("Wrong image form suplled");
        return;
    }
    return [$targetWidth, $targetHeight];
}

function resize_image_by_height($ratio, $targetHeight) {
    if($ratio < 1) {
        $targetWidth = $targetHeight * $ratio;
    } else {
        throw new Exception("Wrong image format supplied");
        return;
    }
    return [$targetWidth, $targetHeight];
}
```



## Festivals.php

```
<?php

/**
 * Form requirements
 */
function getFestivalParams() {
    return [
        'naziv',
        'datum',
        'opis'
    ];
}

function getFestivalValidations() {
    $reNaziv = '/^{1,100}$/';

    return [
        'naziv' => $reNaziv
    ];
}

/**
 * Transforms the state into
 * what should be inserted in the db
 */
function festivalTransform($state, $extra) {
    return [
```

```

        "tableName" => "festivali",
        "data" => array_merge([
            "naziv" => $state["naziv"],
            "datum" => $state["datum"],
            "opis"  => $state["opis"],
        ], $extra)
    ];
}

function festivalPrettyDate($fullDbTimestamp) {
    $datum = explode(" ", $fullDbTimestamp)[0];
    list($year, $month, $day) = explode("-", $datum);
    return date("d / m / Y", mktime(0, 0, 0, $month, $day,
$year));
}

```

### Links.php

```

<?php
function fetchLinks(PDO $conn) {
    return fetchAll($conn, "linkovi");
}

```

## Users.php

```
<?php

/**
 * Form requirements
 */
function getUserParams() {
    return [
        "email",
        "imeprezime",
        "lozinka"
    ];
}

function getUserValidations() {
    $reUserName =
'/^[A-ZŠĐČĆŽ][a-zšđćčž]+(\s[A-ZŠĐČĆŽ][a-zšđćčž]+)+$/';
    $reUserEmail = '/^[a-z0-9\._\+\-]+@[a-z0-9\._]+$/';
    $reUserPass = '/^(?=.*[$%^@#]).{5,100}$/';

    return [
        "email" => $reUserEmail,
        "lozinka" => $reUserPass,
        "imeprezime" => $reUserName
    ];
}

/**
```

```
* Transforms the state into
* what should be inserted in the db
*/
function userTransform($state) {
    $username = makeUserName($state['imeprezime']);
    $password = md5($state['lozinka']);
    $datum = date("Y-m-d H:i:s");

    return [
        "tableName" => "korisnici",
        "data" => [
            "email" => $state["email"],
            "korisnicko_ime" => $username,
            "password_hash" => $password,
            "datum_registracije" => $datum,
            "id_role" => 2
        ]
    ];
}

function makeUserName($imeprezime) {
    $parts = explode(" ", $imeprezime);
    $downcase = array_map(function ($part) {
        return strtolower($part);
    }, $parts);
    return implode('.', $downcase);
}
```

## .htaccess

```
RewriteEngine On

<filesMatch "\.(html|htm|js|css)$">
  FileETag None
  <ifModule mod_headers.c>
    Header unset ETag
    Header set Cache-Control "max-age=0, no-cache, no-store,
must-revalidate"
    Header set Pragma "no-cache"
    Header set Expires "Wed, 11 Jan 1984 05:00:00 GMT"
  </ifModule>
</filesMatch>

RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule . index.php
```

## Index.php

```
<?php
    ob_start();
    session_start();

    // Configuration
    require_once "../config/constants.php";
    require_once "../config/router.php";

    // Database
    require_once PROJECT_ROOT . "/config/database.php";

    // Models
    require_once MODEL_DIR . "/helpers/func.php";
    require_once MODEL_DIR . "/links.php";
    require_once MODEL_DIR . "/users.php";
    require_once MODEL_DIR . "/festivals.php";
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Festivian</title>

    <?php include_once(TEMPLATE_DIR . "/includes.php"); ?>
</head>
<body>
```

```

<?php
    $route = resolveRoute($conn);
    if($route['type'] == 'web') {
        include_once(TEMPLATE_DIR . "/header.php");
        include_once(TEMPLATE_DIR .
"/{$route['route']}.php");
        include_once(TEMPLATE_DIR . "/footer.php");
    } else if($route['type'] == 'api') {
        ob_clean();
        header("Content-Type: appliaction/json");
        require_once(API_DIR . "{$route['route']}.php");
        exit();
    }

?>
</body>
</html>

```

### Admin\_nav.php

```

<a class='big-link' href="admin">Novi Festival</a>
<a class='big-link' href='admin_edit'>Edituj Festival</a>
<a class='big-link' href="admin_anketa">Statistika Ankete</a>
<a class='big-link' href="admin_korisnici">Korisnici</a>

```

## Admin.php

```
<?php if(!adminLogged()) redirect("/") ?>

<?php if(isset($_POST['festival-submit'])) {
    $slika = $_FILES['slika'];
    $type = $slika['type'];
    $size = $slika['size'];
    $naziv = $slika['name'];
    $tmp = $slika['tmp_name'];

    $validTypes = [
        "image/jpeg",
        "image/jpg"
    ];
    $errors = [];

    if(!in_array($type, $validTypes)) {
        $errors[] = "Slika nije u dobrom formatu, mora biti (mora biti jpg ili jpeg)";
    }

    if($size > 4000000) {
        $errors[] = "Slika mora biti manja od 4MB";
    }

    if(empty($errors)) {
        $noviNaziv = time() . "_" . $naziv;
        $putanja = PROJECT_ROOT . "/public/assets/images/" .
```



```

$noviNaziv;
    resize_image($tmp, 450, 'resize_image_by_width');
    $webPutanja = "/assets/images/" . $noviNaziv;

    if(move_uploaded_file($tmp, $putanja)) {
        $toInsert = insertValidate(getFestivalParams(),
getFestivalValidations(),
        'festivalTransform', ["putanja" => $webPutanja]);
        insert($conn, $toInsert);
    }
}
}??>

<div class='site-form'>
    <?php include_once "admin/admin_nav.php"; ?>
    <h1 class='site-form__title'>Novi festival</h1>
    <div class='site-form__wrap'>
        <form action='/admin' enctype='multipart/form-data'
method='post' class='form' data-validator-namespace="login">

            <div class='form__group'>
                <input
                    type='text'
                    class="form__control <?= hasError($error, 'email',
'form-error') ?>"
                    placeholder='Naziv festivala'
                    data-validator-name='not-empty'
                    name='naziv'

```

```
    />
    <span class='form__errors'>
  </span>
</div>

<div class='form__group'>
  <input
    type='text'
    class="form__control date-input"
    name='datum'
    data-validator-name='not-empty'
    placeholder="Datum održavanja"
  />
  <span class='form__errors'>
  </span>
</div>

<div class='form__group'>
  <textarea
    class='form__control'
    placeholder='Description'
    data-validator-name='not-empty'
    name='opis'
  ></textarea>
  <span class='form__errors'></span>
</div>

<div class='form__group'>
```

```
<input
    type='file'
    class='form__control'
    placeholder='Description'
    data-validator-name='not-empty'
    name='slika'
/>
<span class='form__errors'>
    <?php if(isset($errors)): ?>
        <?= implode(", ", $errors) ?>
    <?php endif ?>
</span>
</div>

    <button name='festival-submit' class='form__submit
form__submit--primary'>
        Postavi
    </button>
</form>
</div>
</div>
```

## Admin\_anketa.php

```
<?php if(!adminLogged()) redirect("/") ?>

<div class='site-form'>
    <?php include_once 'admin/admin_nav.php' ?>
    <h1 class='site-form__title'>Statistika ankete</h1>
    <div class='site-form__wrap'>
        <?php
            $upit = "select f.naziv, count(*) as broj_glasova from
glasanja g
                        join festivali f on g.id_festival = f.id
                        group by f.naziv
                        order by broj_glasova desc
                        limit 5";

            $stats = safeQuery($conn, $upit, []);
        ?>
        <table class='vote-table'>
            <thead>
                <tr>
                    <th>Ime festivala</th>
                    <th>Broj glasova</th>
                </tr>
            </thead>
            <tbody>
                <?php foreach($stats as $s): ?>
                    <tr>
                        <td><?= $s->naziv ?></td>
```

```

        <td><?= $s->broj_glasova ?></td>
    </tr>
    <?php endforeach; ?>
</tbody>
</table>
</div>
</div>

```

### Admin\_edit.php

```

<?php if(!adminLogged()) redirect("/") ?>

<?php
if(isset($_POST['festival-obrisi'])) {
    $id = $_POST['id'];
    $query = "DELETE FROM festivali WHERE id=:id";
    safeQuery($conn, $query, ["id" => $id], true);
}

if(isset($_POST['festival-edit'])) {
    $id = $_POST['festID'];

    if(isset($_FILES['slika']) && $_FILES['slika']['size'] >
0) {
        $slika = $_FILES['slika'];
        $type = $slika['type'];
        $size = $slika['size'];
        $naziv = $slika['name'];
    }
}

```

```
$tmp    = $slika['tmp_name'];

$validTypes = [
    "image/jpeg",
    "image/jpg"
];
$errors = [];

if(!in_array($type, $validTypes)) {
    $errors[] = "Slika nije u dobrom formatu, mora biti
(mora biti jpg ili jpeg)";
}

if($size > 4000000) {
    $errors[] = "Slika mora biti manja od 4MB";
}

if(empty($errors)) {
    // TODO: Obrisi staru
    $noviNaziv = time() . "_" . $naziv;
    $putanja = PROJECT_ROOT . "/public/assets/images/" .
$noviNaziv;
    resize_image($tmp, 450, 'resize_image_by_width');
    $webPutanja = "/assets/images/" . $noviNaziv;

    if(move_uploaded_file($tmp, $putanja)) {
        $toUpdate = insertValidate(getFestivalParams(),
getFestivalValidations(),
```

```

        'festivalTransform', ["putanja" => $webPutanja]);
        update($conn, $toUpdate, $id);
    }
}
} else {
    $toUpdate = insertValidate(getFestivalParams(),
getFestivalValidations(),
    'festivalTransform', []);
    update($conn, $toUpdate, $id);
}
}
?>

<div class='site-form site-form--admin_edit'>
    <?php include_once 'admin/admin_nav.php' ?>
    <h1 class='site-form__title'>Edituj festival</h1>
    <div class='site-form__wrap'>
        <?php
            $festivali = safeQuery($conn, "select * from festivali",
[]);
        ?>
        <select id="fetival-select">
            <option value='0'>Izaberite</option>
            <?php foreach($festivali as $fest): ?>
                <option value='<?= $fest->id
?>'><?=$fest->naziv?></option>
            <?php endforeach ?>
        </select>

```

```
<form action='/admin_edit' enctype='multipart/form-data'
method='post' class='form' data-validator-namespace="login">
  <input type='hidden' id='festID' name='festID' />

  <div class='form__group'>
    <input
      type='text'
      class="form__control disabled"
      placeholder='Naziv festivala'
      data-validator-name='not-empty'
      name='naziv'
      id='naziv'
      disabled
    />
    <span class='form__errors'>
    </span>
  </div>

  <div class='form__group'>
    <input
      type='text'
      class="form__control date-input disabled"
      name='datum'
      id='datum'
      data-validator-name='not-empty'
      placeholder="Datum održavanja"
      disabled
    />
  </div>
</form>
```



```
    />
    <span class='form__errors'>
  </span>
</div>

<div class='form__group'>
  <textarea
    class='form__control disabled'
    placeholder='Description'
    data-validator-name='not-empty'
    name='opis'
    id='opis'
    disabled
  ></textarea>
  <span class='form__errors'></span>
</div>

<div class='form__group'>
  <input
    type='file'
    class='form__control disabled'
    placeholder='Description'
    name='slika'
    id='slika'
    disabled
  />
  <span class='form__errors'>
    <?php if(isset($errors)): ?>
```

```

        <?= implode(", ", $errors) ?>
    <?php endif ?>
</span>
</div>

<img id='preview' />

<button id='izmeni' name='festival-edit'
class='form__submit disabled form__submit--primary' disabled>
    Izmeni
</button>
<button id='obrisi' name='festival-obrisi'
class='form__submit disabled form__submit--delete' disabled>
    Obriši
</button>
</form>
</div>
</div>

```

### Admin\_korisnici.php

```

<?php if(!adminLogged()) redirect("/") ?>

<?php

    if(isset($_POST['korisnik-edit'])) {

    }

?>

```

```

<div class='site-form site-form--korisnik_edit'>
  <?php include_once 'admin/admin_nav.php' ?>
  <h1 class='site-form__title'>Edituj Korisnike</h1>
  <div class='site-form__wrap'>
    <?php
      $korisnici = safeQuery($conn, "select k.id, k.email,
k.korisnicko_ime from korisnici k where id_role<>1", []);
    ?>
    <select id="korisnik-select">
      <option value='0'>Izaberite</option>
      <?php foreach($korisnici as $korisnik): ?>
        <option value='<?=$korisnik->id
?>'><?=$korisnik->email ?> (<?=$korisnik->korisnicko_ime
?>)</option>
      <?php endforeach ?>
    </select>
    <form action='/admin_edit' method='post' class='form'
data-validator-namespace="login">
      <input type='hidden' id='korID' name='kodID' />

      <div class='form__group'>
        <input
          type='text'
          class="form__control disabled"
          placeholder='Email'
          data-validator-name='email'

```

```
        name='email'
        id='email'
        disabled
    />
    <span class='form__errors'>
    </span>
</div>

<div class='form__group'>
    <input
        type='text'
        class="form__control disabled"
        name='kor-ime'
        id='kor-ime'
        data-validator-name='not-empty'
        placeholder="Korisničko ime"
        disabled
    />
    <span class='form__errors'>
    </span>
</div>

<div class='form__group'>
    <input
        class='form__control disabled'
        placeholder='Lozinka'
        name='lozinka'
        id='lozinka'
```

```

        disabled
    />
    <span class='form__errors'></span>
</div>

    <button id='izmeni' name='festival-edit'
class='form__submit disabled form__submit--primary' disabled>
    Izmeni
    </button>
    <button id='obrisi' name='festival-obrisi'
class='form__submit disabled form__submit--delete' disabled>
    Obriši
    </button>
</form>
</div>
</div>

```

### Autor.php

```

<div class='autor'>
    <h1 class='text-center'>Stefan Kuprešak</h1>
    <div class='autor__image'>
        <img src='assets/images/moja.jpg'>
        <h2>Big Data // Full Stack Web // Open Source
Software</h2>
    </div>
    <div class='autor__bio'>
        <p>Zdravo! S obzirom da smo svi mi "ambiciozni
programeri", na ovaj ili onaj način

```

ovog puta ću preskočiti formalnosti i preći na stvar.  
Bavim se Big Data developmentom

u programskom jeziku **Elixir** kao i  
interesovanju za moderne web frameworke

za full stack development poput React-a, Laravel-a i  
eliksirovog Feniksa.

</p>

<p>

Posedujem gomilu iskustva na velikom spektru polja u IT  
sektoru, uključujući rad na mnogim  
projektima i to na izgradnji i skaliranju istih od nule.  
Trenutno sam zaposlen u firmi

BlueGrid, gde učestvujem na mnogim internim i spoljnim  
projektima. Takođe, imam veliko

interesovanje za experimentalne jezike poput GoLang-a i  
njemu sličnim pa povremeno poklanjam

deo svog vremena istraživanju takvih tehnologija.

</p>

<p>

**Festivian** je platforma koju sam  
inicijalno zamislio da sadrži mnogo više

zanimljivih stvari, međutim na žalost zbog vremenskih  
ograničenja nisam uspeo da sprovedem

krajnu zamisao već samo sastaviti ovaj kratak prototip  
nekih funkcionalnosti sa kranje platforme.

</p>

</div>

</div>

## Festivali.php

```

<div class="festivali">
  <form class='form'>
    <input id='search-festivals' class='form__control'
type='search' placeholder='Pretraži festivale...' />
  </form>
  <div class="festivali__wrap">
    <h1>Svi festivali</h1>
    <div class='festivali__list'>
      <?php

        $page = 1;
        if(isset($_GET['page'])) {
          $page = $_GET['page'];
        }

        $od = ($page-1) * PAGINATION;
        $do = PAGINATION;

        $count = safeQuery($conn, "select count(*) as num from
festivali", [], true)->num;
        $festivali = safeQuery($conn, "select * from festivali
limit $od, $do", []);
        $pages = ceil($count / PAGINATION);
      ?>
      <?php foreach($festivali as $festival): ?>
        <div class='festival'>
          <figure class='festival__slika'>

```

```

        <img src='<?=$festival->putanja ?>' />
        <figcaption>
            <div class='festival__title'>
                <h2><?=$festival->naziv ?></h2>
                <?php if(userLogged()): ?>
                    <a data-id='<?=$festival->id ?>'
class='vote-button' href='#'>Glasaj</a>
                <?php endif ?>
            </div>
            <h3><i class="far fa-calendar-alt"></i>
                <?=$festivalPrettyDate($festival->datum)
?></h3>
            <!-- <h3><i class="fas fa-users"></i> 12
000</h3> -->
        </figcaption>
    </figure>
    <p><?=$festival->opis ?></p>
</div>
<?php endforeach ?>
</div>
</div>
<div class='pagination'>
    <?php
        for($i=1; $i<=$pages; $i++):?>
            <a href="<?=$festival->url?>page=$i" ?>><?=$i ?></a>
        <?php endfor ?>
    </div>
</div>

```



## Footer.php

```

<footer>
  <div class='footer-wrap'>
    <div class='footer-center'>
      <div class='col'>
        <h1>O Festivian</h1>
        <p>
          Festivian platforma je moderna, skalabilna
          platforma koja omogućava brz, lak, i
          ekonomičan, jednostavan
          pristup festivalima i korisnicima i
          organizatorima
        </p>
      </div>
      <div class='col-2'>
        <h1>Konektujte se</h1>
        <div class='icons-connected'>
          <i class="fab fa-facebook-f"></i>
          <i class="fab fa-linkedin-in"></i>
          <i class="fab fa-twitter"></i>
          <i class="fab fa-instagram"></i>
        </div>
      </div>
      <div class='col logo-foot'>
        <h1>Domen</h1>
        <a href='https://bluegrid.io/'
target='_blank'>
          <img src='assets/images/bluegrid.png'

```



```

        href='<?= "/" . $link->name ?>'
        target='<?= $link->target_blank == 1 ?
"_blank" : "_self" ?>'
    >
        <?= $link->labela ?>
    </a>
    <?php endif ?>
</li>
<?php endforeach ?>
</ul>
</nav>
</header>

```

### Includes.php

```

<!-- Styles & CSS dependencies -->
<meta name="viewport" content="width=device-width,
initial-scale=1">
<link
href="https://fonts.googleapis.com/css?family=Montserrat|Perm
anent+Marker" rel="stylesheet">
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.0.13/css/all.cs
s"
integrity="sha384-DNOHZ68U8hZfKXOortjWvjxusGo9WQnrNx2sqG0tfsgh
AvtVlRW3tvkXWZh58N9jp" crossorigin="anonymous">
<link rel="stylesheet" type="text/css" media="screen"
href="assets/css/app.css" />

```

### Logout.php

```
<?php
unset($_SESSION['user']);
session_destroy();
redirect('/');
```

### Pocetna.php

```
<div id="center">
  <div id="hero-title">
    <h1>
      Festivian
    </h1>
    <p>
      Pronađite bilo koji festival u Srbiji.
      Brzo, lako i jednostavno!
    </p>
    <p>
      Planirajte unapred i rezervišite karte
      već danas!
    </p>
    <a href='/festivali' class='btn-pretraizi'>Počni
pretragu</a>
  </div>
  <div id='map'>
    <?= file_get_contents(PROJECT_ROOT .
"/public/assets/images/serbia.svg"); ?>
  </div>
</div>
<div class='bg-wrap p-8rem'>
```

```
<div id="acc-tabs" class='tabs'>
  <div class='tab active'>
    <i class="fas fa-user"></i>
  </div>
  <div class='tab'>
    <i class="fas fa-users"></i>
  </div>
</div>
<div class="tab-content" data-tab='1'>
  <h1>Festivian za korisnike</h1>
  <ul>
    <li>
      <i class='fas fa-star'></i>
      Pretražite najaktuelnije festivale u Srbiji
    </li>
    <li>
      <i class='fas fa-star'></i>
      Rezervišite karte po 20% jeftinijoj ceni
    </li>
    <li>
      <i class='fas fa-star'></i>
      Proverite utiske i ocene festiavla
    </li>
  </ul>
</div>
<div class="tab-content" data-tab='2'>
  <h1>Festivian za organizatore</h1>
  <ul>
```

```

        <li>
            <i class='fas fa-star'></i>
            Organizujte vaš festival uz pomoć naših
volontera
        </li>
        <li>
            <i class='fas fa-star'></i>
            Marketing preko naše platforme uključen u
ponudu
        </li>
        <li>
            <i class='fas fa-star'></i>
            Ostavite planiranje na nama
        </li>
    </ul>
</div>
</div>
<div class='thoughts-slider'>
    <blockquote>
        <p>"Rad za Festivian platformom je bio pun pogodak.
Brzo organizovanje, odličan
sistem naplate i sve na jednom mestu. Sve
pohvale!"</p>
        <span>-Belgrade Beer Fest</span>
    </blockquote>
</div>
<div class='bg-wrap'>

```

```
<div class='contact-form'>
  <h1>Pišite nam</h1>
  <form action='#'>
    <div class='form-wrap'>
      <input type='text' id='email'
placeholder='Mail' />
      <span id='err-email'
class='err-tooltip'></span>
    </div>
    <div class='form-wrap'>
      <textarea id='message'
placeholder='Poruka'></textarea>
      <span id='err-message'
class='err-tooltip'></span>
    </div>
    <button>Posašalji</button>
  </form>
</div>
</div>
```

## Prijava.php

```
<?php if(userLogged()) {  
    redirect("/");  
}?>  
  
<div class='site-form'>  
    <h1 class='site-form__title'>Prijava</h1>  
    <?php  
        if(isset($_POST['login-submit'])) {  
            $email = $_POST['email'];  
            $lozinka = $_POST['lozinka'];  
            $query = "select k.*, r.naziv from korisnici k join  
role r on k.id_role = r.id  
                    where k.email=:email and  
k.password_hash=:lozinka";  
            $user = safeQuery($conn, $query, [  
                "email" => $email,  
                "lozinka" => md5($lozinka)  
            ], true);  
  
            if($user) {  
                $_SESSION['user'] = $user;  
                redirect("/");  
            } else {  
                $error['email'] = 'Email/Lozinka nije dobra';  
            }  
        }  
    }  
?>
```



```
<div class='site-form__wrap'>
  <form action='/prijava' method='post' class='form'
data-validator-namespace="login">

    <div class='form__group'>
      <input
        type='text'
        class="form__control <?= hasError($error, 'email',
'form-error') ?>"
        placeholder='Mail'
        data-validator-name='mail'
        name='email'
      />
      <span class='form__errors'>
        <?= hasError($error, 'email', $error['email']) ?>
      </span>
    </div>

    <div class='form__group'>
      <input
        type='password'
        class="form__control"
        placeholder='Lozinka'
        data-validator-name='lozinka'
        name='lozinka'
      />
      <span class='form__errors'></span>
    </div>
```

```

        <button name='login-submit' class='form__submit
form__submit--primary'>
            Prijavi se
        </button>
    </form>
    <p>Nemas nalog? <a href='/registracija'>Registruj
se</a></p>
</div>
</div>

```

### Registracija.php

```

<?php if(userLogged()) {
    redirect("/");
} ?>
<div class='site-form'>
    <h1 class='site-form__title'>Registracija</h1>
    <?php
        if(isset($_POST['register-submit'])) {
            try {
                $toInsert = insertValidate(getUserParams(),
getUserValidations(), 'userTransform');
                if($id = insert($conn, $toInsert)) {
                    $query = "select k.*, r.naziv from korisnici k
join role r on k.id_role = r.id where k.id=:id";
                    $user = safeQuery($conn, $query, ["id" => $id],
true);
                    $_SESSION['user'] = $user;

```

```
        redirect("/");
    } else {
        $error['email'] = 'Mail adresa je vec zauzeta';
    }
} catch(Exception $e) {
    echo $e->getMessage();
}
}
?>
<div class='site-form__wrap'>
    <form action='/registracija' method='post' class='form'
data-validator-namespace="login">

        <div class='form__group'>
            <input
                type='text'
                class="form__control <?= hasError($error, 'email',
'form-error') ?>"
                placeholder='Email'
                data-validator-name='mail'
                name='email'
            />
            <span class='form__errors'>
                <?= hasError($error, 'email', $error['email']) ?>
            </span>
        </div>

        <div class='form__group'>
```

```
<input
  type='text'
  class="form__control"
  placeholder='Ime i prezime'
  data-validator-name='ime'
  name='imeprezime'
/>
<span class='form__errors'></span>
</div>

<div class='form__group'>
  <input
    id='lozinka'
    type='password'
    class="form__control"
    placeholder='Lozinka'
    data-validator-name='lozinka'
    name='lozinka'
  />
  <span class='form__errors'></span>
</div>
<div class='form__group'>
  <input
    type='password'
    class="form__control"
    placeholder='Lozinka ponovo'
    data-validator-same-as='#lozinka'
  />
```

```
        <span class='form__errors'></span>
    </div>
    <button name='register-submit' class='form__submit
form__submit--primary'>
        Registruj se
    </button>
</form>
    <p>Imaš nalog? <a href='/prijava'>Prijavi se</a></p>
</div>
</div>
```

### .gitignore

```
# Ignore the env json file
env.json

# Ignore node modules
assets/node_modules

# Ignore vscode
.vscode

# Ignore idea folder
.idea
```

**Env.json**

```
{  
  "username": "stefan",  
  "password": "stefke",  
  "host": "festivian.bluegrid.io"  
}
```