Web aplikacija za upravljanje plesnim studiom

Matea, Benčić

Undergraduate thesis / Završni rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:923718

Rights / Prava: Attribution-NonCommercial 3.0 Unported/Imenovanje-Nekomercijalno 3.0

Download date / Datum preuzimanja: **2024-04-17**

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Repository / Repozitorij:

Faculty of Organization and Informatics - Digital Repository

zir.nsk.hr

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Matea Golub

WEB APLIKACIJA ZA UPRAVLJANJE PLESNIM STUDIOM

ZAVRŠNI RAD

Sisak, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Matea Golub

Matični broj: 35918/14–izv.

Studij: Primjena informacijske tehnologije u poslovanju

WEB APLIKACIJA ZA UPRAVLJANJE PLESNIM STUDIOM

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Dragutin Kermek

Sisak, rujan 2022.

Matea Golub

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Cilj ovog završnog rada je izrada web aplikacije za upravljanje plesnim studiom. U radu je opisan proces razvoja web aplikacije i tehnologije koje su korištene pri tom procesu. Aplikacija je izrađena kako bi omogućila pristup informacijama o plesnom studiju te polaznicima uvid u raspored termina tečajeva.

U današnje vrijeme gotovo svi imaju pristup internetu te vrlo lako mogu pristupiti web aplikaciji. Pomoću web aplikacije za plesni studio korisnici mogu saznati korisne informacije o plesnom studiju, pregledati tečajeve i termine te se prijaviti ili registrirati kako bi mogli vidjeti obavijesti.

Pri izradi aplikacije korišten je programski jezik PHP i MySQL baza podataka te phpMyAdmin sučelje. Aplikacija je razvijena pomoću Symfony razvojnog okvira. Symfony znatno olakšava izradu aplikacije zbog svojih funkcionalnosti. Alat koji je korišten za izradu ove web aplikacije je Visual Studio Code.

Ključne riječi: web; php; web aplikacija; css; html; upravljanje; javascript

Sadržaj

1. Uvod 1
2. Tehnologije razvoja aplikacije 2
2.1. XAMPP 2
2.2. PHP 2
2.3. MySQL 3
2.4. MariaDB 3
2.5. Symfony 4
3. Baza podataka 7
3.1. Era dijagram 9
4. Instalacija alata 10
4.1. Visual Studio Code 10
4.2. PowerShell 10
4.3. Composer 11
4.4. Git 12
4.5. Scoop 12
5. Symfony aplikacija 13
5.1. Webpack Encore 14
5.2. Bootstrap 15
5.3. Baza podataka i Doctrine ORM 15
5.3.1. Konfiguracija baze podataka 16
5.3.2. Entiteti 16
5.4. EasyAdmin i CRUD upravljači 20
5.5. Polja 24
5.6. Sigurnost 25
5.7. Konačan izgled aplikacije 28
6. Zaključak 31
Popis literature 32
Popis slika 33

1. Uvod

Unazad nekoliko godina sve je veći broj web aplikacija. Kako se zbog trenutne situacije u svijetu uzrokovane pandemijom COVID-19 većina poslova morala odrađivati od kuće, nastala je mnogo veća potražnja za web aplikacijama. Shodno tome raste i broj tehnologija za razvoj aplikacija.

Ova aplikacija je izrađena u svrhu olakšavanja korisnicima da bržim i lakšim putem dođu do potrebnih informacija o plesnom studiju na jednom mjestu. Pomoću aplikacije budući korisnici imaju uvid u raspored termina plesova iz kojeg mogu saznati vrijeme i mjesto održavanja treninga ili nastupa, a polaznici mogu vidjeti i obavijesti vezane za plesni studio. Također, aplikacija sadrži informacije vezane za plesni studio te galeriju slika sa nastupa i videozapise.

Nakon kratkog uvoda u kojem je objašnjena svrha aplikacije i ideja, u drugom poglavlju će biti opisane tehnologije koje su korištene pri razvoju aplikacije. Ukratko će biti opisan razvojni okvir Symfony pomoću kojeg je razvijena aplikacija te općenito koje su zajedničke značajke razvojnih okvira.

U slijedećem poglavlju će biti prikazana izrada i struktura baze podataka i njenih tablica, te opisani najvažniji koncepti.

U četvrtom poglavlju će biti nabrojani i opisani alati koji su korišteni pri izradi aplikacije.

U zadnjem poglavlju počinje praktični dio izrade rada tj. glavni i najvažniji dio ovog rada. U praktičnom dijelu će biti opisan kompletan postupak razvoja aplikacije od početka do konačnog izgleda aplikacije. Biti će objašnjene najvažnije funkcionalnosti aplikacije uz popratne primjere koda. Potom slijedi zaključak koji sadrži završnu riječ kao osvrt na kompletni rad.

Motivacija za izradu ove aplikacije proizašla je iz želje da se pojednostavni pristup informacijama o plesnom studiju i da se olakša prijava na tečaj plesa.

2. Tehnologije razvoja aplikacije

U ovom poglavlju biti će opisane tehnologije korištene u razvoju web aplikacije za plesni studio. Na poslužiteljskoj strani (eng. *Backend*) korištene su najpoznatije tehnologije kod izrade web aplikacija, a to su: skriptni jezik PHP i MySQL baza podataka. Prilikom izrade aplikacije korišten je i Symfony razvojni okvir pomoću kojeg je napravljen i dizajn stranice.

2.1. XAMPP

XAMPP je potpuno besplatan serverski paket otvorenog koda koji u sebi sadrži sve potrebno za pokretanje web aplikacije. Za web aplikaciju je potrebna baza korisnika kako bi se mogla napraviti prijava i registracija korisnika. Da bi kreirali bazu podataka potrebno je instalirati XAMPP. Instalacijom XAMPP-a dobivamo: Apache i MySQL poslužitelj, kao i MariaDB bazu i skriptni jezik PHP. XAMPP je namijenjen za upotrebu u lokalnoj mreži, a ne kao web server. [1]

2.2. PHP

„PHP je jednostavan, ali moćan programski jezik kreiran za stvaranje HTML (eng. *HyperTextMarkup Language*) sadržaja.“ [2] PHP je skriptni programski jezik koji se može koristiti na dva osnovna načina [2]:

1. skriptiranje na strani poslužitelja (eng. *server-side scripting*) – PHP je kreiran za kreiranje dinamičkih web-sadržaja te omogućava generiranje HTML-a za što je potreban PHP parser i web-server preko kojeg se šalju kodirani dokumenti
2. skriptiranje naredbenog retka (eng. *command-line scripting*) – PHP može slati skripte i preko naredbenog sučelja kao npr. Perl ili Unix; mogu se koristiti skripte naredbenog retka za kreiranje sigurnosnih kopija (eng. *backup*) ili za parsiranje logova.

Prema Tatroe i MacIntyre [2], Rasmus Lerdorf je 1994. godine izumio PHP. Do danas je izdana verzija 8.1 a prema istraživanjima i dalje se najviše koristi verzija 5.0. Zbog činjenice da je PHP programski jezik jedan od najpopularnijih i najkorištenijih skriptnih jezika te zbog njegove široke upotrebe, razvijeni su mnogi okviri koji olakšavaju izradu web aplikacija. Razvojni okviri služe kako bi se smanjio broj koraka pri izradi aplikacije, tj. kako bi se preskočilo pisanje čestih i ponavljajućih dijelova koda pri izradi svake aplikacije.

Neki od najpopularnijih PHP razvojnih okvira su Laravel, CodeIgniter, CakePHP i Symfony. Symfony će biti korišten prilikom izrade ove aplikacije.

2.3. MySQL

MySQL (eng. *relational database management system)* je najpoznatiji besplatan sustav otvorenog koda za upravljanje bazom podataka. MySQL je izradila švedska tvrtka MySQL AB 1995. godine, koja je osnovana iste godine. Osnivači su joj Michael Widenius, David Axmark i Allan Larsson. Razlog popularnosti MySQL-a je: cijena, brzina obrade naredbi i lakoća izvođenja u zahtijevanju poslužiteljskih resursa. MySQL je točnije sustav za upravljanje relacijskim bazama podataka koji radi preko SQL-a. [3]

Tvrtka MySQL AB je prodana Sun Microsystems-u za 1 bilijun dolara 2008. godine. Dvije godine kasnije Oracle kupuje većinski paket dionica Sun Microsystems-a te postaje jedna od vodećih svjetskih informatičkih kompanija. Nakon toga dolazi do razvoja novog sustava temeljenog na MySQL-u – MariaDB. Neki od korisnika MySQL-a su: Facebook, Netflix, YouTube, Spotify, Tesla, Twitter, ebay, Pinterest, NASA i mnoge druge poznate kompanije. [4]

2.4. MariaDB

U ovom radu za izradu samih tablica korišten je alat phpMyAdmin za administraciju MariaDB baze preko weba.

Sustav za upravljanje bazama podataka MariaDB kreiran je od istih autora kao što je to i MySQL. Voditelj projekta za razvoj SUBP MariaDB je Michael Widenius koji je osnivač MySQL-a. MariaDB nudi temeljno novi pristup bazi podataka koji odgovara današnjem modernom okruženju. Poticaj za razvoj ovoga sustava došao je zbog prodaje MySQL-a u vlasništvo korporacije Oracle. Autori su željeli stvoriti nov sustav koji će biti otvorenoga koda i dostupan svima na korištenje, a podupirati jednake funkcionalnosti kao i MySQL. Ideja je bila zadržati što veću kompatibilnost s MySQL-om kako bi prijelaz s jednog sustava na drugi bio što jednostavniji. [5]

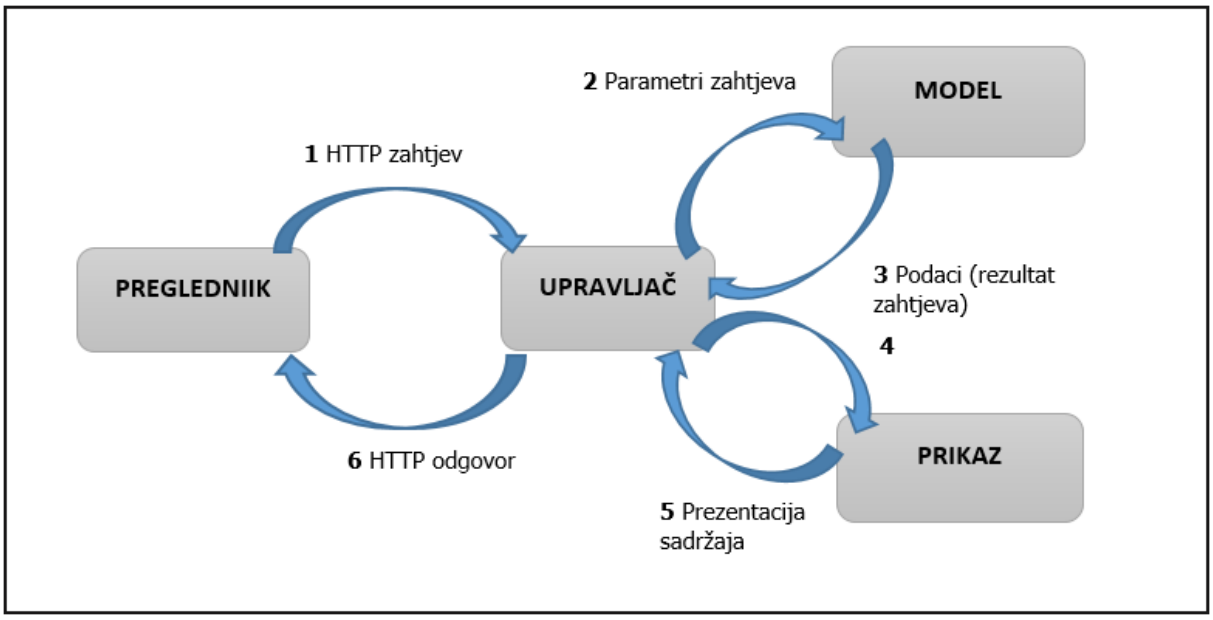
Sustav MariaDB razvijen je na temeljima MySQL- a tako da su naredbe i rad jednaki u oba sustava. Oba podržavaju relacijski model podataka, koriste tablice, ograničenja, okidače, uloge, pohranjene procedure, poglede i ostale karakteristike sustava za upravljanje bazama podataka. Ove zajedničke osobine omogućavaju lak prijelaz s jednoga sustava na drugi bez mnogo truda.

2.5. Symfony

Symfony je jedan od popularnijih PHP razvojnih okvira koji omogućuje izgradnju web stranica ugodnijom i bržom. Kao razvojni okvir nudi sve što bi očekivali od njega, od pružanja potpune kontrole nad konfiguracijom, strukture direktorija, do vanjskih programskih proširenja tj. biblioteka (eng. *plugins/libraries*) uz fokus na ponovnu iskoristivost, brzinu i fleksibilnost [6].

PHP razvojni okviri imaju zajedničke značajke, a ujedno i prednosti [7]:

- Model-prikaz-upravljač (MVC) arhitektura koja osigurava brz razvoj aplikacije
- CRUD (eng. *Create, Read, Update and Delete*) operacije koje omogućavaju preuzimanje podataka iz baze bez pisanja složenih upita
- „ne ponavljaj se“ načelo (eng. *DRY – don't repeat yourself*) kojim je osiguran maksimalan utjecaj minimalne količine koda
- osiguravanje skalabilnosti sustava što omogućava rast i širenje radnog okvira
- povećanje sigurnosti web-aplikacije od različitih sigurnosnih prijetnji.



Slika 1: Grafički prikaz rada MVC (Prema: Smijulj i Meštrović, 2014)

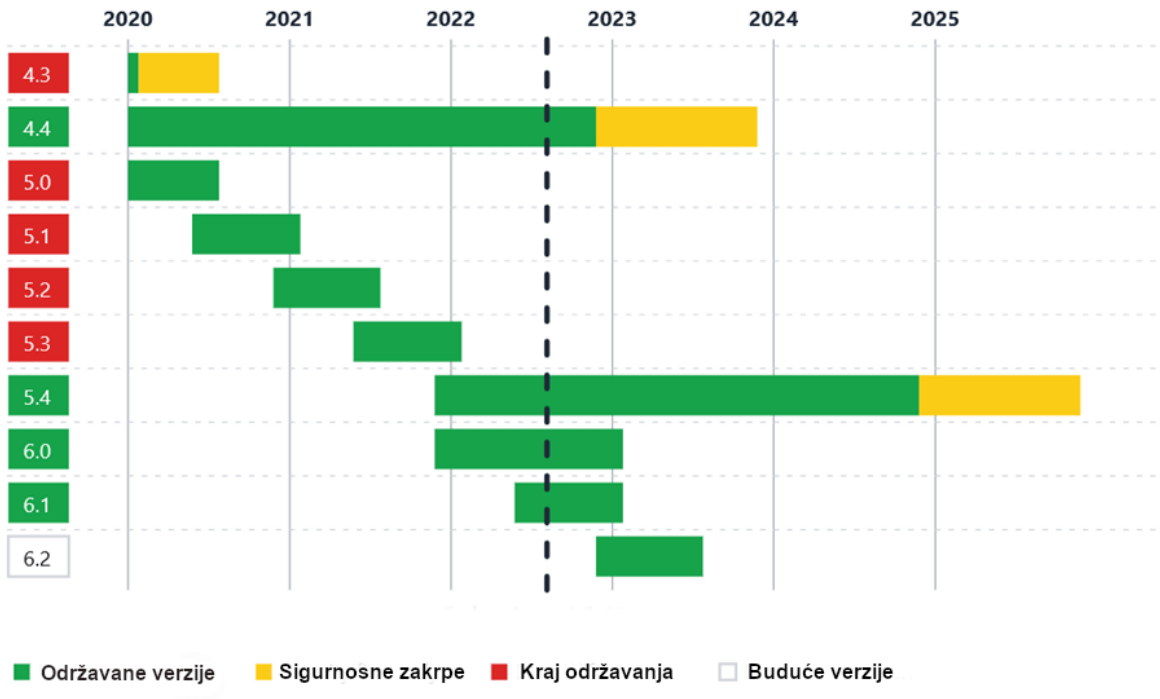
Glavne ideje MVC arhitekture su razdvajanje koda u zasebne cjeline, odnosno dosljedna struktura i mogućnost ponovnog korištenja istog koda. MVC arhitektura se sastoji od triju specifičnih i međusobno zavisnih komponenti: model, prezentacija (eng. *view*) i upravljač (eng. *controller*). [7]

Slika 1 prikazuje proces koji počinje korisnikovim zahtjevom (upisivanje web-adrese u pregledniku) kojeg najprije analizira upravljač (koji sadrži funkcije koje je potrebno izvršiti). Upravljač prosljeđuje modelu i prikazu potrebne parametre, ali i kontrolira (upravlja) daljnjim slijedom procesa. Uloga modela je rad s podacima, komunikacija s bazom podataka, dodatna obrada primljenih podatke te vraćanje rezultata upravljaču. U nekim slučajevima upravljač dobivene podatke šalje drugim procesima te će oni biti naknadno obrađivani ili, najčešće, upravljač završava proces pozivajući prikaz koji onda generira zadani predložak po kojem će se podaci prikazati. Takav predložak (HTML kod) upravljač šalje prema korisniku, tj. pregledniku. [8]

Neke od značajki razvojnog okvira Symfony su:

- Symfony MakerBundle generator koda koji se koristi za kreiranje praznih naredbi, upravljača i testova
- Twig sustav za predloške
- PHPUnit predložak za testiranje aplikacije
- Korištenje postojećih rješenja treće strane i instalacija pomoću Composera
- Opširna službena dokumentacija
- Velika zajednica
- Profesionalan i težak za svladavanje

Symfony je stvoren za velike aplikacije i kao takav se koristi od samih početaka. Najbolji primjeri koji koriste Symfony su velike platforme poput Drupala i Magenta. Zadnja verzija je 6.1.3. [9]

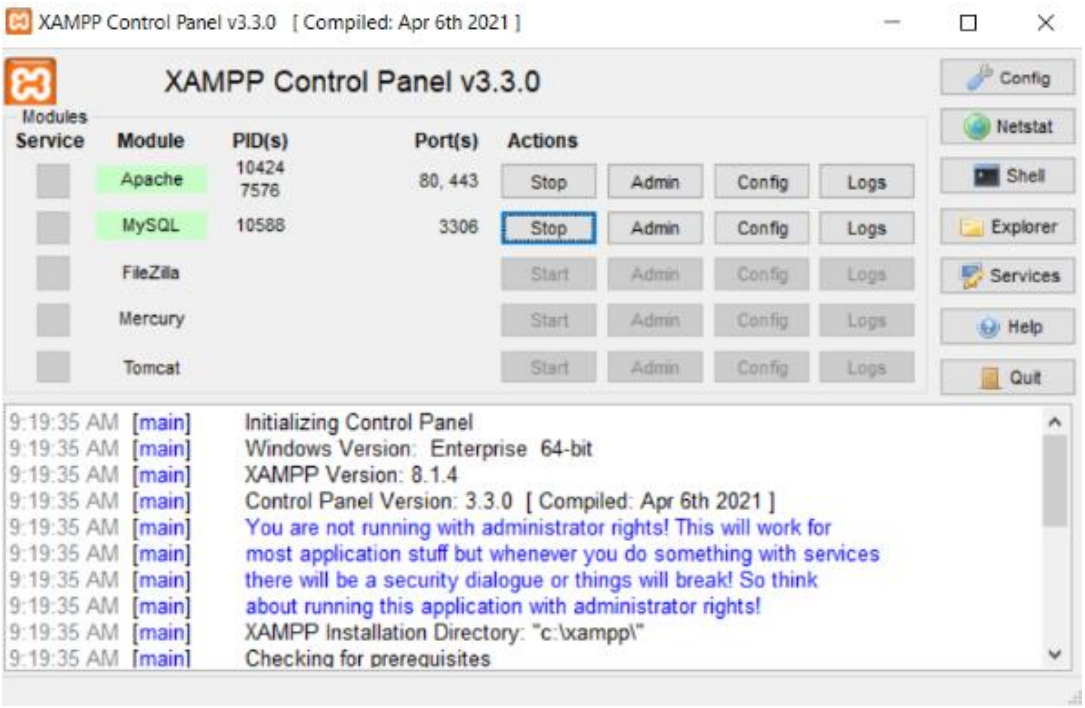


Slika 2: Symfony verzije (Izvor: Symfony)

Na slici 2 se mogu vidjeti trenutne održavane verzije Symfony-a i kada izlaze nove buduće verzije, a također se vidi koje se verzije više ne održavaju.

3. Baza podataka

Prije početka izrade web aplikacije, potrebno je izraditi bazu podataka kako bi se mogla napraviti prijava i registracija korisnika za web aplikaciju. Za kreiranje baze podataka, korištena je najpopularnija baza među web aplikacijama – MySQL. Da bi se kreirala baza potrebno je preuzeti i instalirati XAMPP alat.

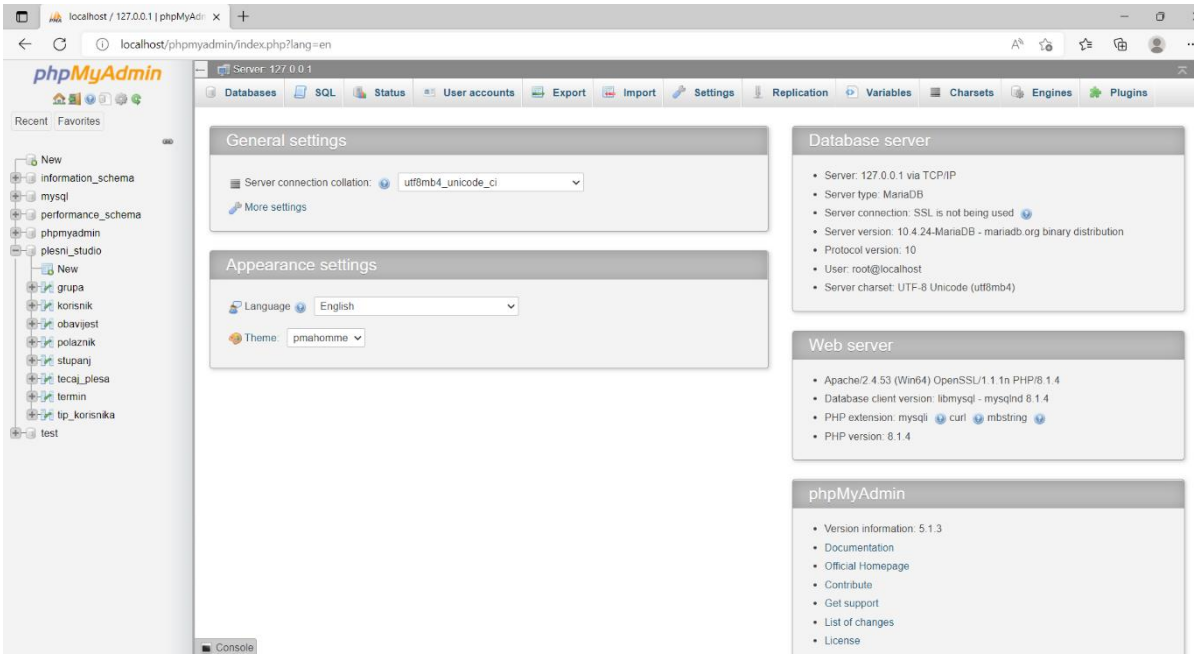


Slika 3: XAMPP

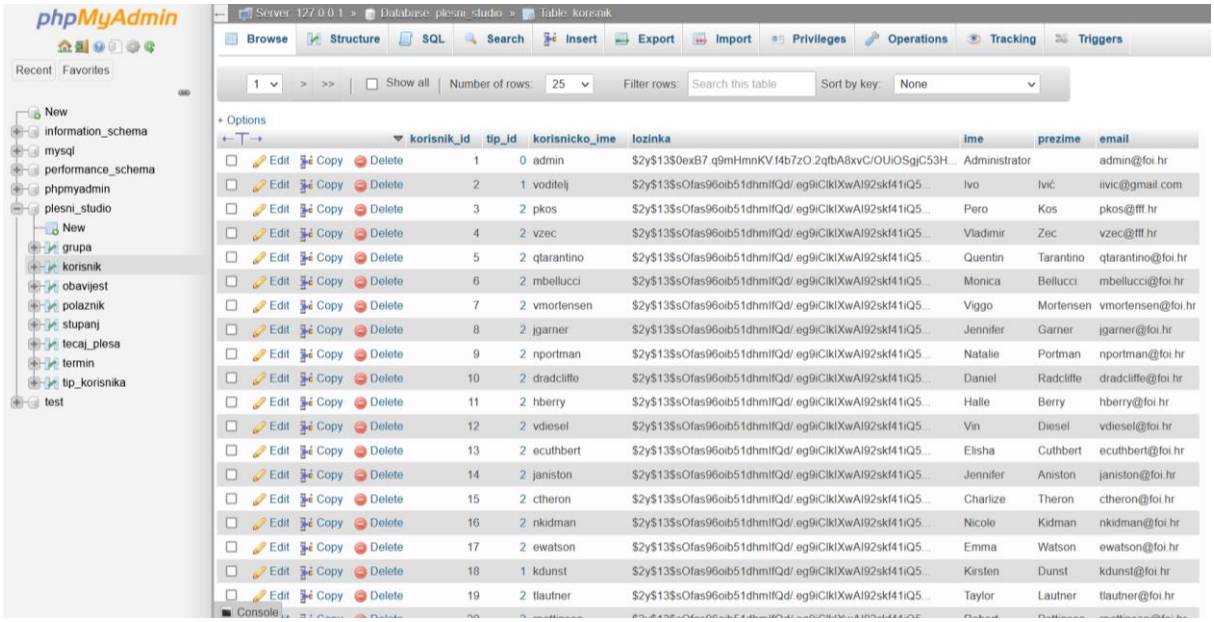
Instalacijom XAMPP-a instalira se i PHP samo treba postaviti environment varijablu-PATH da gleda na php folder u xampp folderu. Kako bi se mogao vidjeti lokalni server mora se uključiti Apache i MySQL u XAMPP alatu. Apache mora biti uključen jer je PHP serverski (server-side) skriptni jezik koji zahtjeva server za rad.

Nakon toga da bi se kreirala baza i njene tablice potrebno je pristupiti phpMyAdmin-u na taj način da se u pretraživač unese adresa http://localhost/phpmyadmin/. PhpMyAdmin je besplatan alat napisan u PHP-u koji služi za upravljanje MySQL bazama podataka. Pomoću njega se mogu kreirati baze, vidjeti sve dosad kreirane baze, njihovu strukturu, mogu se čitati, dodavati, brisati i mijenjati podaci u bazi.

Na desnoj strani slike 4 nalaze se osnovni podaci iz kojih je vidljivo da se koristi server MariaDB verzije 10.4.24, te da je verzija PHP-a 8.1.4. S lijeve strane se nalazi popis baza i njihovih tablica koje su kreirane za potrebu izrade web aplikacije za plesni studio.



Slika 4: phpMyAdmin

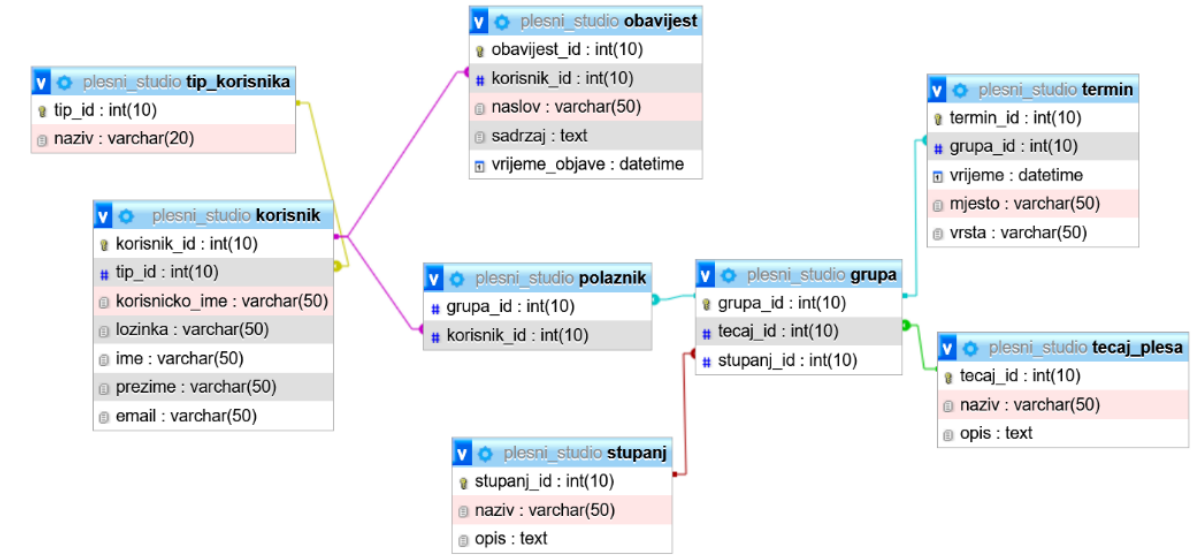


Slika 5: Tablica korisnik

Na slici 5 se vidi da kad se odabere neka tablica npr. korisnik vide se svi podaci vezani za korisnike. Svakog korisnika se može obrisati, urediti ili dodati novog.

3.1. Era dijagram

Shema baze podataka je prikazana pomoću Era dijagrama (eng. *entity relationship model*). Uvidom u Era dijagram može se vidjeti struktura baze podataka, relacije i atributi te veze između njih.



Slika 6: Era dijagram

Iz prikazanog dijagrama vidi se da baza podataka ima 8 tablica koje su međusobno povezane. Tablice imaju primarni ključ id koji se automatski generira na bazi. Tablica korisnik je povezana sa tablicom tip_korisnika preko vanjskog ključa tip_id. Tablica obavijest je povezana sa tablicom korisnik preko vanjskog ključa korisnik_id. Na tablicu grupa se vežu tablice stupanj i tecaj_plesa preko svojih ključeva tecaj_id i stupanj_id. Tablica termin je povezana s tablicom grupa preko vanjskog ključa grupa_id. Tablicu polaznik čine vanjski ključevi tablice korisnik i tablice grupa.

4. Instalacija alata

Da bi se kreirala prva stranica aplikacije, koristit će se ovi alati koje je potrebno prvo instalirati: Visual Studio Code, PowerShell, Composer, Git, Scoop i na kraju sam razvojni okvir za izradu aplikacije Symfony. U nastavku će biti objašnjeni korišteni alati.

4.1. Visual Studio Code

Visual Studio Code je korišten za izradu projekta. VS Code predstavlja integrirano razvojno okruženje (eng. *Integrated Development Environment* – *IDE*) koje korisnicima nudi mogućnost uređivanja različitih vrsta teksta i koda. Jedna od prednosti je što omogućava besplatnu instalaciju dodataka. IntelliSense je ugrađena podrška za dovršavanje koda, koje koristi bogato razumijevanje semantičkog koda, navigaciju i preradu koda. Uza sve to sadrži i podršku za Git (alat za praćenje izmjena u kodu). [10]

Potrebno ga je preuzeti sa stranice https://code.visualstudio.com/download.

4.2. PowerShell

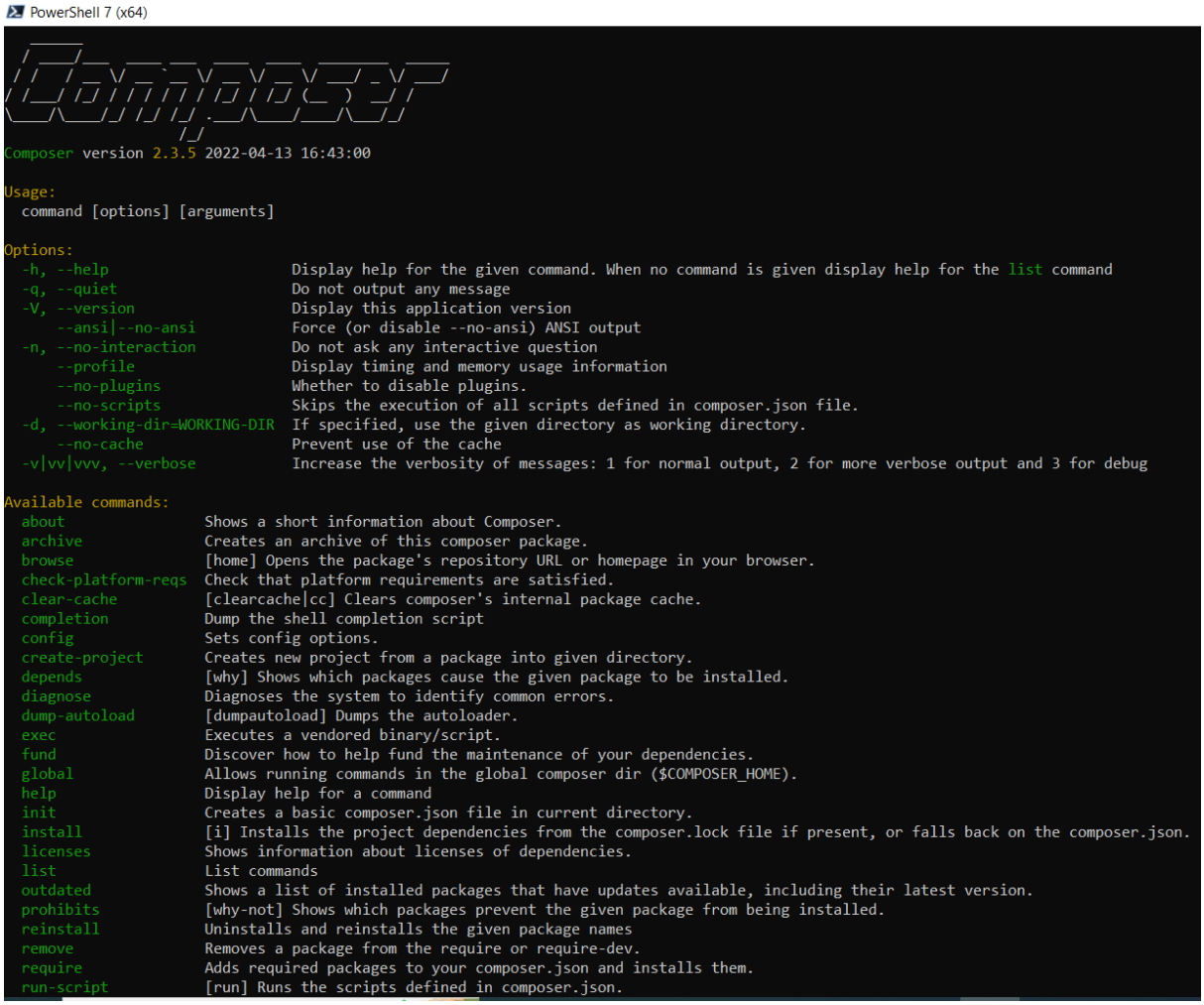
Od prvih verzija Windowsa korisnicima je na raspolaganju osnovni alat za upis i izvođenje naredbi (eng. *Command Prompt*), a od 2006. godine dostupan je napredniji alat pod nazivom PowerShell. PowerShell je prvobitno razvijen od strane Microsoft kompanije, kao alat za automatiziranje radnji, i kao upravljački alat za različite konfiguracije OS-a i aplikacija. Sada je PowerShell otvoreni kod.

PowerShell se može koristiti za izvođenje pojedinačnih naredbi, ali najveće prednosti pokazuje tijekom korištenja različitih vrsti skripti s većim brojem naredbi i parametara namijenjenih automatizaciji različitih zadataka, odnosno za pregled i upravljanje konfiguracijom operativnog sustava. Prilikom pripreme skripti mogu se koristiti različite dodatne mogućnosti alata, kao što su varijable, polja, kontrolne strukture, objektna proširenja i ostalo. Uz pomoć PowerShella možemo pokrenuti već pripremljene skripte, koje izvršavaju jednu ili više naredbi. [11]

4.3. Composer

Za instalaciju i ažuriranje razvojnog okvira Symfony potreban je Composer. Composer je alat za upravljanje ovisnim paketima (eng. *package manager*) u aplikacijama napisanim u PHP programskom jeziku. Pomoću alata Composer moguće je instalirati i ažurirati sve pakete koji su navedeni kao potrebni u aplikaciji koju programer razvija. Da bi Composer radio na operacijskom sustavu potrebno je imati instaliran PHP interpreter (instaliran je unutar Xampp paketa). Neovisno o operacijskom sustavu, Composer se koristi iz naredbenog retka. Composer nudi određeni skup naredbi pomoću kojih možemo obavljati različite radnje na projektu. [12]

Composer je prije instalacije potrebno skinuti sa službene web-stranice https://getcomposer.org/. Nakon preuzimanja programa, potrebno ga je instalirati izvršavajući naredbu ***composer install***. Na slici je vidljiv rezultat instalacije.



Slika 7: Composer

4.4. Git

Git je besplatan alat za praćenje izmjena u kodu. Dizajniran je za upravljanje od najmanjih do najvećih projekata, brzo i efikasno. Prva službena verzija objavljena je 2005. godine, a iznimno brzo je dobio na popularnosti. U usporedbi s drugim sličnim alatima, Git je jednostavan za korištenje, responzivan, dizajniran da dobro radi i s tekstualnim datotekama i najvažnije, omogućava grananje (eng. *branching*) tj. kreiranje nove grane na kojoj se može nezavisno nastaviti raditi od produkcijske verzije, što je odlično za potrebe testiranje, rada na nekoj novo stvari, a uvijek se jednostavno prebaciti na onu granu gdje si stao i želiš nastaviti i povezati (eng. *merge*) trenutni rad s njom. [13]

Potrebno ga je preuzeti sa stranice https://git-scm.com/downloads, pa instalirati.

4.5. Scoop

Scoop služi za instalaciju programa pomoću naredbenog retka. Ima mogućnost eliminiranja skočnih prozora, izbjegava neočekivane nuspojave instaliranja i deinstaliranja programa, sprječava da se PATH ne preoptereti instalacijom puno programa. Da bi instalirali Scoop potrebno je pokrenuti PowerShell i pokrenuti naredbu:

```
iwr -useb get.scoop.sh | iex
```

Potom se pokreće druga naredba kojom će se instalirati Symfony:

```
scoop install symfony-cli
```

Nakon instalacije Symfony-a kreće se s kreiranjem aplikacije.

5. Symfony aplikacija

Izrada aplikacije prati Symfony dokumentaciju koja se nalazi na stranicama Symfony-a: https://symfony.com/doc/current/index.html. Dokumentacija znatno olakšava izradu aplikacije pogotovo početnicima. Za kreiranje aplikacije u Symfony-u potrebno je pokrenuti PowerShell i zatim pokrenuti naredbu:

```
symfony new --webapp plesni_studio
```

Nakon što je aplikacija spremna potrebno je u terminalu pokrenuti lokalni server kako bi se moglo pristupiti aplikaciji, prvo je potrebno pristupiti mapi u kojoj je spremljen projekt te nakon toga pokrenuti server. Pokreće se s naredbom:

```
PS C:\Users\Acer> cd C:\symfony\plesni_studio
PS C:\symfony\plesni_studio> symfony server:start
```

Slika 8: Naredba za pokretanje servera

Kad je server pokrenut, aplikaciji se može pristupiti tako što će se u pretraživač upisati adresa http://localhost:8000/. Kroz daljnji rad pristupati će se aplikaciji preko te adrese. Da bi se mogla pokrenuti početna stranica prvo je potrebno postaviti rutu. U terminalu se prvo pokreće naredba za instalaciju potrebnog paketa:

```
PS C:\symfony\plesni_studio> composer require annotations
```

Slika 9: Naredba za instalaciju paketa

Sada se može dodati ruta direktno u kod na taj način da će se prvo pristupiti u Visual Studio Code u kojem će se pisati kod za aplikaciju. Nakon što je dodana ruta početne stranice može joj se pristupiti. Za svaku stranicu će biti potrebno na isti način postaviti rutu. U nastavku je prikazan kod dokumenta ONamaDashboardController.php i postavljena ruta.

```
<?php

namespace App\Controller;

use EasyCorp\Bundle\EasyAdminBundle\Router\AdminUrlGenerator;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class ONamaDashboardController extends PlesniStudioDashboardController
{
    /**
     * @Route("/", name="o_nama")
```

```
     */
    public function index(): Response
    {
        $adminUrlGenerator = $this->container-
>get(AdminUrlGenerator::class);
        return $this->redirect($adminUrlGenerator-
>setController(ONamaController::class)->generateUrl());
    }
}
```

5.1. Webpack Encore

Predlošci su najbolji način za organizaciju HTML-a, a sustav za pisanje predložaka koji koristi Symfony zove se Twig. Twig omogućuje pisanje sažetih, čitljivih predložaka koji su jednostavniji web dizajnerima i bolji od predložaka PHP-a. Instalira se ovom naredbom:

```
PS C:\symfony\plesni_studio> composer require twig
```

Slika 10: Naredba za instalaciju twig-a

Potom je potreban Webpack Encore. Webpack je alat koji uzima sve module aplikacije s proširenjima (eng. *dependencies*), slikama i slično koji su na neki način povezani i "zapakira" ih u standardni JavaScript, CSS te neki od formata za slike. Encore je dodatak orijentiran za korištenje kod Symfony aplikacija (može se koristiti i samostalno) koji će uzeti sve te module s proširenjima i od njih napraviti grupaciju modula. [14]

Prije instalacije Encore-a potrebno je instalirati node.js. Node.js je drugačija inačica JavaScripta, upravo zato jer Node.js dodaje nove mogućnosti Javascript-u koje nisu moguće unutar Web preglednika, primjerice, pristupanje i modificiranje lokalnih datoteka. Node.js ima i mogućnost proširivanja pomoću paketa. Paketi su dodaci Node.js okviru u obliku JavaScript biblioteka koje se mogu uključiti u izvedbu bilo koje skripte. Ti paketi se mogu dodavati u Node.js okruženje putem Node Package Manager-a (npm).

Kad je Node.js instaliran, Encore će se instalirati pomoću slijedeće naredbe:

```
PS C:\symfony\plesni_studio> npm install @symfony/webpack-encore --save-dev
```

Slika 11: Naredba za instalaciju Encore-a

5.2. Bootstrap

Bootstrap je radni okvir (eng. *framework*) koji služi za dizajn i izgled stranica. Da bi se instalirao i integrirao Bootstrap u Symfony aplikaciju prvo se mora pokrenuti naredba:

```
PS C:\symfony\plesni_studio> npm install bootstrap --save-dev
```

Slika 12: Naredba za instalaciju Bootstrap-a

Potrebno je još i instalirati jQuery:

```
PS C:\symfony\plesni_studio> npm install jquery @popperjs/core --save-dev
```

Slika 13: Naredba za instalaciju jQuery-a

JQuery je ugrađena knjižnica (eng. *library*) pisanih CSS i JavaScript alata, skripti i funkcija koje omogućavaju stvaranje brze, dinamične i lijepe web aplikacije koja se prikazuje jednako na svim pretraživačima, neovisno o rezoluciji monitora i na mobitelima.

5.3. Baza podataka i Doctrine ORM

Symfony omogućuje sve moguće alate za rad s bazama podataka zahvaljujući Doctrine-u. Doctrine je set PHP biblioteka primarno fokusiranih na pružanju apstrakcije s radom na bazi podataka. Razina apstrakcije proizlazi kroz objektno relacijsko mapiranje (eng. ORM object relation mapper). Prvo je potrebno instalirati podršku za Doctrine preko orm Symfony paketa, a zatim MakerBundle koji pomaže pri generiranju koda:

```
PS C:\symfony\plesni_studio> composer require symfony/orm-pack
```

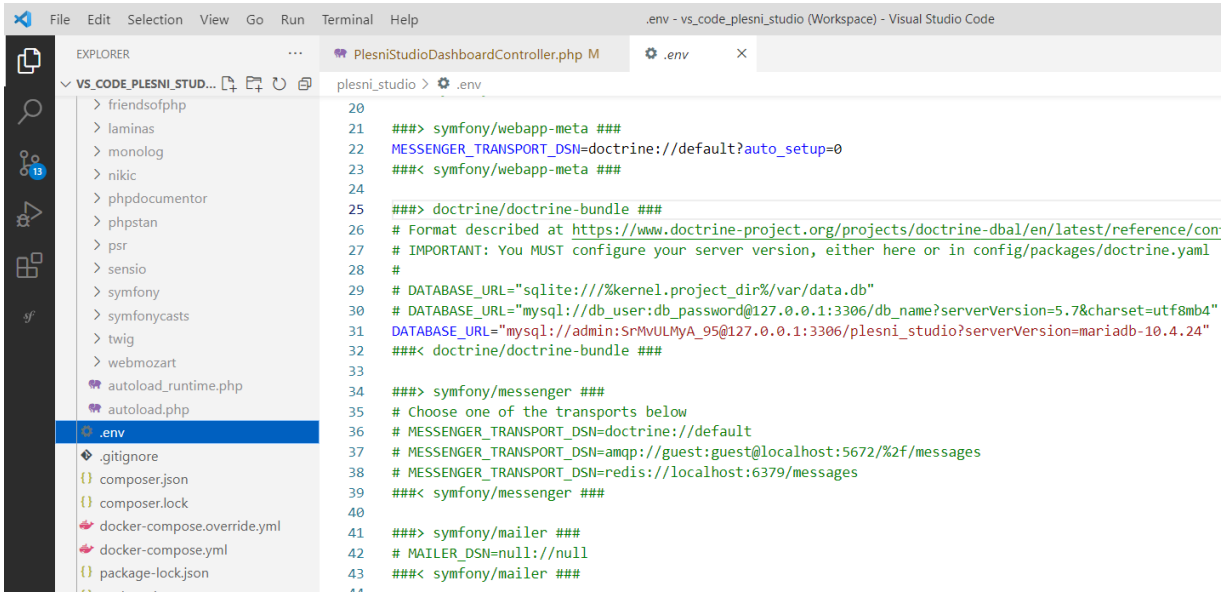
Slika 14: Naredba za instalaciju orm paketa

```
PS C:\symfony\plesni_studio> composer require --dev symfony/maker-bundle
```

Slika 15: Naredba za instalaciju MakerBundle-a

5.3.1. Konfiguracija baze podataka

Informacije o spajanju baze podataka pohranjene su kao DATABASE_URL varijabla okruženja(eng. *environment variable*). Parametri potrebni za konfiguraciju baze podataka se nalaze u .env datoteci. Prvo se moraju unesti podaci koji su potrebni za spajanje baze.



Slika 16: Konfiguracija baze podataka

Sa slike 16 je vidljivo kako je korištena mysql baza podataka koja sadrži ove podatke:

korisničko ime (db_user): admin

lozinka (db_password): SrMvULMyA_95

naziv baze (db_name): plesni_studio

Nakon što se projekt povezao s bazom podataka, trebalo bi biti moguće izraditi entitete automatski.

5.3.2. Entiteti

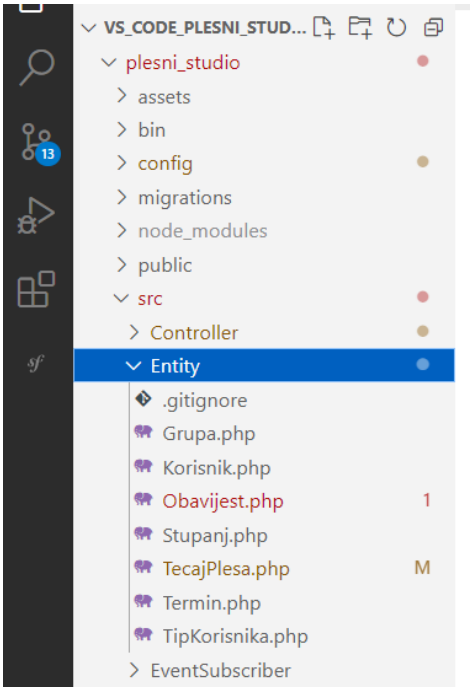
Entiteti su PHP objekti identificirani s unikatnom vrijednošću ili primarnim ključem. Klase koje ne trebaju nasljeđivati neku apstraktnu ili baznu klasu. Veza s bazom proizlazi kroz EntityManager koji je Doctrin-ovo javno sučelje za upravljanje radom baze podataka i poslovne logike aplikacije. On je zadužen za osnovne CRUD operacije, transformaciju podatka s entiteta u tabličnu strukturu.

Naredna naredba koristit će se kako bi se generirali entitete (eng. *entities*) iz tablica:

```
php bin/console doctrine:mapping:import "App\Entity" annotation --path=src/Entity
```

Slika 17: Naredba za generiranje entiteta

Nakon što su izgenerirani entiteti za sve tablice, za svaki entitet stvoren je dokument koji je pohranjen u src/Entity. Na slici 18 se vide datoteke koje su nastale:



Slika 18: Entiteti

S obzirom da su dobiveni samo entiteti, kod nije potpun jer nedostaju metode get i set. Potrebno je pokrenuti iduću naredbu:

```
PS C:\symfony\plesni_studio> php bin/console make:entity --regenerate App
```

Slika 19: Naredba za nadopunu

U nastavku se vidi kako izgleda potpun kod entiteta stupanj u primjeru dokumenta Stupanj.php:

```
<?php

namespace App\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
```

```
 * Stupanj
 *
 * @ORM\Table(name="stupanj")
 * @ORM\Entity
 */
class Stupanj
{
    /**
     * @var int
     *
     * @ORM\Column(name="stupanj_id", type="integer", nullable=false)
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="IDENTITY")
     */
    private $stupanjId;

    /**
     * @var string
     *
     * @ORM\Column(name="naziv", type="string", length=50,
nullable=false)
     */
    private $naziv;

    /**
     * @var string
     *
     * @ORM\Column(name="opis", type="text", length=65535,
nullable=false)
     */
    private $opis;

    public function getStupanjId(): ?int
    {
        return $this->stupanjId;
    }

    public function getNaziv(): ?string
    {
        return $this->naziv;
    }

    public function setNaziv(string $naziv): self
    {
        $this->naziv = $naziv;

        return $this;
    }

    public function getOpis(): ?string
    {
        return $this->opis;
    }

    public function setOpis(string $opis): self
    {
        $this->opis = $opis;

        return $this;
    }
```

```
    public function __toString(): string
    {
        return $this->getNaziv();
    }
}
```

Da bi se mogli dohvaćati i spremati entiteti potreban je repozitorij (eng. *repository*). U dijelu koda u nastavku se vidi da je u 11. liniji unutar komentara `@ORM\Entity ()` specificiran repositoryClass s nazivom repozitorija koji se treba kreirati. Za svaki entitet potrebno je napraviti isto.

```
<?php

namespace App\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * Obavijest
 *
 * @ORM\Table(name="obavijest", indexes={@ORM\Index(name="korisnik_id",
columns={"korisnik_id"})})
 * @ORM\Entity(repositoryClass="App\Repository\ObavijestRepository")
 */
```

Repozitorij će se kreirati kada se pokrene naredba koja se već koristila:

```
PS C:\symfony\plesni_studio> php bin/console make:entity --regenerate App
```

Slika 20: Naredba

Za svaki entitet generiran je repozitorij, a u nastavku se može vidjeti kod dokumenta ObavijestRepository.php:

```
<?php

namespace App\Repository;

use App\Entity\Obavijest;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\ORM\Exception\ORMException;
use Doctrine\ORM\OptimisticLockException;
use Doctrine\Persistence\ManagerRegistry;

/**
 * @extends ServiceEntityRepository<Obavijest>
 *
 * @method Obavijest|null find($id, $lockMode = null, $lockVersion =
null)
 * @method Obavijest|null findOneBy(array $criteria, array $orderBy =
null)
 * @method Obavijest[]    findAll()
 * @method Obavijest[]    findBy(array $criteria, array $orderBy = null,
$limit = null, $offset = null)
```

```
 */
class ObavijestRepository extends ServiceEntityRepository
{
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, Obavijest::class);
    }

    /**
     * @throws ORMException
     * @throws OptimisticLockException
     */
    public function add(Obavijest $entity, bool $flush = false): void
    {
        $this->_em->persist($entity);
        if ($flush) {
            $this->_em->flush();
        }
    }

    /**
     * @throws ORMException
     * @throws OptimisticLockException
     */
    public function remove(Obavijest $entity, bool $flush = false): void
    {
        $this->_em->remove($entity);
        if ($flush) {
            $this->_em->flush();
        }
    }
```

5.4. EasyAdmin i CRUD upravljači

Za kreiranje administrativnog sučelja korišten je EasyAdmin. EasyAdmin je Symfony paket koji pomaže kod kreiranja administracijskog sučelja bez potrebe za pisanja koda, odnosno sve se može kreirati kroz samu konfiguraciju. Podržava automatsko generiranje CRUD (*eng. create, read, update, delete*) operacija, translacije, ima punu podršku s radom na bazi kroz ORM. Da bi se instalirao koristi se naredba:

```
PS C:\symfony\plesni_studio> composer require easycorp/easyadmin-bundle
```

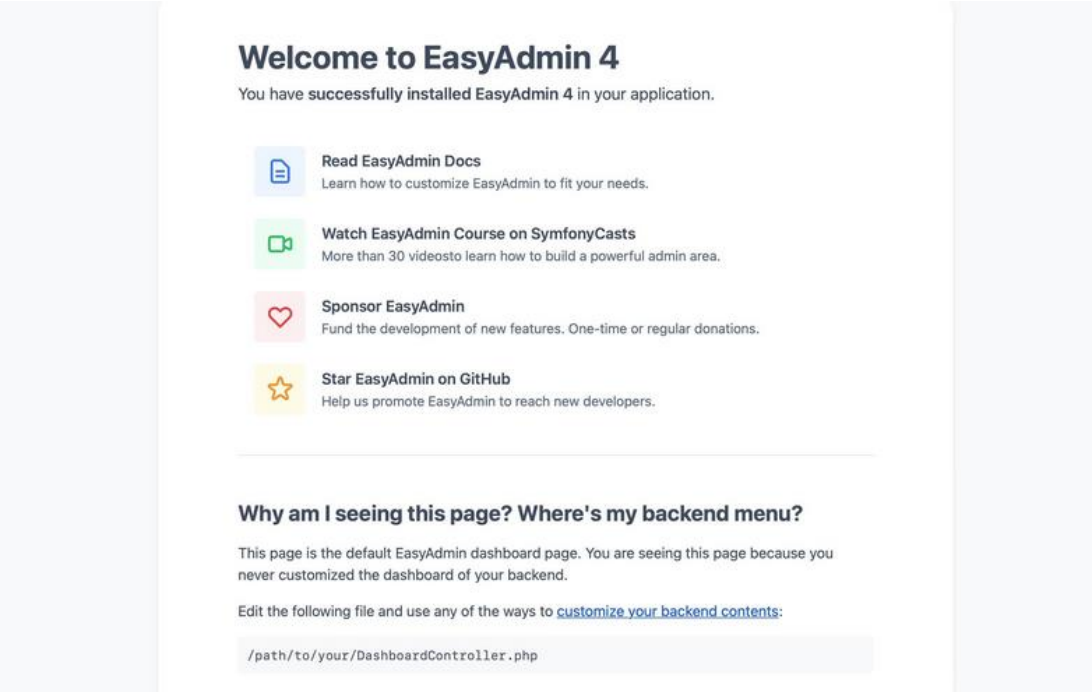
Slika 21: Naredba za instalaciju EasyAdmin-a

Nakon instalacije EasyAdmina potrebno je napraviti ulaznu točku, tj. naslovnicu (eng. *dashboards)*. Za to će se pokrenuti naredba:

```
PS C:\symfony\plesni_studio> php bin/console make:admin:dashboard
```

Slika 22: Naredba za kreiranje naslovnice

Da bi se vidio izgled naslovnice potrebno je u preglednik unesti adresu localhost:8000/admin. Rezultat se vidi na slici 23:



Slika 23: Izgled naslovnice

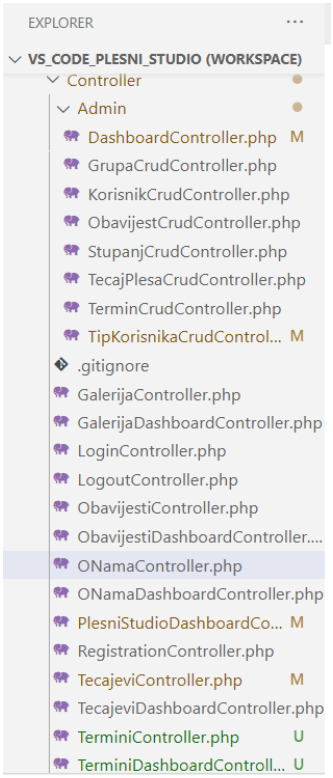
Zatim su potrebni CRUD upravljači. CRUD upravljači pružaju CRUD operacije Doctrine ORM entitetima. Oni će se instalirati pomoću naredbe:

```
PS C:\symfony\plesni_studio> php bin/console make:admin:crud
```

Slika 24: Naredba za kreiranje upravljača

CRUD generator će automatski registrirati rutu s imenom entiteta koji je korišten tj. upravljač koji će automatski upravljati osnovama CRUD-a (prikaz, uređivanje, brisanje).

Na slici 25 mogu se vidjeti svi upravljači koji su nastali generiranjem. Admin sadrži upravljače za naslovnicu i svaki entitet. Zatim se tu nalaze i upravljači za svaku stranicu aplikacije.



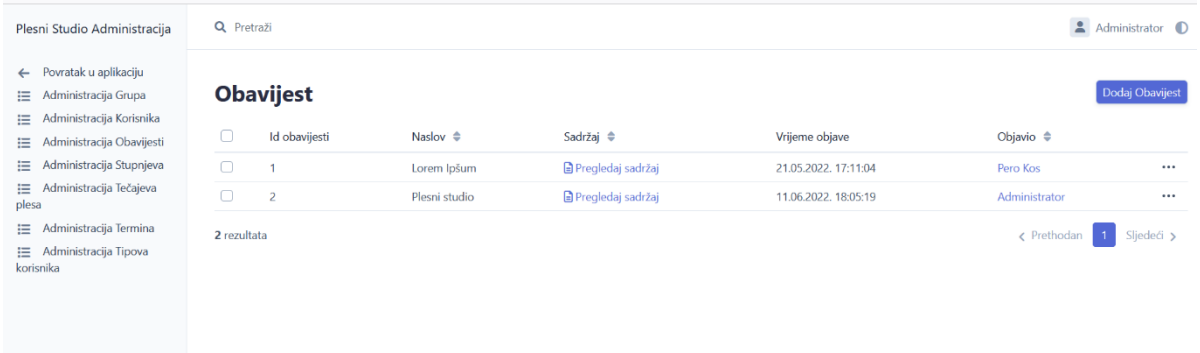
Slika 25: Upravljači

U nastavku se vidi kod upravljača na primjeru dokumenta ObavijestCrudController.php:

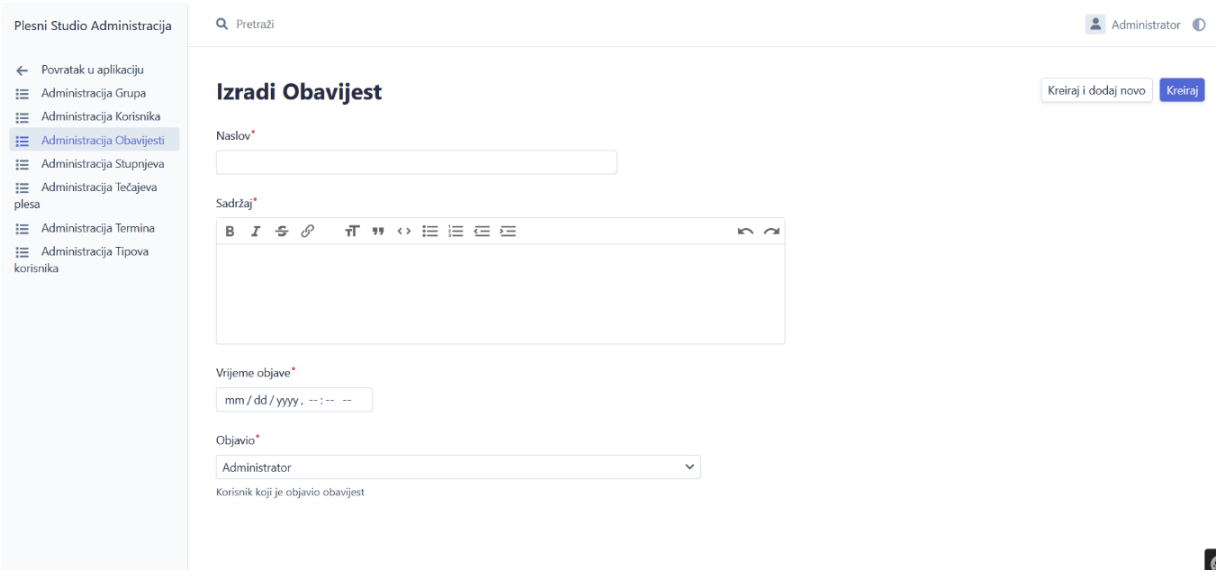
```
<?php

namespace App\Controller\Admin;

use App\Entity\Obavijest;
use EasyCorp\Bundle\EasyAdminBundle\Controller\AbstractCrudController;
use EasyCorp\Bundle\EasyAdminBundle\Field\AssociationField;
use EasyCorp\Bundle\EasyAdminBundle\Field\DateTimeField;
use EasyCorp\Bundle\EasyAdminBundle\Field\IdField;
use EasyCorp\Bundle\EasyAdminBundle\Field\TextEditorField;
use EasyCorp\Bundle\EasyAdminBundle\Field\TextField;

class ObavijestCrudController extends AbstractCrudController
{
    public static function getEntityFqcn(): string
    {
        return Obavijest::class;
    }

    public function configureFields(string $pageName): iterable
    {
        return [
            IdField::new('obavijest_id', 'Id obavijesti')
                ->onlyOnIndex(),
            TextField::new('naslov', 'Naslov')
                ->setRequired(true),
```

```
                TextEditorField::new('sadrzaj', 'Sadržaj')
                    ->setRequired(true),
                DateTimeField::new('vrijeme_objave', 'Vrijeme objave')
                    ->setRequired(true)
                    ->setFormat('dd.MM.Y. HH:mm:ss'),
                AssociationField::new('korisnik', 'Objavio')
                    ->setRequired(true)
                    ->setHelp('Korisnik koji je objavio obavijest'),
            ];
        }
    }
```

Na slici 26 vidi se izgled stranice za administraciju obavijesti. S lijeve strane je navigacija na kojoj se vidi koje se sve stavke mogu administrirati. U desnom kutu se nalazi dugme za dodavanje nove obavijesti. Na slici 27 vidi se obrazac za kreiranje nove obavijesti koji se otvori nakon pritiska dugmeta „Dodaj obavijest“.



Slika 26: Administracija Obavijesti



Slika 27: Obrazac za obavijest

5.5. Polja

Polja (eng. *fields*) omogućavaju prikaz sadržaja iz entiteta za svaku CRUD stranicu. S obzirom da se CRUD upravljači ove aplikacije nadovezuju (eng. *extends*) na AbstractCrudController (vidi se u primjeru koda), EasyAdmin automatski konfigurira polja. Pomoću metode configureFields određujemo što će se prikazati na stranici. Primjer dijela koda dokumenta KorisnikCrudController.php slijedi u nastavku.

```
class KorisnikCrudController extends AbstractCrudController
{
    public static function getEntityFqcn(): string
    {
        return Korisnik::class;
    }

    public function configureFields(string $pageName): iterable
    {
        return [
            IdField::new('korisnik_id', 'Id korisnika')
                ->onlyOnIndex(),
            TextField::new('korisnicko_ime','Korisničko ime')
                ->setRequired(true),
            TextField::new('ime', 'Ime')
                ->setRequired(true),
            TextField::new('prezime', 'Prezime')
                ->setRequired(true),
            TextField::new('lozinka', 'Lozinka')
                ->setRequired(true)
                ->onlyOnForms()
                ->setFormType(PasswordType::class),
            TextField::new('email', 'E-mail')
```

```
                ->setRequired(true),
            AssociationField::new('tip', 'Tip korisnika')
                ->setRequired(true),
            AssociationField::new('grupa', 'Grupe')
                ->setHelp('Grupe kojima pripada korisnik'),
        ];
    }
}
```

5.6. Sigurnost

Symfony omogućava mnogo alata za sigurnost aplikacije. SecurityBundle je paket koji sadrži sve potrebno za autentifikaciju i autorizaciju kako bi se zaštitila aplikacija. Za instalaciju se pokreće slijedeća naredba:

```
PS C:\symfony\plesni_studio> composer require symfony/security-bundle
```

Slika 28: Instalacija SecurityBundle-a

Kad se korisnik prijavi Symfony poziva getRoles () metodu da bi mogao prepoznati koju ulogu (eng. role) korisnik ima. Zbog toga se trebaju formirati uloge. Uloge su dodane u kod dokumenta korisnik.php.

```
public function getRoles(): array
    {
        if ($this->getTip()->getTipId() === TipKorisnika::TIP_ID_ADMIN)
{
            return ['ROLE_ADMIN', 'ROLE_VODITELJ','ROLE_USER'];
        }
        if ($this->getTip()->getTipId() ===
TipKorisnika::TIP_ID_VODITELJ) {
            return ['ROLE_VODITELJ', 'ROLE_USER'];
        }
        return ['ROLE_USER'];
    }
```

Na primjeru koda se vidi da postoje 3 uloge, admin, voditelj i korisnik. Admin ima pristup administraciji. Admin ima ovlasti nad svime, dok voditelj nema. Voditelj može uređivati i brisati obavijesti te dodati nove obavijesti. Korisnik se može prijaviti i vidjeti obavijesti i informacije za plesni studio koje neregistrirani korisnik ne može. Neregistrirani korisnik se može registrirati. Nakon što su dodane uloge potrebno je pokrenuti naredbu za instalaciju obrasca za prijavu (eng. *form login*):

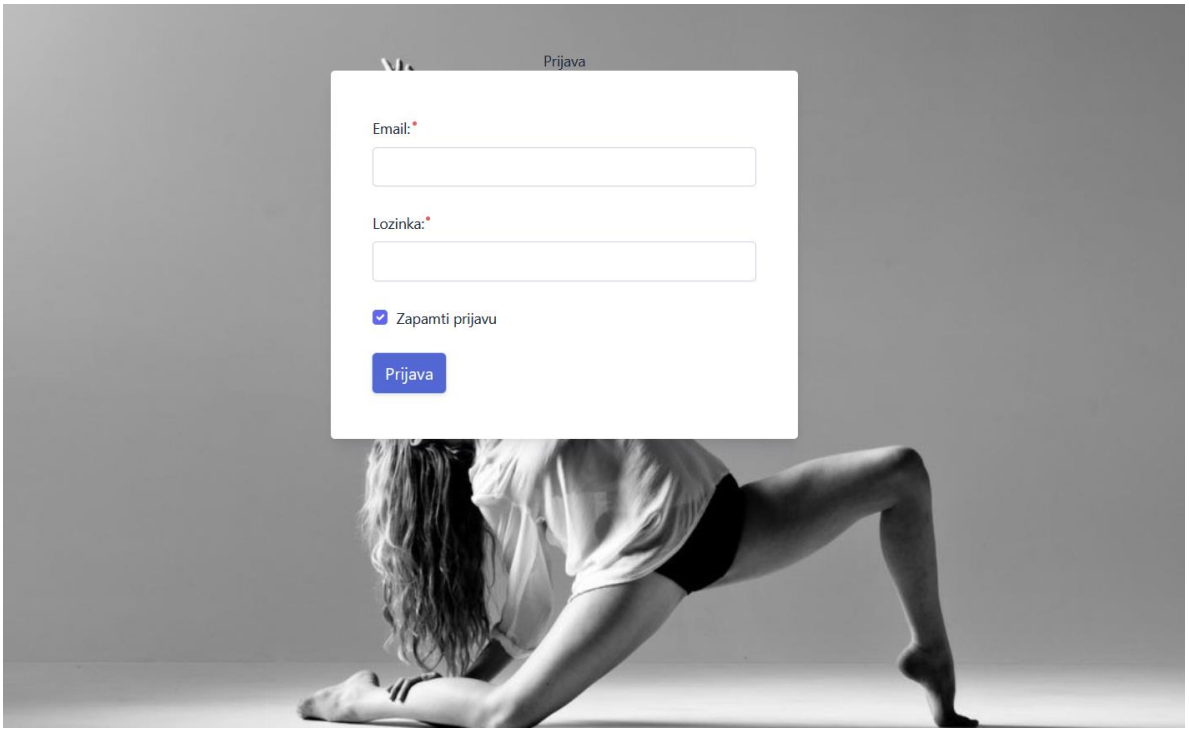
```
PS C:\symfony\plesni_studio> php bin/console make:controller Login
```

Slika 29: Instalacija obrasca za prijavu

Nakon instalacije dobije se LoginController.php. U nastavku se može vidjeti dio koda:

```
 public function login(AuthenticationUtils $authenticationUtils):
Response
    {
        $error = $authenticationUtils->getLastAuthenticationError();
        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render('login/login.html.twig', [
            // parameters usually defined in Symfony login forms
            'error' => $error,
            'last_username' => $lastUsername,
            'translation_domain' => 'admin',
            'favicon_path' => '/favicon-admin.svg',
            'page_title' => 'Prijava',
            'csrf_token_intention' => 'authenticate',
            'target_path' => $this->generateUrl('obavijesti'),
            'username_label' => 'Email:',
            'password_label' => 'Lozinka:',
            'sign_in_label' => 'Prijava',
            'username_parameter' => '_username',
            'password_parameter' => '_password',
            'forgot_password_enabled' => false,
            'forgot_password_path' => $this->generateUrl('o_nama'),

            'forgot_password_label' => 'Zaboravljena lozinka',
            'remember_me_enabled' => true,
            'remember_me_parameter' => 'custom_remember_me_param',
            'remember_me_checked' => true,
            'remember_me_label' => 'Zapamti prijavu',
        ]);
    }
```

Na slici 30 vidi se konačan izgled obrasca za prijavu:



Slika 30: Obrazac za prijavu

Na sličan način se dobije i obrazac za registraciju. Prvo je potrebno pokrenuti naredbu sa slike 31 kako bi se verificirao email:

```
PS C:\symfony\plesni_studio> composer require symfonycasts/verify-email-bundle
```

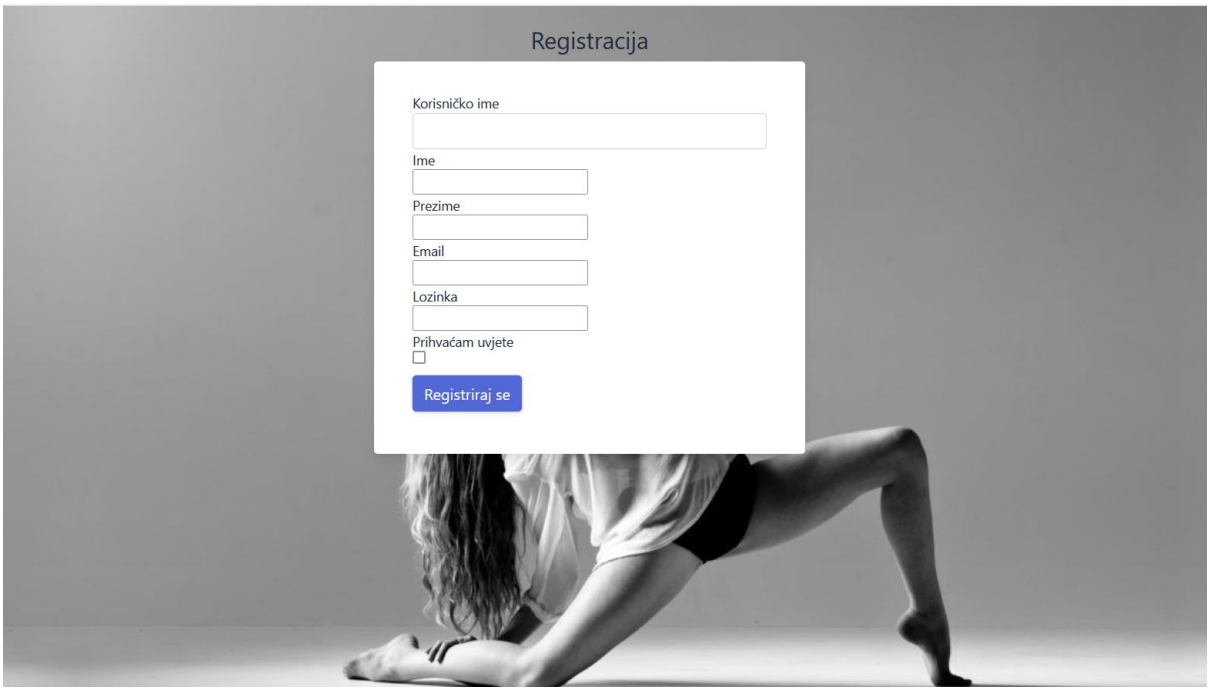
Slika 31: Naredba za verifikaciju email-a

Nakon toga se može pokrenuti naredba za izradu obrasca za registraciju:

```
PS C:\symfony\plesni_studio> php bin/console make:registration-form
```

Slika 32: Naredba za instalaciju obrasca za registraciju

Na slici 33 se može vidjeti finalni izgled obrasca za registraciju:

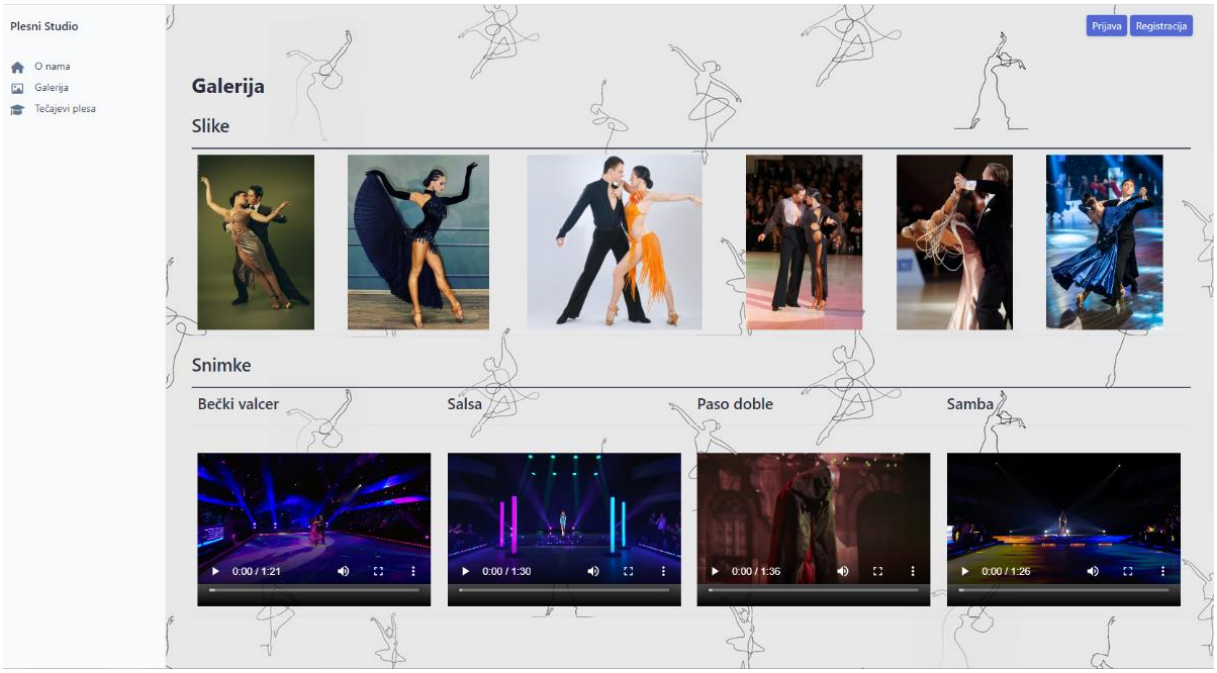


Slika 33: Obrazac za registraciju

5.7. Konačan izgled aplikacije

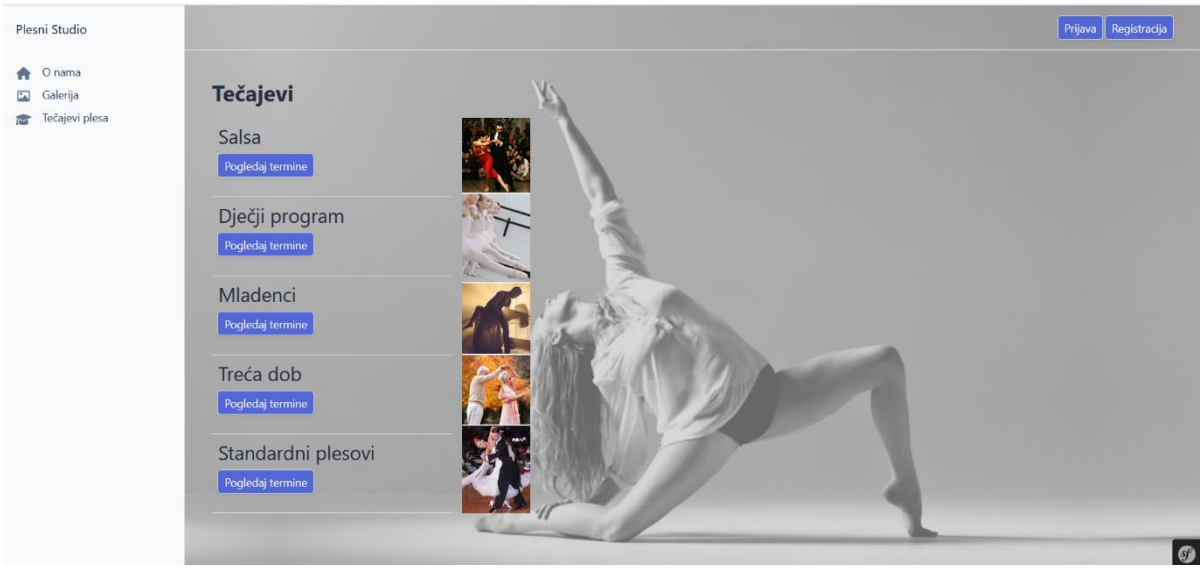
Na naslovnici aplikacije nalaze se kratke informacije o plesnom studiju. S desne strane je navigacija koja je različita ovisno o tipu uloge koja je prijavljena. Neregistrirani korisnik može vidjeti samo naslovnicu, galeriju i koji su tečajevi plesa dostupni. Također u desnom gornjem kutu ima mogućnost prijave ili registracije. Primjer je na slici 34.

Slika 34: Naslovnica



Slika 35: Galerija

Na slici 35 prikazana je galerija koja sadrži slike i video zapise s nastupa.



Slika 36: Tečajevi

Na slici 36 vide se tečajevi koji su dostupni i odabirom tečaja mogu se vidjeti termini treninga i nastupa. Na slici 37 vide se termini.

Slika 37: Termini

Prijavljeni korisnik može vidjeti obavijesti koje su vezane za plesni studio.

Slika 38: Obavijesti

6. Zaključak

Korištenjem različitih tehnologija i pridržavanja standarda pisanja programskog koda izrađena je aplikacija za korisnike koja nudi informacije o plesnom studiju. Aplikacija sadrži naslovnicu, galeriju, stranicu s obavijestima i stranicu s tečajevima i terminima. Aplikacija ima i mogućnost administracije. Administracija sadrži funkcionalnosti kao što su uređivanje podataka o korisnicima i dodavanje novih, upravljanje svim dijelovima aplikacije kao što su administracija tečajeva, termina, grupa i obavijesti. Web aplikacija je prilagođena za sve rezolucije monitora. Također ima napravljenu potpuno funkcionalnu prijavu i registraciju.

Kroz razvoj aplikacije je detaljno prikazan Symfony razvojni okvir. Od početka rada sa Symfony razvojnim okvirom informacije su bile lako dostupne zahvaljujući detaljnoj Symfony dokumentaciji. Symfony olakšava razvoj osnovne aplikacije poput ove i omogućava implementaciju mnogobrojnih funkcionalnosti zbog kojih ne dolazi do bespotrebnog ponavljanja koda.

Izrada web aplikacije za upravljanje plesnim studiom mi je bilo posebno iskustvo i izazov jer sam svoje osnovno znanje HTML-a, CSS-a, PHP-a i MySQL-a ovim završnim radom nadopunila i dovela na jednu veću razinu s obzirom da nisam imala iskustva u radu s razvojnim okvirima. Također su korišteni neki novi alati s kojima se dosad nisam susrela. U prikazu razvoja prikazane su osnovne funkcionalnosti Symfony razvojnog okvira, no aplikacija u budućnosti ima prostora za nadogradnje u kontekstu daljnje prilagodbe dizajna sučelja i proširivanja novim funkcionalnostima.

Popis literature

[1] Apache friends, „What is XAMPP?“ (bez dat.) [Na internetu] Dostupno: https://www.apachefriends.org/ [pristupano 28.4.2022.]

[2] Tatroe, K., & MacIntyre, P. Programming PHP, 4th edition. O'Reilly Media, Inc., 2020.

[3] MySql, „Why MySql“ (bez dat.) [Na internetu] Dostupno: https://www.mysql.com/why-mysql/ [pristupano 30.4.2022.].

[4] MySQL, MySQL Customers (bez dat.) [Na internetu] Dostupno: https://www.mysql.com/customers/ [pristupano 20.5.2022.]

[5] SRCE, „MariaDB baza podataka“ (24.6.2010.) [Na internetu] Dostupno: https://www.srce.unizg.hr/arhiva_weba/sistemac2015/index.php%3Fid=35&no_cache=1&tx_ttnews%255Btt_news%255D=503.html [pristupano 28.5.2022.]

[6] Symfony, „What is Symfony“ (bez dat.) [Na internetu] Dostupno: https://symfony.com/what-is-symfony [pristupano 7.5.2022.]

[7] Laaziri, M., Benmoussa, K., Khoulji, S., & Larbi Kerkeb, M. „A comparative study of PHP frameworks performance“ Procedia Manufacturing, 32, 864-871(2019.) [Na internetu] Dostupno: https://www.sciencedirect.com/science/article/pii/S2351978919303312 [pristupano 30.4.2022.].

[8] Smijulj, A., & Meštrović, A., Izgradnja MVC modularnog radnog okvira. Zbornik Veleučilišta u Rijeci, 2(1), 2014. [Na internetu] Dostupno: https://hrcak.srce.hr/128904 [pristupano 25.5. 2022.]

[9] Symfony, „Symfony Releases“ (bez dat.) [Na internetu] Dostupno: https://symfony.com/releases [pristupano 7.5.2022.]

[10] Visual Studio Code [Na internetu] Dostupno: https://code.visualstudio.com/docs/supporting/FAQ [pristupano 15.6.2022.]

[11]Nenad Crnko, „Powershell-uvod“ (10.09.2019.) Dostupno: https://sistemac.srce.hr/node/162 [pristupano 10.6.2022.]

[12] Composer, „Introduction-Composer“ [Na internetu] Dostupno: https://getcomposer.org/doc/ [pristupano 15.6.2022.]

[13] Git, „About-Git“ [Na internetu] Dostupno: https://git-scm.com/about [pristupano: 20.6.2022.]

[14] Webpack Encore https://symfony.com/doc/current/frontend.html [pristupano: 25.6.2022.]

Popis slika

Slika 1: Grafički prikaz rada MVC (Prema: Smijulj i Meštrović, 2014) 4
Slika 2: Symfony verzije (Izvor: Symfony) 6
Slika 3: XAMPP 7
Slika 4: phpMyAdmin 8
Slika 5: Tablica korisnik 8
Slika 6: Era dijagram 9
Slika 7: Composer 11
Slika 8: Naredba za pokretanje servera 13
Slika 9: Naredba za instalaciju paketa 13
Slika 10: Naredba za instalaciju twig-a 14
Slika 11: Naredba za instalaciju Encore-a 14
Slika 12: Naredba za instalaciju Bootstrap-a 15
Slika 13: Naredba za instalaciju jQuery-a 15
Slika 14: Naredba za instalaciju orm paketa 15
Slika 15: Naredba za instalaciju MakerBundle-a 15
Slika 16: Konfiguracija baze podataka 16
Slika 17: Naredba za generiranje entiteta 17
Slika 18: Entiteti 17
Slika 19: Naredba za nadopunu 17
Slika 20: Naredba 19
Slika 21: Naredba za instalaciju EasyAdmin-a 20
Slika 22: Naredba za kreiranje naslovnice 20
Slika 23: Izgled naslovnice 21
Slika 24: Naredba za kreiranje upravljača 21
Slika 25: Upravljači 22
Slika 26: Administracija Obavijesti 23
Slika 27: Obrazac za obavijest 24
Slika 28: Instalacija SecurityBundle-a 25
Slika 29: Instalacija obrasca za prijavu 25
Slika 30: Obrazac za prijavu 26
Slika 31: Naredba za verifikaciju email-a 27
Slika 32: Naredba za instalaciju obrasca za registraciju 27
Slika 33: Obrazac za registraciju 27
Slika 34: Naslovnica 28
Slika 35: Galerija 28
Slika 36: Tečajevi 29
Slika 37: Termini 29
Slika 38: Obavijesti 30