AKADEMIJA TEHNIČKO-UMETNIČKIH STRUKOVNIH STUDIJA BEOGRAD

ODESK VISOKA ŠKOLA ZA INFORMACIONE I KOMUNIKACIONE TEHNOLOGIJE

INTERNET TEHNOLOGIJE

Web aplikacija za prezentaciju i prodaju e-kurseva „Gačanović Academy”

ZAVRŠNI RAD

Mentor:

Dr Nenad Kojić, dipl. inž.

Student:

Marko Gačanović 38/17

Beograd, 2023.

AKADEMIJA TEHNIČKO-UMETNIČKIH STRUKOVNIH STUDIJA BEOGRAD

ODESK VISOKA ŠKOLA ZA INFORMACIONE I KOMUNIKACIONE TEHNOLOGIJE

INTERNET TEHNOLOGIJE

Predmet: Web programiranje PHP 2

Tema: **Web aplikacija za prezentaciju i prodaju e-kurseva**

Ocena (_____)

Članovi komisije:

_____

_____

_____

Sadržaj

Contents

1 Uvod 8

2 Radno okruženje 9

3 Organizacija 10

3.1 Stranice 11

3.1.1 Home 11

3.1.2 Courses 12

3.1.3 Courses/{id} 13

3.1.4 Contact 14

3.1.5 Login 15

3.1.6 Author 16

3.1.7 Wishlist 17

3.1.8 Instructor 18

3.1.9 Instructor/{id}/edit 20

3.1.10 Cart 21

3.1.11 Checkout.Stripe 22

3.1.12 Learnings 31

3.1.13 Admin Logs 32

3.1.14 Admin Courses Create 33

3.1.15 Admin Courses {id} Edit 34

3.1.16 Admin Categories Create 35

3.1.17 Admin Categories {id} Edit 36

3.1.18 Admin Topics Create 37

3.1.19 Admin Topics {id} Edit 38

3.1.20 Admin Users Create 39

3.1.21 Admin Users {id} Edit 40

3.1.22 Admin Contact 41

3.1.23 Admin Orders 42

4 Kodovi 43
4.1 Models 43
4.1.1 Category.php 43
4.1.2 ContactMail.php 44
4.1.3 Course.php 45
4.1.4 CourseOrder.php 48
4.1.5 CourseTopic.php 49
4.1.6 Instructor.php 49
4.1.7 Lesson.php 50
4.1.8 Order.php 51
4.1.9 Role.php 52
4.1.10 Topic.php 53
4.1.11 User.php 54
4.1.12 Wish.php 56
4.2 Controllers 57
4.2.1 AuthController.php 57
4.2.2 CartController.php 59
4.2.3 CheckoutController.php 60
4.2.4 ContactController.php 63
4.2.5 FrontController.php 64
4.2.6 InstructorController.php 68
4.2.7 LessonController.php 72
4.2.8 WishController.php 74
4.2.9 Admin/CategoryController.php 76
4.2.10 Admin/ContactController.php 79
4.2.11 Admin/CourseController.php 81
4.2.12 Admin/TopicController.php 87
4.2.13 Admin/UserController.php 89
4.3 Middlewares 92
4.3.1 AdminMiddleware.php 92
4.3.2 AuthoriseMiddleware404.php 93
4.3.3 AuthoriseMiddleware.php 93
4.3.4 RecordAccessToPage.php 94

4.4 Services 95
4.4.1 Helper.php 95
4.4.2 ImageHelper.php 95
4.4.3 Logs.php 96
4.4.4 UserService.php 97
4.5 Requests 98
4.5.1 AddCategoryRequest.php 98
4.5.2 AddCourseRequest.php 98
4.5.3 AddUpdateTopicRequest.php 100
4.5.4 AddUserRequest.php 100
4.5.5 ContactRequest.php 101
4.5.6 FilterCoursesRequest.php 102
4.5.7 LoginRequest.php 103
4.5.8 RegistrationRequest.php 104
4.5.9 UpdateCategoryRequest.php 105
4.5.10 UpdateCourseRequest.php 106
4.5.11 UpdateUserRequest.php 107
4.6 Views 108
4.6.1 components 108
4.6.1.1 admin/form.blade.php 108
4.6.1.2 instructor/form.blade.php 113
4.6.1.3 single_course.blade.php 118
4.6.2 email 119
4.6.2.1 contact.blade.php 119
4.6.3 fixed 119
4.6.3.1 admin 119
4.6.3.1.1 header.blade.php 119
4.6.3.1.2 head.blade.php 120
4.6.3.1.3 footer.blade.php 120
4.6.3.1.4 sidebar.blade.php 120
4.6.3.2 user 122
4.6.3.2.1 banner.blade.php 122
4.6.3.2.2 footer.blade.php 122

4.6.3.2.3 head.blade.php 123
4.6.3.2.4 header.blade.php 124
4.6.3.2.5 scripts.blade.php 125
4.6.4 layout 126
4.6.4.1 admin.blade.php 126
4.6.4.2 instructor.blade.php 126
4.6.4.3 user.blade.php 127
4.6.5 pages 127
4.6.5.1 admin 127
4.6.5.1.1 categories_edit.blade.php 127
4.6.5.1.2 categories_blade.php 128
4.6.5.1.3 contact_mails.blade.php 129
4.6.5.1.4 courses_edit.blade.php 129
4.6.5.1.5 courses.blade.php 132
4.6.5.1.6 log.blade.php 134
4.6.5.1.7 topics_edit.blade.php 137
4.6.5.1.8 topics.blade.php 138
4.6.5.1.9 users_edit.blade.php 138
4.6.5.1.10 users.blade.php 141
4.6.5.2 Intructor 143
4.6.5.2.1 instructor.blade.php 143
4.6.5.3 user 144
4.6.5.3.1 author.blade.php 144
4.6.5.3.2 cart.blade.php 145
4.6.5.3.3 checkout-cancel.blade.php 146
4.6.5.3.4 checkout-success.blade.php 147
4.6.5.3.5 contact.blade.php 149
4.6.5.3.6 courses.blade.php 151
4.6.5.3.7 home_blade.php 154
4.6.5.3.8 learnings.blade.php 155
4.6.5.3.9 login.blade.php 156
4.6.5.3.10 orders_blade.php 158
4.6.5.3.11 single_course.blade.php 159

4.6.5.3.12 wishes.blade.php 161
4.7 Routes 162
4.7.1 web.php 162
4.8 JavaScript 164
4.8.1 main.js 164
4.8.2 cart.js 181
4.8.3 contact.js 183
4.8.4 wish.js 185
4.9 CSS 188
5 Baza 189
6 Zaključak 190
7 Literatura 191

1 Uvod

Web aplikacija „Gačanović Academy” je online platforma izrađena u Laravel-u, na kojoj korisnici mogu da izvrše kupovinu željenih e-kurseva iz različitih kategorija i tema.

Prilikom dolaska na aplikaciju korisnici imaju pregled svih e-kurseva koji su dostupni za kupovinu. Klikom na dugme za pretragu ipisuju se e-kursevi koji zadovoljavaju kriterijume iz filtera za pretragu.

Klikom na bilo koji e-kurs mogu se videti svi detalji samog e-kursa.

Autorizovani korisnici mogu da budu i autori e-kursa tako što na posebnoj stranici odgovore na jedno pitanje iz ankete i pošalju odgovor. Nakon toga imaju mogoćnost kreiranja i izmene sopstvenih e-kurseva.

Uspešnom registracijom i logovanjem omogućuje se korisnicima da željeni e-kurs dodaju u listu želja ili da odmah dodaju u korpu. Sistem korpe je urađen AJAX-om i uz pomoć Local Storage-a. Na stranici za korpu, korisnik ima uvid u e-kurseve koje želi da kupi i gde može ukloniti iste iz korpe.

Nakon što odluče da kupe e-kurs, jednostavnim klikom na dugme redirektuju se na stranicu gde treba da ostave svoje podatke sa kartice kako bi se izvršilo plaćanje preko Stripe-a. Kada se plaćanje uspešno izvrši onda imaju pristup sadržaju i detaljima samog e-kursa.

Adminu je omogućeno da vrši unos, brisanje i izmenu sledećih stavki:

- E-kursevi
- Kategorije
- Teme
- Korisnici

Adminu je takođe omogućen pregled e-mailova, kao i brisanje, nakon što ih korisnici pošalju kroz kontakt formu.

Na početnoj strani za admina nalazi se tabela sa logovima koji prate sve ključne aktivnosti korisnika na sajtu. Admin ima pregled svih porudžbina u sistemu.

2 Radno okruženje

Razvojni alati:

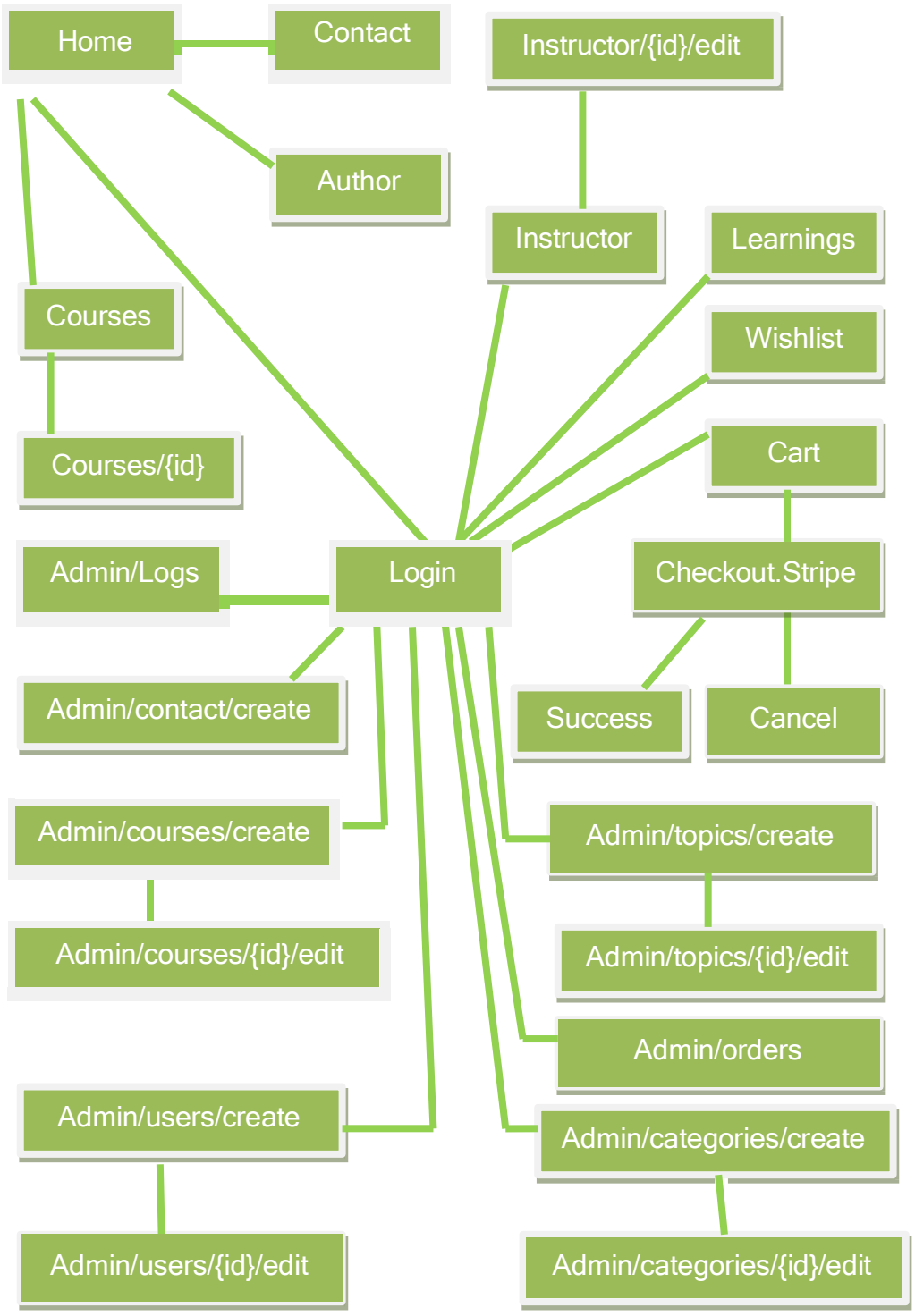
- Visual Studio Code
- phpMyAdmin
- XAMPP

Tehnologije:

- PHP (Laravel Framework)
- JavaScript
- HTML
- CSS
- Bootstrap 4
- jQuery
- MySQL

Admin nalog
Username: Admin08
Password: admin007

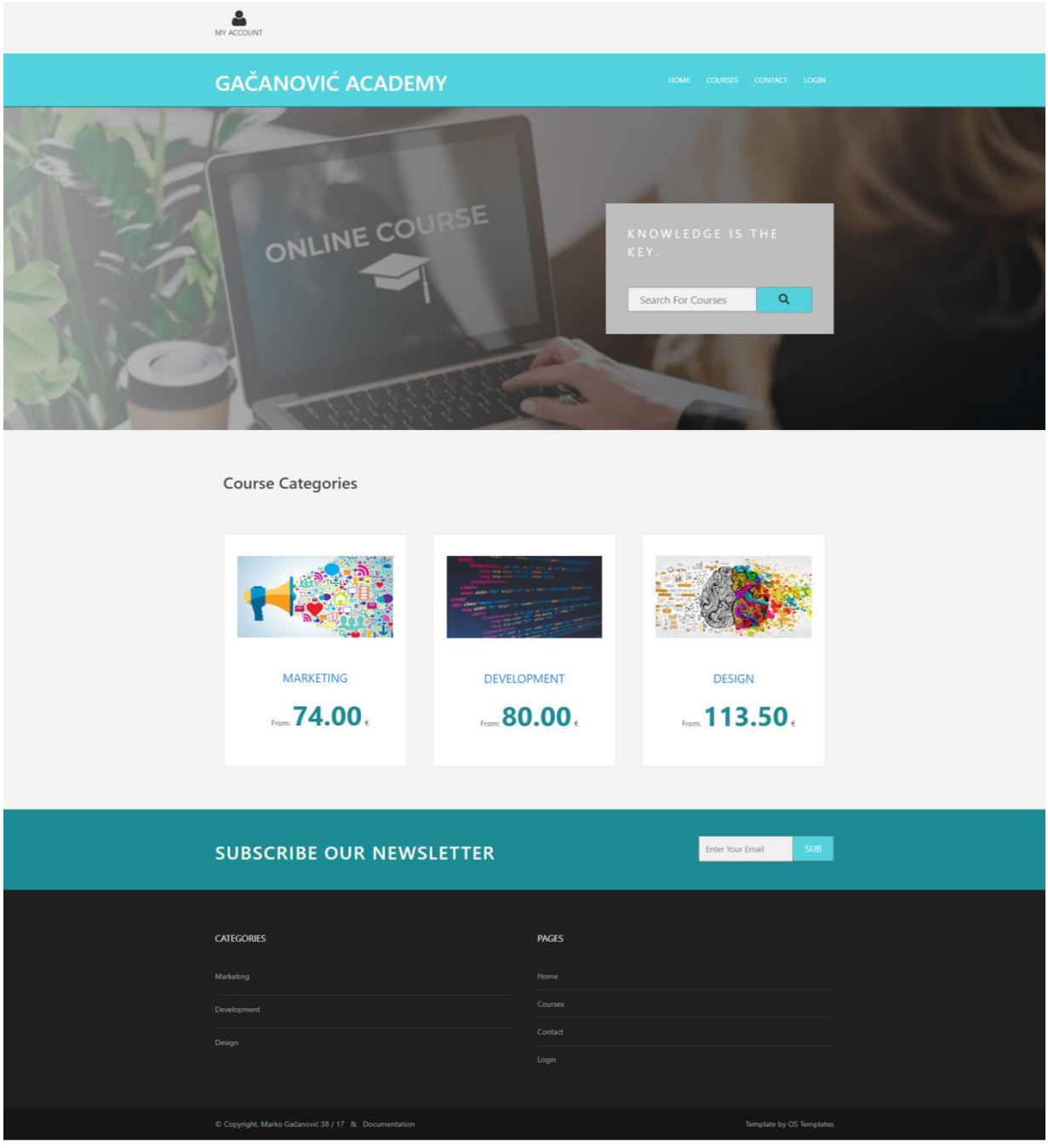
3 Organizacija



Slika 1. organizacija sajta

3.1 Stranice

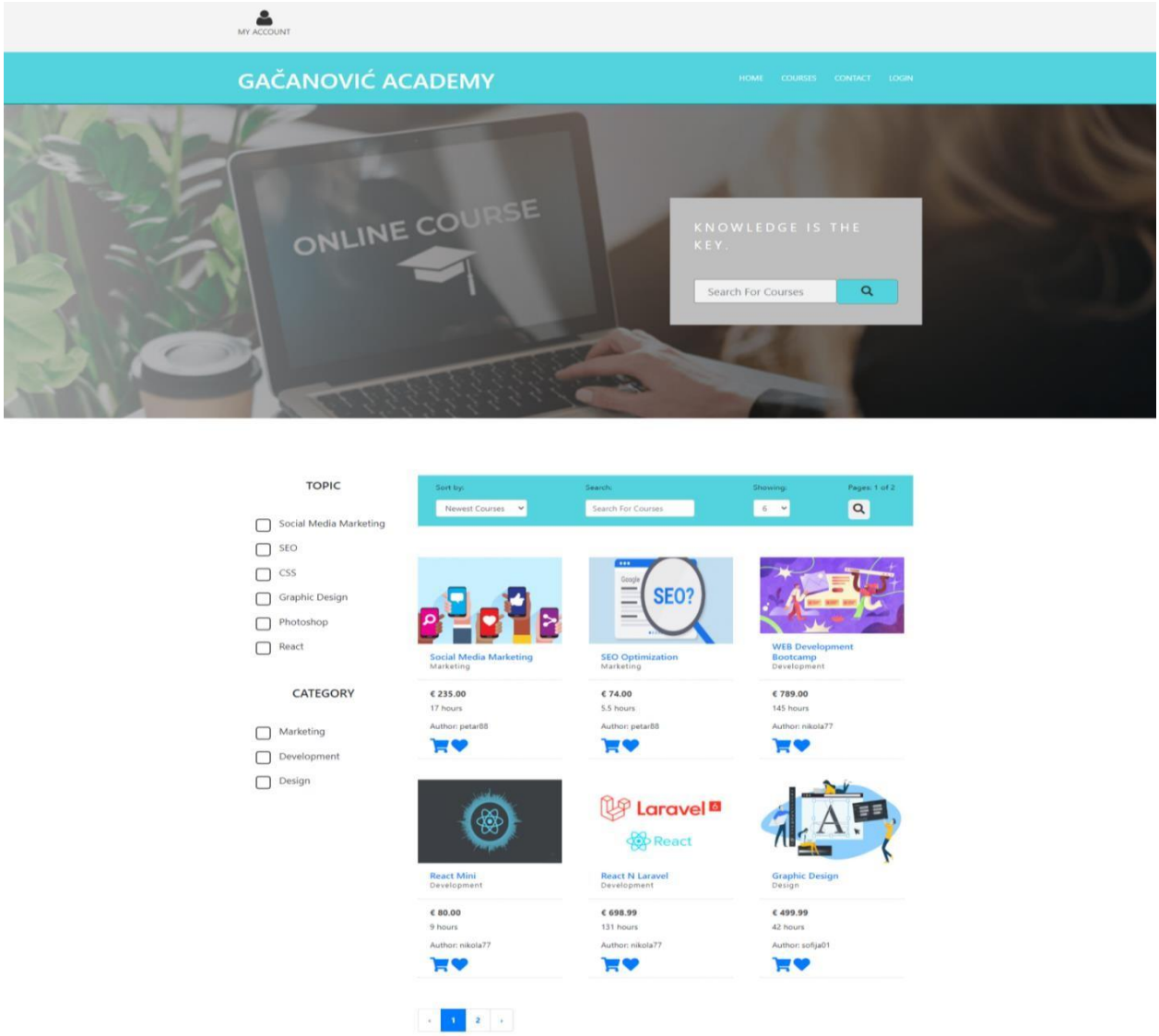
3.1.1 Home



Slika 2. Home strana

Na home strani (slika 2) se nalaze dinamički ispisane kategorije. Klikom na naziv kategorije prikazuju se kursevi iz te kategorije na stranici za kurseve. Prikazane cene predstavljaju najnižu cenu kursa u izabranoj kategoriji.

3.1.2 Courses

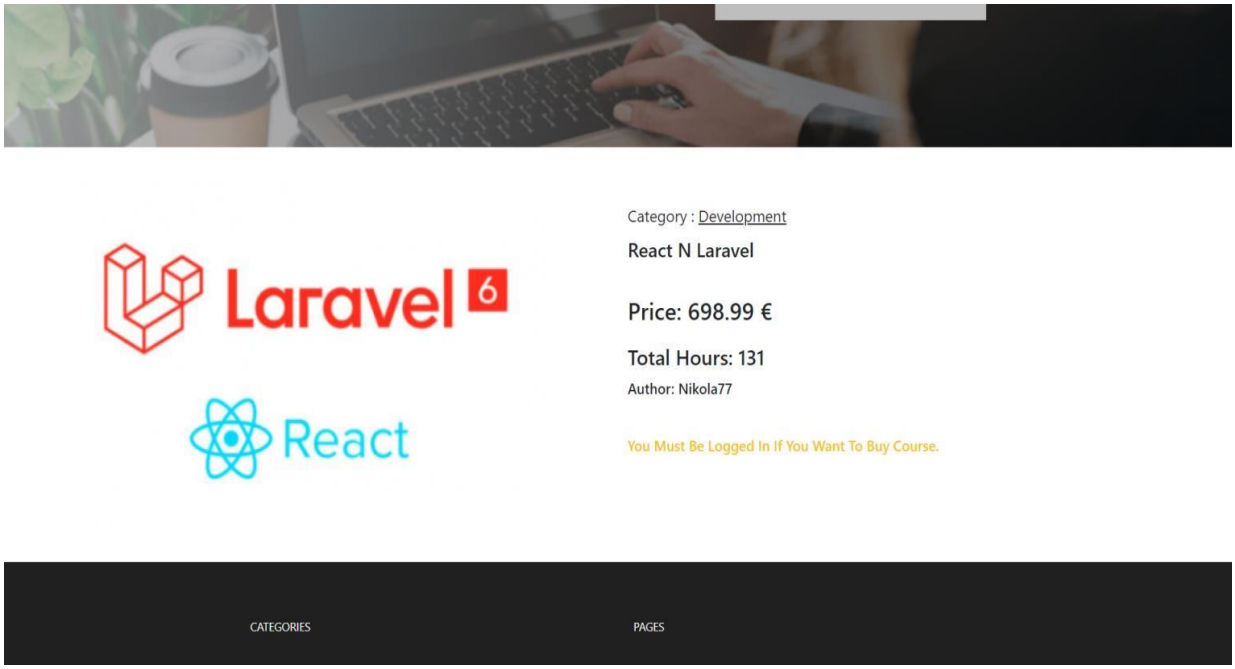


Slika 3. Courses strana

Sav sadržaj courses strane (slika 3) se učitava dinamički iz baze podataka. Filteri su napisani dinamički. Klikom na dugme za pretragu ili paginaciju ispisuju se e-kursevi koji zadovoljavaju kriterijume i određeni broj strana u paginaciji.

Ako je korisnik autor nekih e-kurseva onda ih neće on videti na ovoj stranici. Omogućeno je da se bira ukupan broj e-kurseva koji može da se vidi na stranici, u skladu sa paginacijom, sortiranje po ceni e-kurseva, pretraga po autoru ili nazivu e-kursa, filtriranje po temama i kategorijama.

3.1.3 Courses/{id}

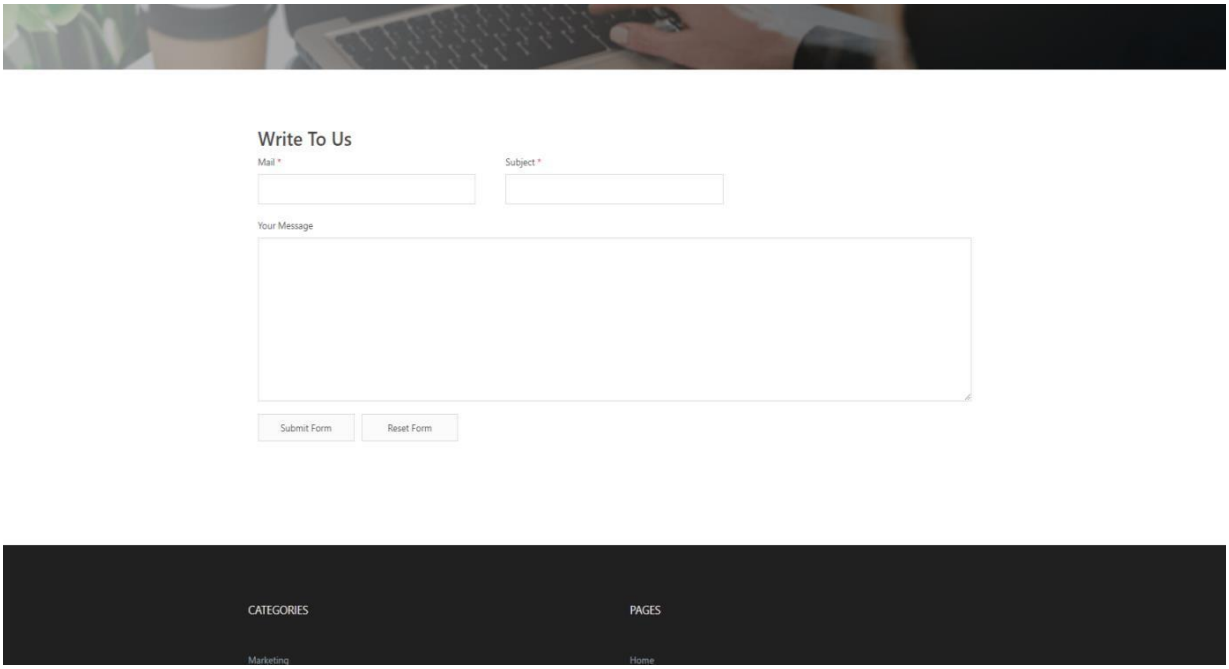


Slika 4. Course/{id} strana

Prikaz strane pojedinačnog e-kursa (slika 4).

Samo autorizovani korisnici mogu da dodaju e-kurs u korpu ili da ga dodaju u listu želja. Prikazuje se naziv kategorije, naziv e-kursa, cena, broj sati, naziv autora.

3.1.4 Contact



Slika 5. Contact strana

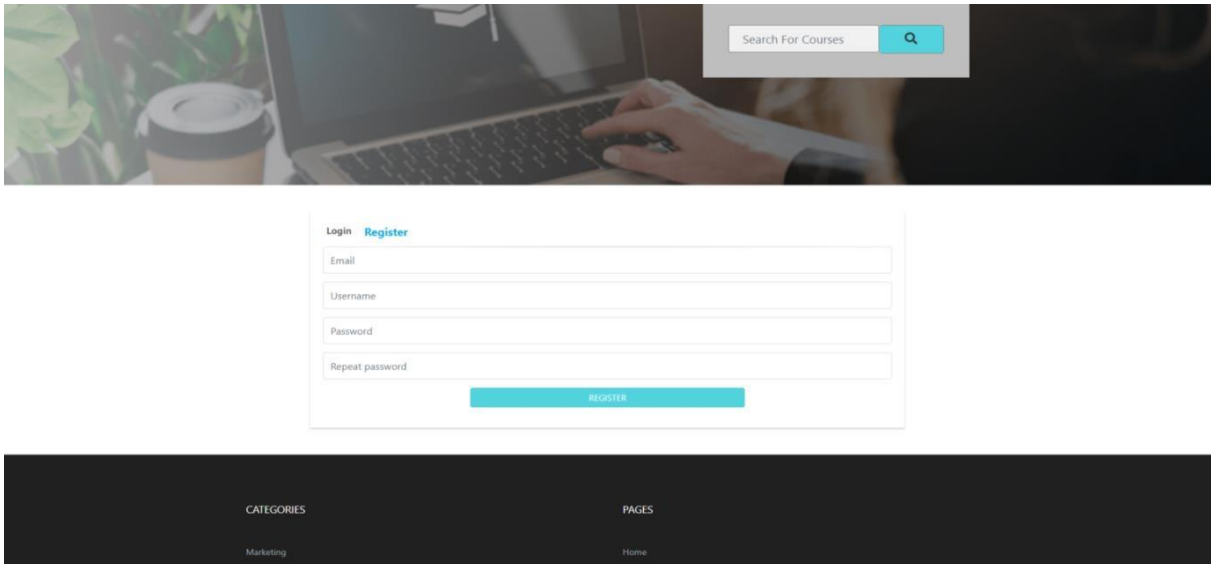
Na kontakt strani (slika 5) korisnik može da pošalje poruku administratoru. Neophodno je da se unesu svrha poruke, email adresa i sadržaj poruke.

Izvršena je serverska i klijentska provera podataka korišćenjem regularnih izraza.

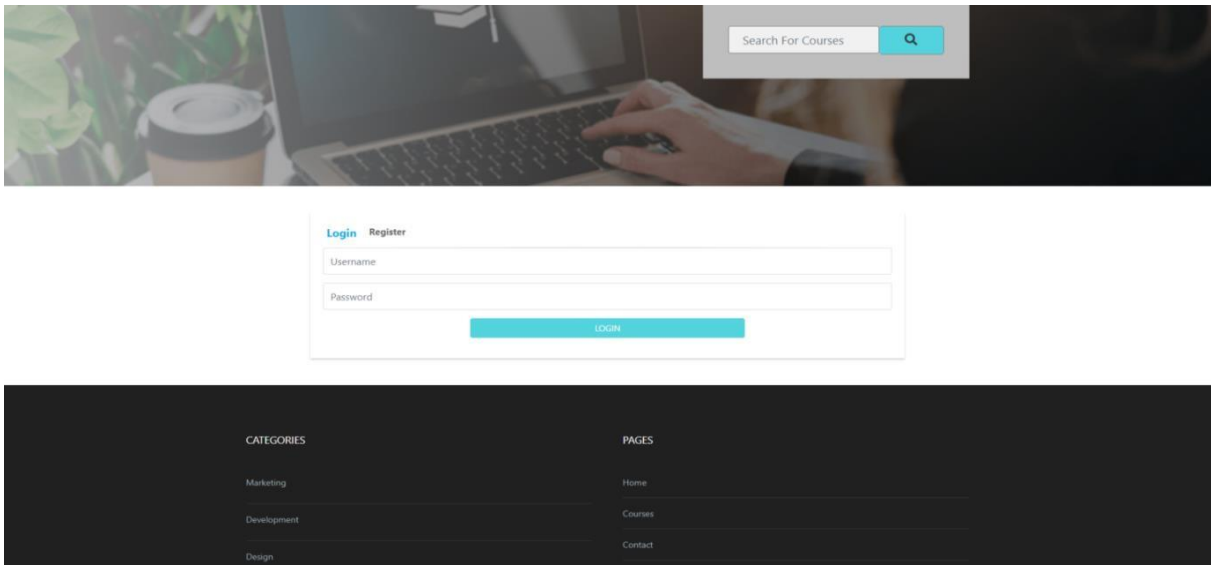
Klikom na dugme za slanje poruke svi navedeni podaci se upisuju u bazu podataka.

Postoji i dugme za čišćenje input polja.

3.1.5 Login



Slika 6. Register strana

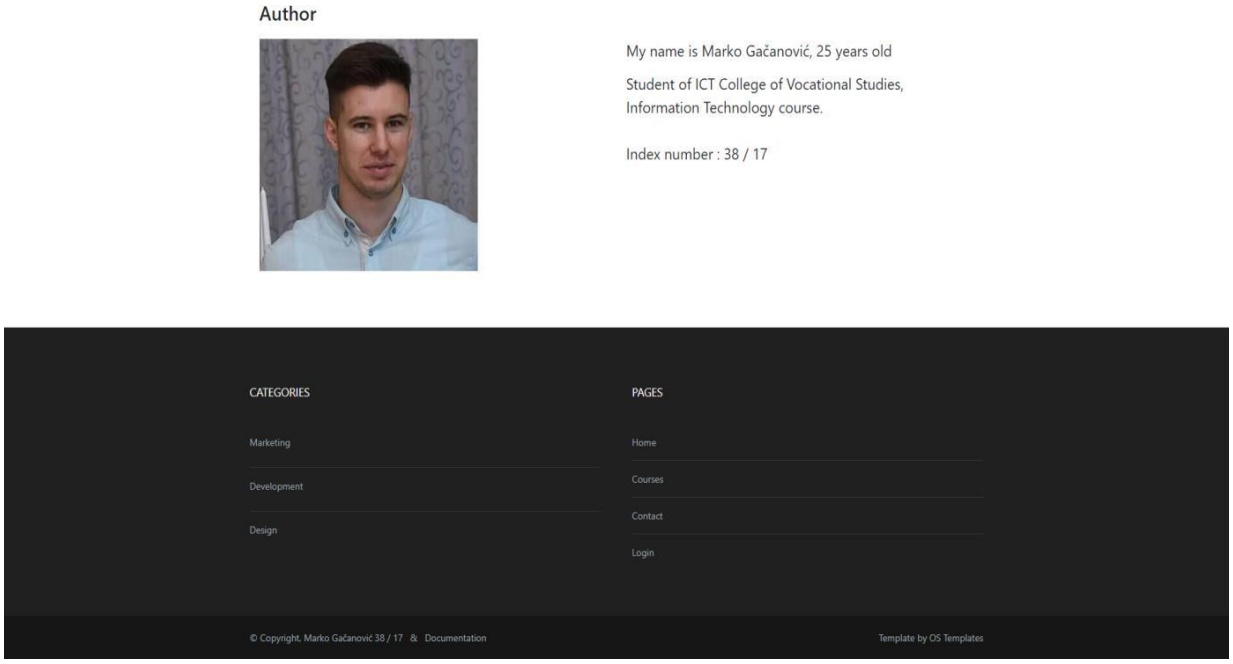


Slika 7. Login strana

Provera podataka prilikom registracije (slika 6) je urađena na klijentskoj(AJAX) i na serverskoj strani.

Provera podataka prilikom logovanja (slika 7) se vrši na serverskoj strani i ako podaci nisu ispravni kreira se sesija koja obaveštava korisnika gde je pogrešio prilikom logovanja. Ako su ,,username“ i ,,password“ ispravni kreira se sesija koja sadrži podatke korisnika. Korisnicima se ažurira aktivnost i vreme logovanja.

3.1.6 Author

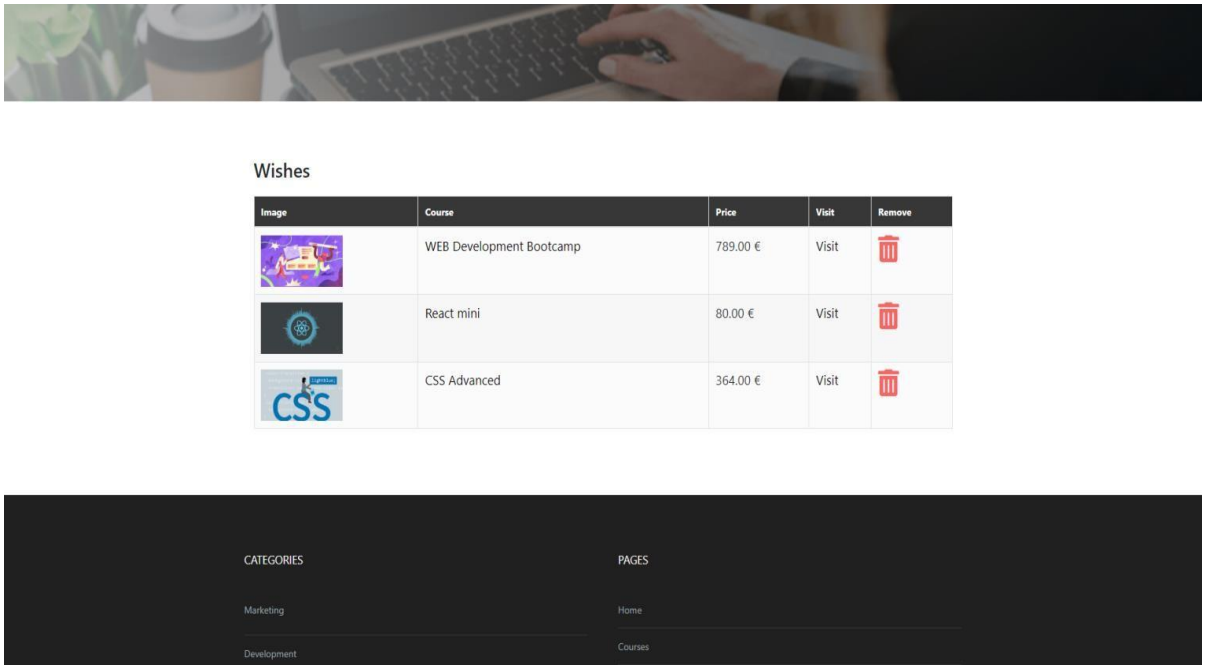


Slika 8. Author strana

Strana o opisu autora projekta (slika 8).

Link ka strani se nalazi u dnu futera.

3.1.7 Wishlist



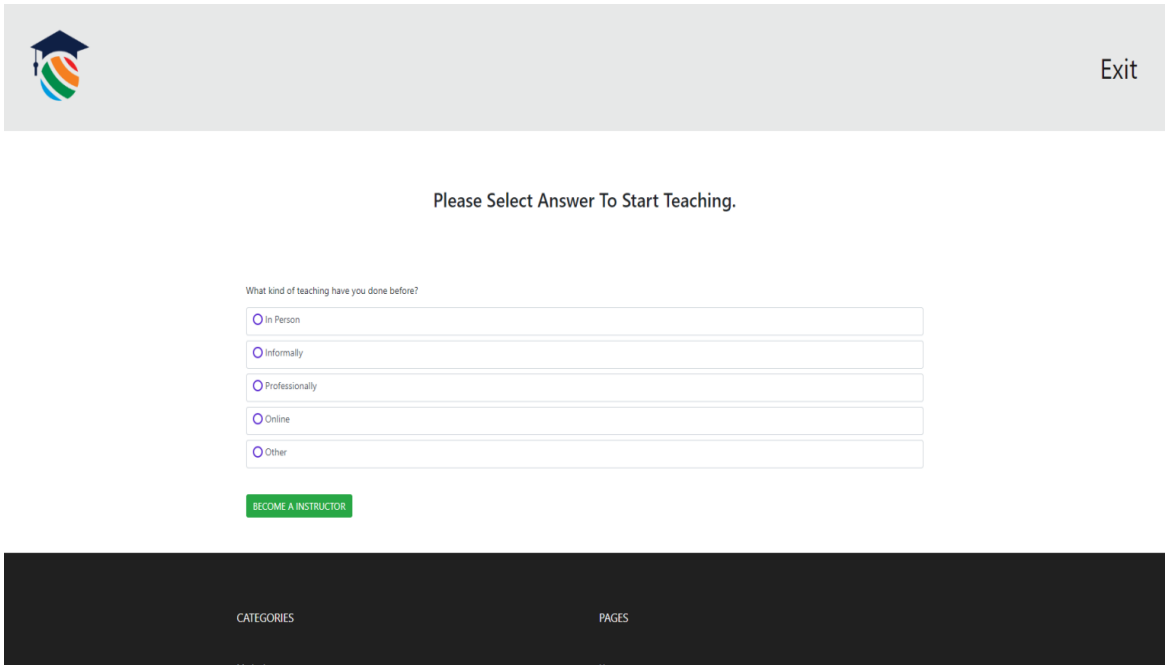
Slika 9. Wishlist strana

Strana sa listama želja (slika 9) svakog korisnika.

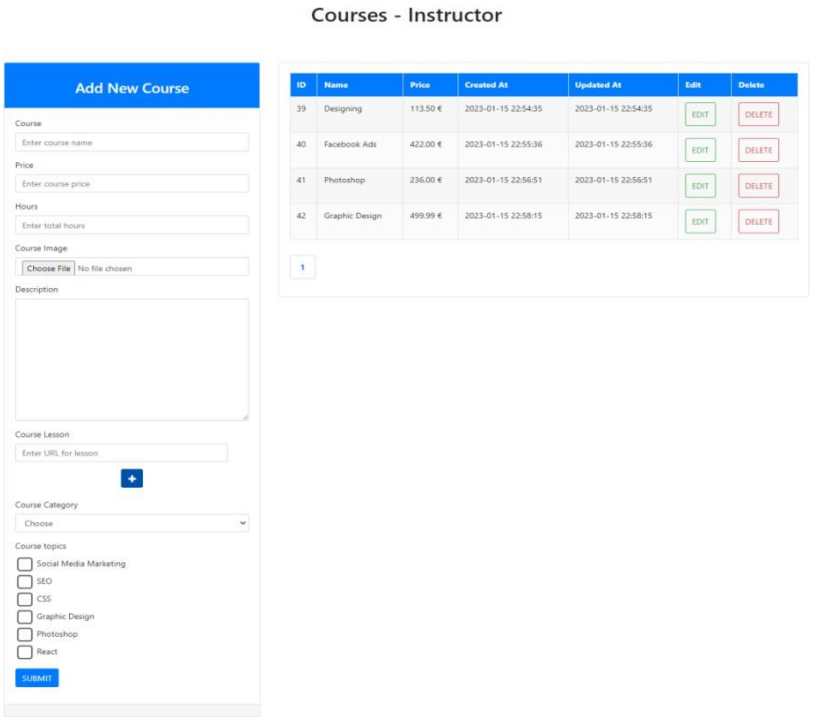
Svaki autorizovani korisnik može da izabere e-kurs koji želi da doda u listu želja. Proverava se da li je korisnik već uneo e-kurs u listu želja. Ako jeste onda mu se onemogućuje da isti e-kurs opet doda u listu.

Klikom na ikonicu u zaglavlju otvara se stranica koja sadrži tabelu sa svim dodatim željama tog korisnika i omogućuje se brisanje iz liste putem AJAX-a.

3.1.8 Instructor



Slika 10. Instructor strana – korak pre



Slika 11. Instructor strana

Strana namenjena autorima e-kurseva. (Slika 10) prikazuje izgled stranice za korisnike koji žele tek da postanu autori.

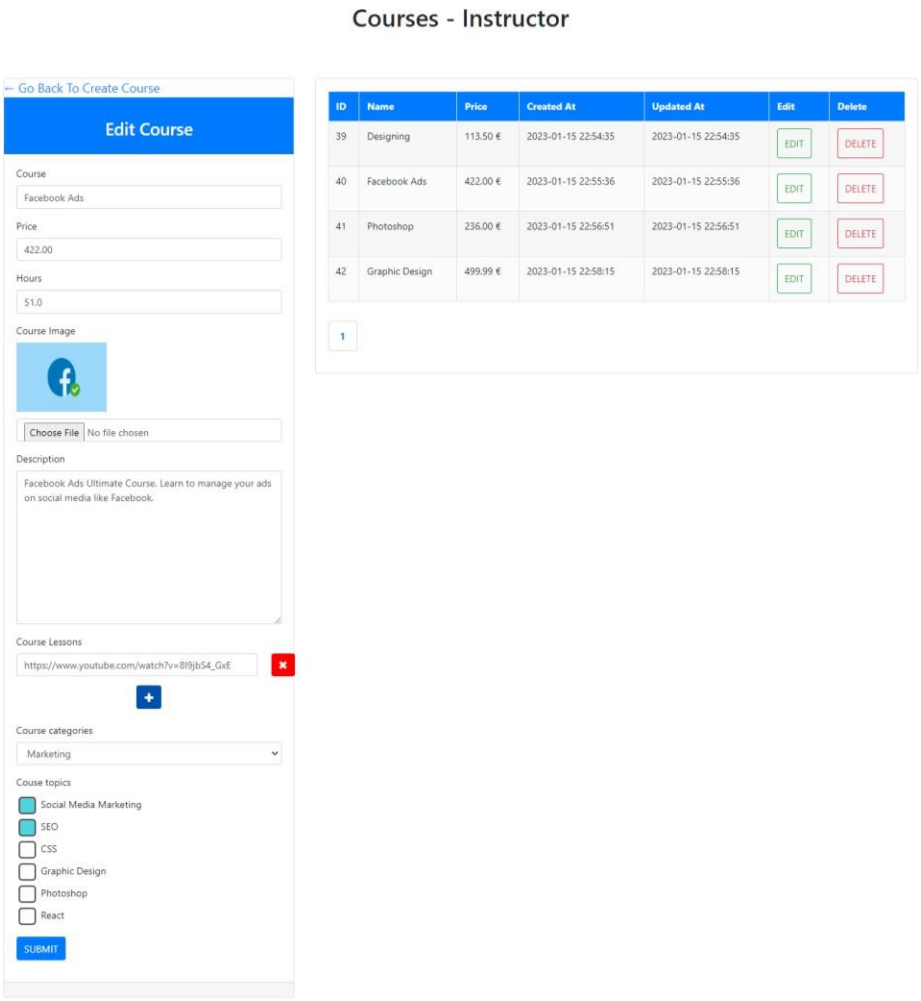
Odgovorom na pitanje iz ankete autorizovani korisnik automatski postaje autor i redirektuje se na stranicu za instruktora prikazanu na (slici 11).

Na toj strani se vidi forma za unos novih e-kurseva kao i tabelarni prikaz svih do sada kreiranih e-kurseva tog autora.

Administrator može da promeni privilegiju korisniku da više nema ulogu autora.

Stranica za kreiranje i izmenu e-kursa je identična za instruktora i za administratora, što će se videti kasnije na (slikama 21 i 22).

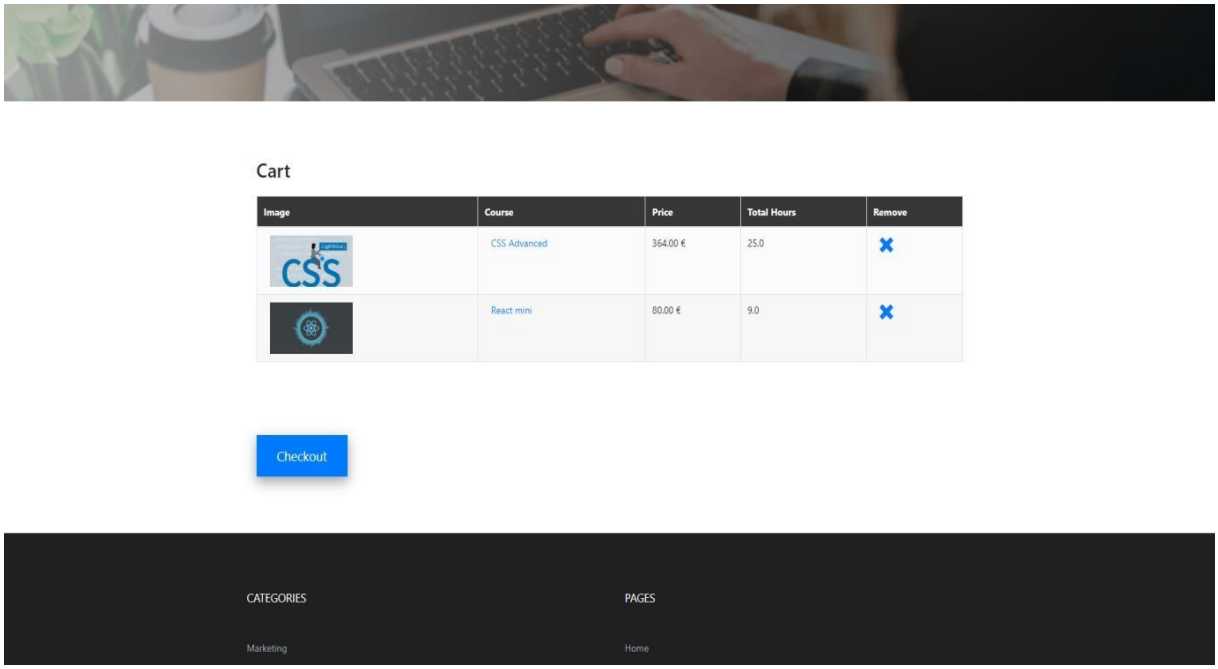
3.1.9 Instructor/{id}/edit



Slika 12. Instructor edit strana

Stranica za izmenu e-kursa od strane autora (slika 12) gde autor može da izmeni detalje svojih e-kurseva. U tabelarnom prikazu AJAX-om može da obriše svoj e-kurs, paginacija je takođe AJAX-om kreirana.

3.1.10 Cart



Slika 13. Cart strana

Strana gde se prikazuje sadržaj korisničke korpe (slika 13).

Sistem korpe je urađen AJAX-om i uz pomoć LocalStorage-a. Nije moguće kupiti 2 ista e-kursa za jednog korisnika. Omogućeno je brisanje iz korpe AJAX-om. Kupovinu mogu da vrše korisnici koji nisu administratori. Kada se plaćanje obavi korpa se prazni.

3.1.11 Checkout.Stripe

Klikom na dugme ,,Checkout“ dolazimo do Stripe-a.

Treba set-ovati API key i kreirati sesiju. Potrebno je instalirati Stripe PHP package**:** ,,**composer require stripe/stripe.php**,,

Naš server pravi request ka Stripe serveru (generiše sesiju sa e-kursevima).

Nalog na Stripe-u se kreira sa proizvoljnim podacima kako bi se omogućilo test okruženje.

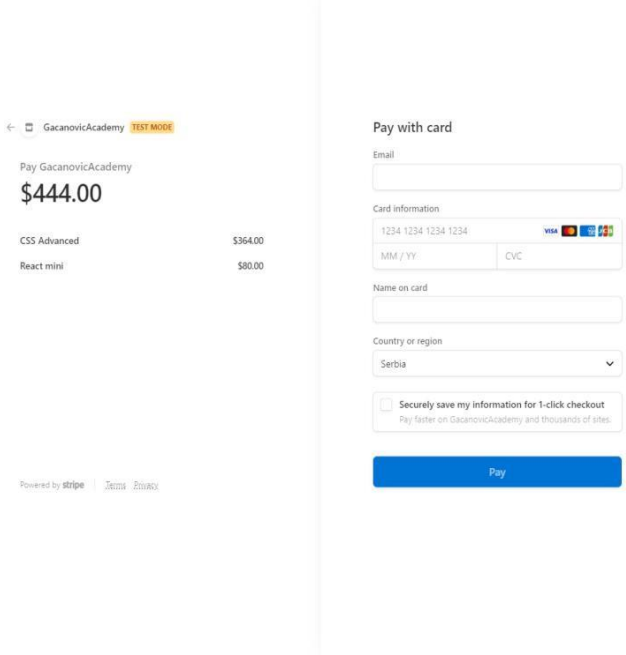
Naš server drži secret key sa kojim se šalje request ka Stripe-u, inače nas Stripe ne prepoznaje. Secret key se uzima sa Stripe-ovog dashboard-a.

URL:https://dashboard.stripe.com/test/dashboard

Slika 14. Stripe secret keys

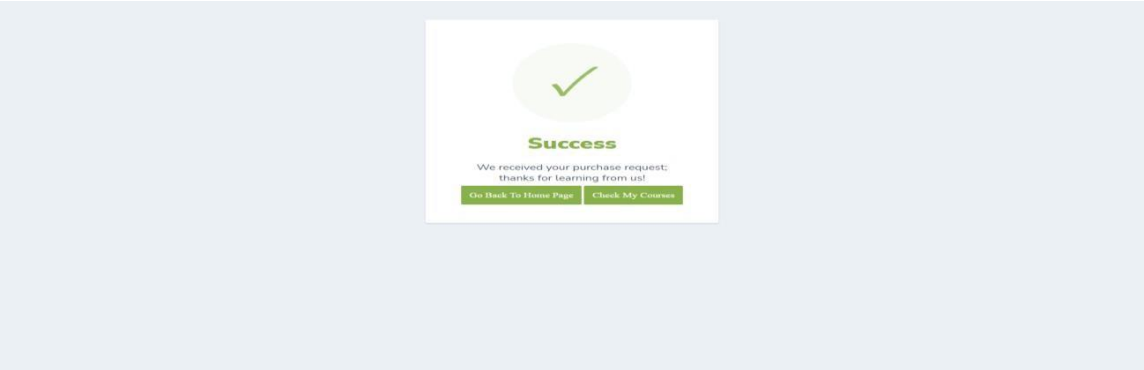
Sledeći korak, Stripe će generisati session objekat i vratiti ga našem serveru. Dalje naš server pravi porudžbinu koja je ,,unpaid“ i čuva je u bazi podataka i vezuje primljeni session_id sa porudžbinom.

Naš server će uzeti session_url iz session objekta i onda imamo dve opcije. Naš server može redirektovati korisnika na session_url direktno ili može vratiti session_id Browser-u i onda on redirektuje korisnika na stranicu za plaćanje. U oba slučaja rezultat je isti: videćemo neku stranicu koja ima Stripe checkout formu.

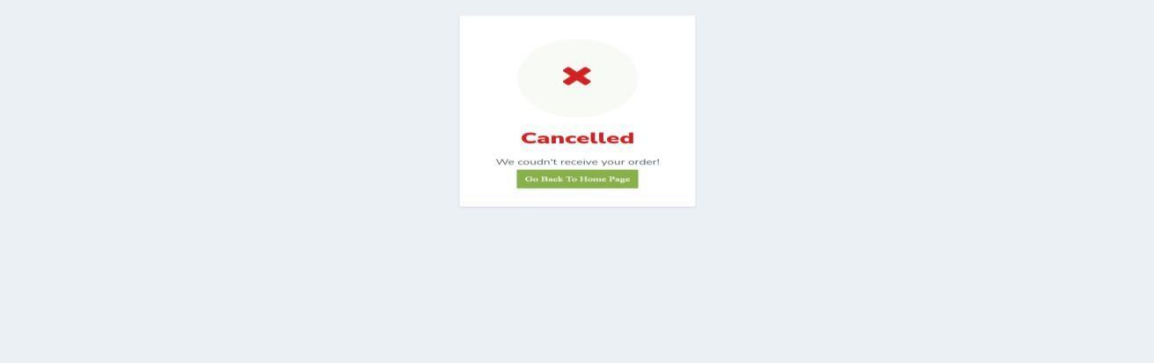


Slika 15. Stripe checkout forma

Kada korisnik popuni formu (slika 15) i klikne na ,,Pay“ on će biti redirektovan na ,,success“ (slika 16) ili ,,cancel“ (slika 17) stranicu koju mi konfigurišemo.



Slika 16. Success strana



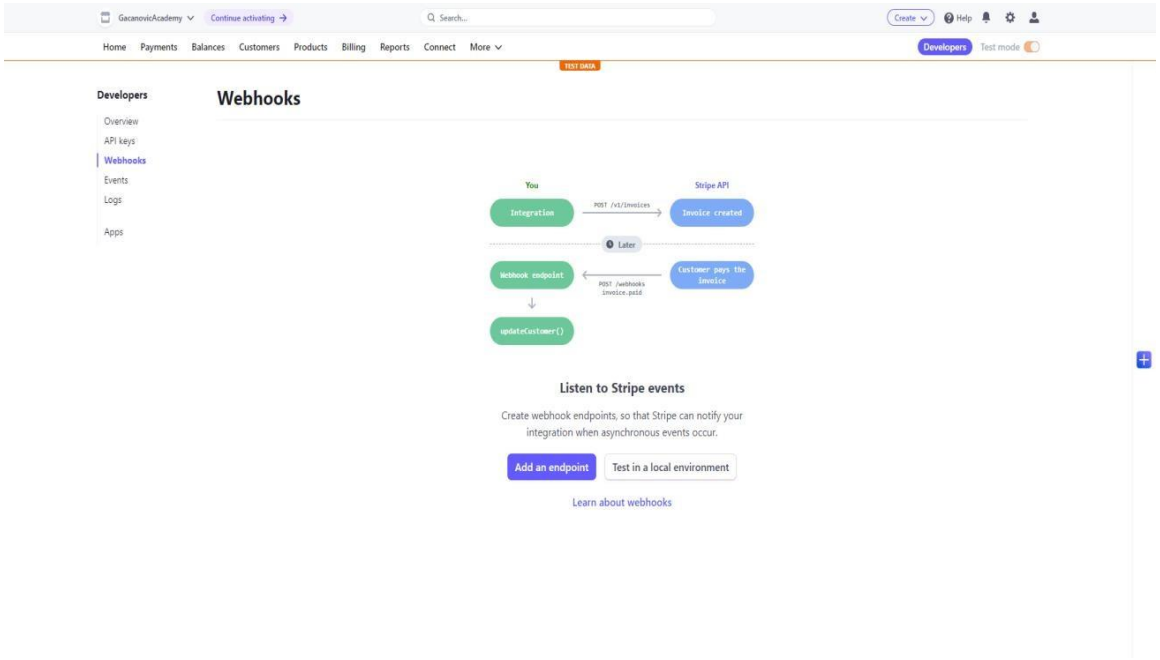
Slika 17. Cancel strana

Prosleđuje se u success i session id što će biti bitno za validaciju sesije kasnije. Kada god je korisnik redirektovan na success url pravi se request sa klijenta ka serveru, i server će imati pristup id-u sesije.

U tom momentu uzimamo session_id i pravimo request ka Stripe-u prosleđujući naš secret token i session id i pitamo da nam Stripe validira taj id. Stripe će za nas validirati i vratiti nam session objekat.

U ovom momentu selektuje se porudžbia koja je ,,unpaid“, na osnovu session id-a. Pre toga proveravamo da li postoji ta porudžbina, isto uz pomoć session id-a. Ako postoji ta porudžbina onda ćemo je označiti kao ,,paid“ i vraćamo success stranu korisniku.

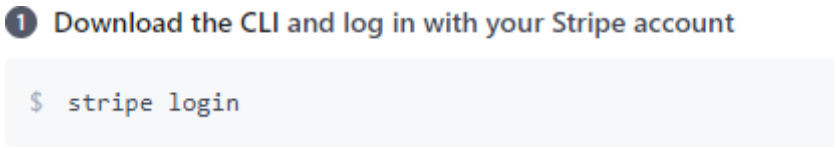
Kada korisnik klikne na ,,Pay“, pre nego što se redirekcija dogodi na Browser-u internet konekcija može da se izubi, može da nestane struje ili korisnik može greškom da zatvori tab i onda se redirekcija neće izvršiti. U ovom slučaju novac će se skinuti sa računa korisnika, ali id sesije se neće proslediti serveru i server neće označiti porudžbinu kao plaćenu. Ovo treba da izbegnemo. Na Stripe dashboard-u možemo mnoge stvari uraditi – videti plaćanja, itd. Ono što je potrebno nama jeste da konfigurišemo Webhooks (to su posebni URL-ovi na koje će Stripe praviti request-ove kada god se nešto desi). Na mnogo događaja se mogu konfigurisati webhook-ovi, ali nama su potrebni checkout evente-ovi.

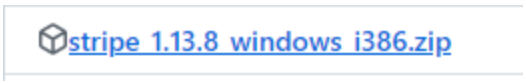


Slika 18. Webhooks

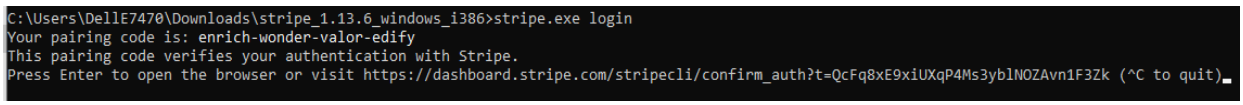
Request dolazi sa Stripe-a i imamo csrf validacijski problem, zato treba da se isključi za URL ,,/webhook“ u klasi VerifyCsrfToken.

Kada izaberemo da testiramo u lokalnom okruženju (slika 18) u tom slučaju moramo da skinemo stripeCLI.

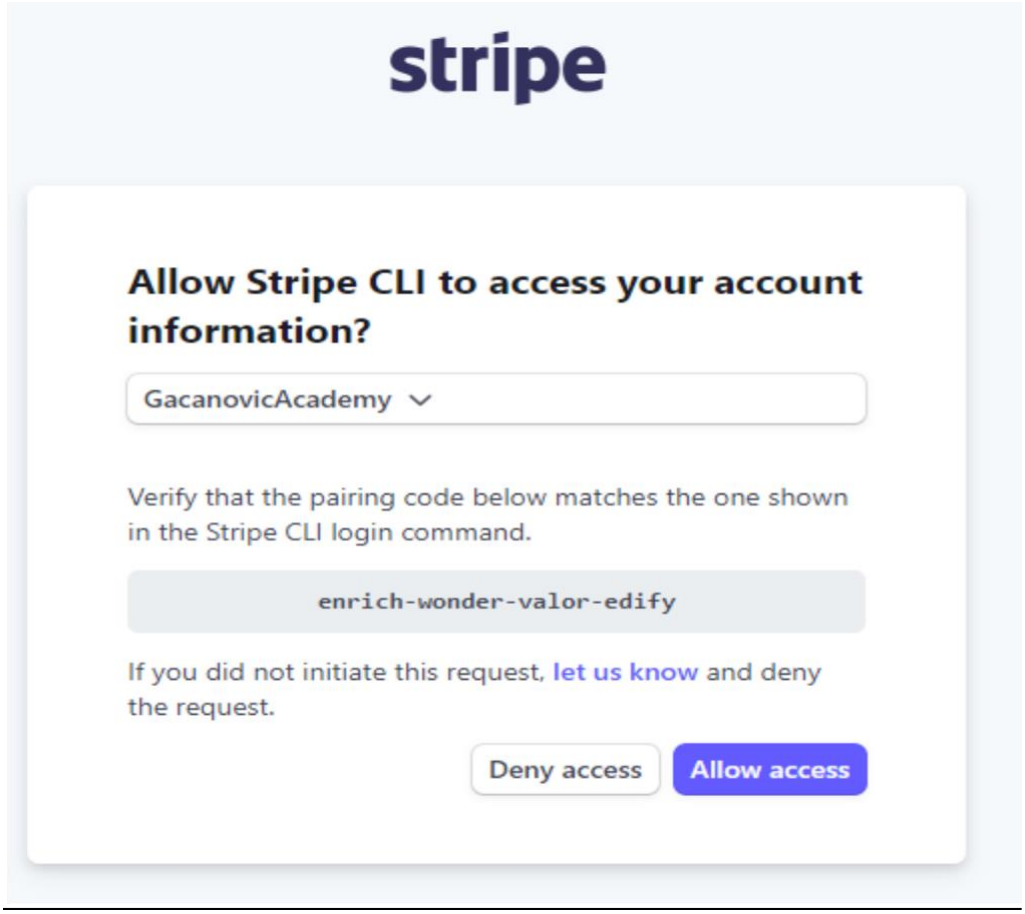




URL: https://github.com/stripe/stripe-cli/releases/tag/v1.13.8

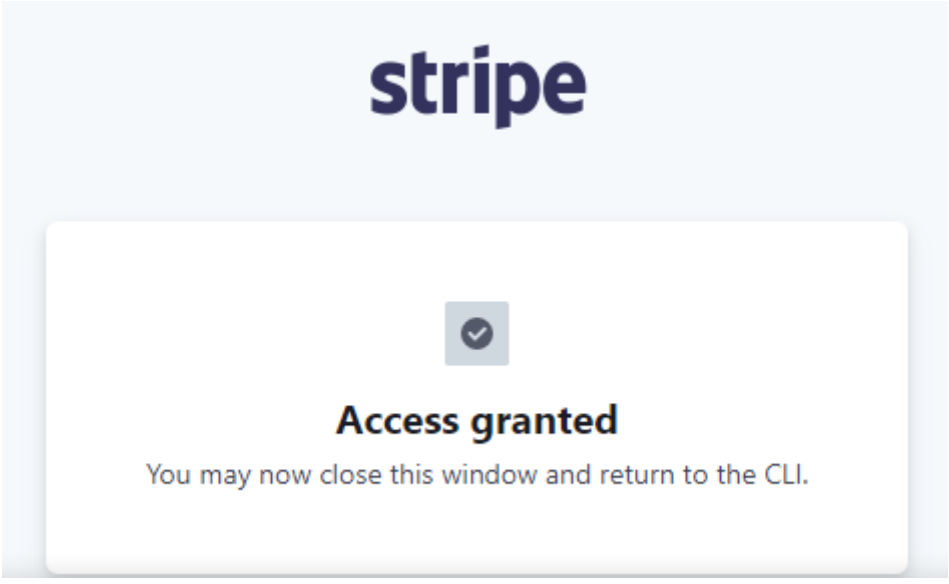


Kada pritisnemo enter odvodi nas na stranicu:https://dashboard.stripe.com/auth_stripecli




```
Your pairing code is: enrich-wonder-valor-edify
```

Uspoređujemo kodove iz terminala sa kodom sa Stripe-a. Ako su isti onda samo kliknemo ,,Allow access“.



```
> Done! The Stripe CLI is configured for GacanovicAcademy with account id acct_1M6FPJLWXnryLuL4

Please note: this key will expire after 90 days, at which point you'll need to re-authenticate.
```

Sada smo autorizovani u Stripe CLI-u.

URL za webhooks:
https://dashboard.stripe.com/test/webhooks/create?endpoint_location=local

Sada ide drugi korak, odnosno prosleđivanje evente-ova na naš Webhook. Kada god se dešava checkout proces Stripe ne može da pošalje request na naš server jer je naš server lokalno hostovan.

Međutim Stripe CLI može da osluškuje ove Webhook-ove i da ih prosledi na našu lokalnu instalaciju.

```
C:\Users\DellE7470\Downloads\stripe_1.13.6_windows_i386>stripe listen --forward-to localhost:8000/webhook
A newer version of the Stripe CLI is available, please update to: v1.13.8
> Ready! You are using Stripe API Version [2022-11-15]. Your webhook signing secret is whsec_0fb25c7c9f0846d453948ad3651a63cb6b12f9507
6edae2e553b5b1ec3e56cd3 (^C to quit)
```

Dobili smo secret key.

Sada prelazimo na treći korak kada trigger-ujemo event sa CLI koji će doći na naš Webhook.

Otvaramo još jedan cmd.

Terminal 2:

```
C:\Users\DellE7470\Downloads\stripe_1.13.6_windows_i386>stripe.exe trigger payment_intent.succeeded
```

Terminal 1:

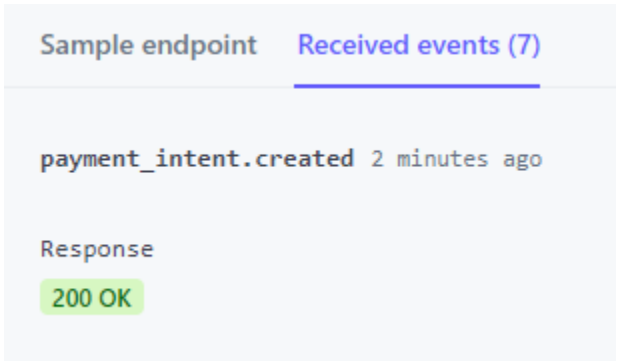
```
2023-01-19 23:28:27   --> charge.succeeded [evt_3MS6aqLWXnryLuL41krncfhI]
2023-01-19 23:28:27   --> payment_intent.succeeded [evt_3MS6aqLWXnryLuL41gRjBgvu]
2023-01-19 23:28:27   --> payment_intent.created [evt_3MS6aqLWXnryLuL41NHzejnu]
2023-01-19 23:28:27  <--  [200] POST http://localhost:8000/webhook [evt_3MS6aqLWXnryLuL41krncfhI]
2023-01-19 23:28:27  <--  [200] POST http://localhost:8000/webhook [evt_3MS6aqLWXnryLuL41NHzejnu]
2023-01-19 23:28:28  <--  [200] POST http://localhost:8000/webhook [evt_3MS6aqLWXnryLuL41gRjBgvu]
```

Tri requesta su se desila u našem Webhook-u, ali nam nisu relevantni. Relevantan nam je checkout process.

Čitamo input koji dolazi i konstruišemo event na osnovu: payload-a koji dolazi, signature header-a koji je return-ovan kao response header i endpoint secret-a.

Kada korisnik klikne na dugme ,,Pay“:

```
2023-01-19 23:38:25   --> checkout.session.completed [evt_1MS6kWLWXnryLuL4MB6J7GpJ]
2023-01-19 23:38:25   --> payment_intent.succeeded [evt_3MS6kULWXnryLuL40Cvh61KG]
2023-01-19 23:38:25   --> charge.succeeded [evt_3MS6kULWXnryLuL40wY13wxH]
2023-01-19 23:38:25   --> payment_intent.created [evt_3MS6kULWXnryLuL40OnhOwVV]
2023-01-19 23:38:26  <--  [200] POST http://localhost:8000/webhook [evt_3MS6kULWXnryLuL40Cvh61KG]
2023-01-19 23:38:26  <--  [200] POST http://localhost:8000/webhook [evt_3MS6kULWXnryLuL40wY13wxH]
2023-01-19 23:38:26  <--  [200] POST http://localhost:8000/webhook [evt_1MS6kWLWXnryLuL4MB6J7GpJ]
2023-01-19 23:38:26  <--  [200] POST http://localhost:8000/webhook [evt_3MS6kULWXnryLuL40OnhOwVV]
```



Session id koji smo generisali. To je id na osnovu koga selektujemo porudžbinu u bazi podataka i menjamo joj status u ,,paid“.

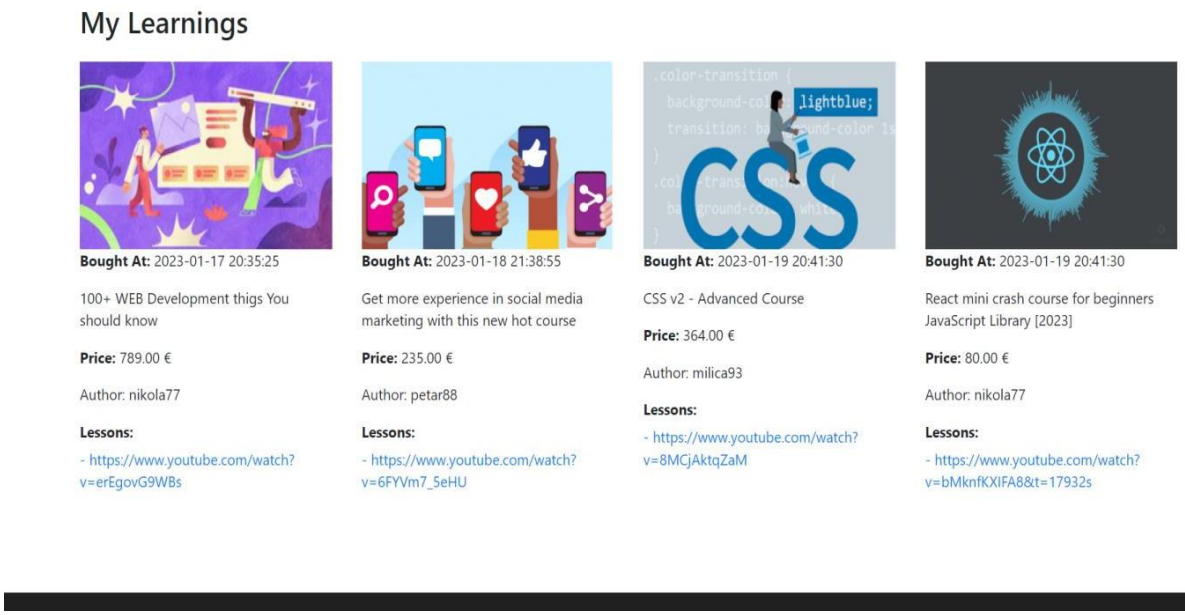

```
{
  "id": "evt_1MS6kWLWXnryLuL4MB6J7GpJ",
  "object": "event",
  "api_version": "2022-11-15",
  "created": 1674167904,
  "data": {
    "object": {
      "id": "cs_test_a1E2JKWNaRGxQ6Ekszr635BDtMBOjAfXOWkUWg1DEsHy1yNmTW785f47GT",
      "object": "checkout.session",
```


Ruta za webhook mora da bude izvan ruta koje su u grupi za autorizaciju, u suprotnom će se prikazivati statusni kod 302.

Stripe flow:

1. Browser generiše sesiju sa e-kursevima na server
2. Server generiše sesiju sa e-kursevima na Stripe
3. Server šalje objekat sesije na Stripe
4. Server pravi ,,unpaid“ porudžbinu i povezuje session id
5. Server vraća session id Browser-u ili server vrši redirekciju na session URL
6. Plaćanjem se vrši redirekcija na success ili cancel stranu
7. Browser šalje session id serveru
8. Server validira sesiju i šalje je Stripe-u
9. Stripe šalje objekat sesije serveru
10. Server proverava porudžbinu sa primljenim session id-em
11. Server označava status porudžbine na ,,paid“
12. Browser prikazuje success stranu
13. Stripe šalje webhook na server
14. Server proverava porudžbinu sa primljenim session id-em (10.)
15. Server označava status porudžbine na ,,paid“ (11.)

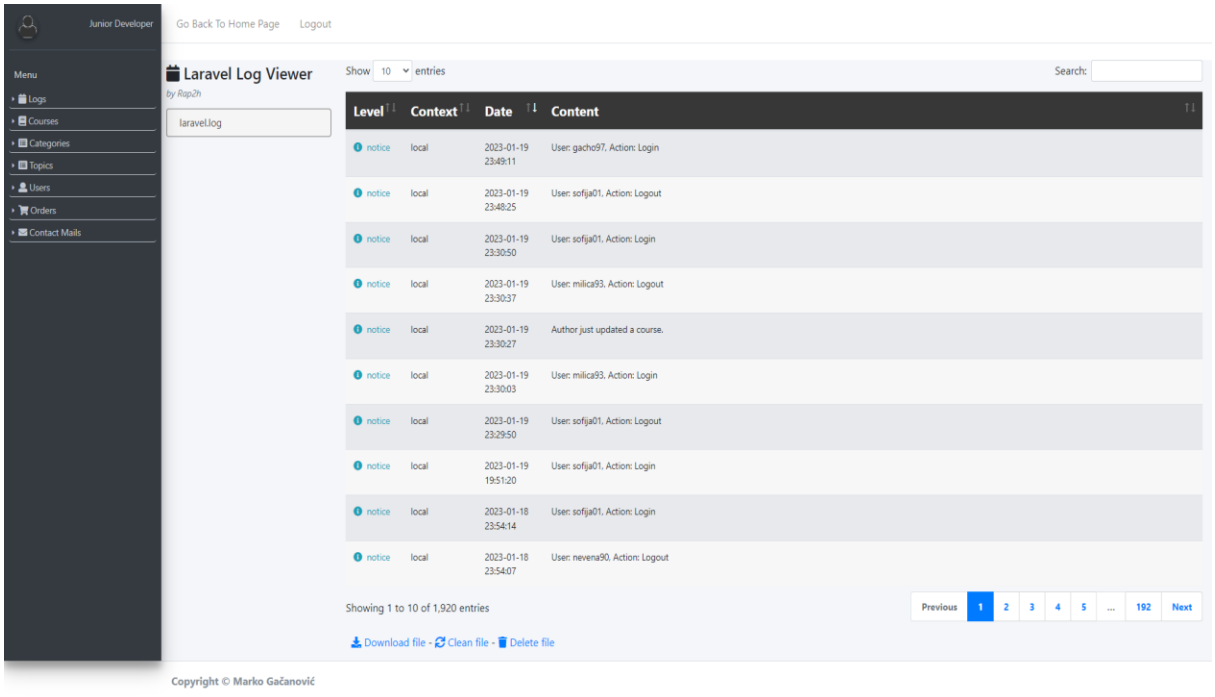
3.1.12 Learnings



Slika 19. Learnings strana

Na stranici kupljenih e-kurseva (slika 19) postoji prikaz svih e-kurseva koje je određeni korisnik kupio. Za svaki e-kurs je prikazan datum poručivanja, opis, cena, naziv autora i sve lekcije koje e-kurs sadrži. E-kurs može da ima više lekcija.

3.1.13 Admin Logs



Slika 20. Strana Admin logs

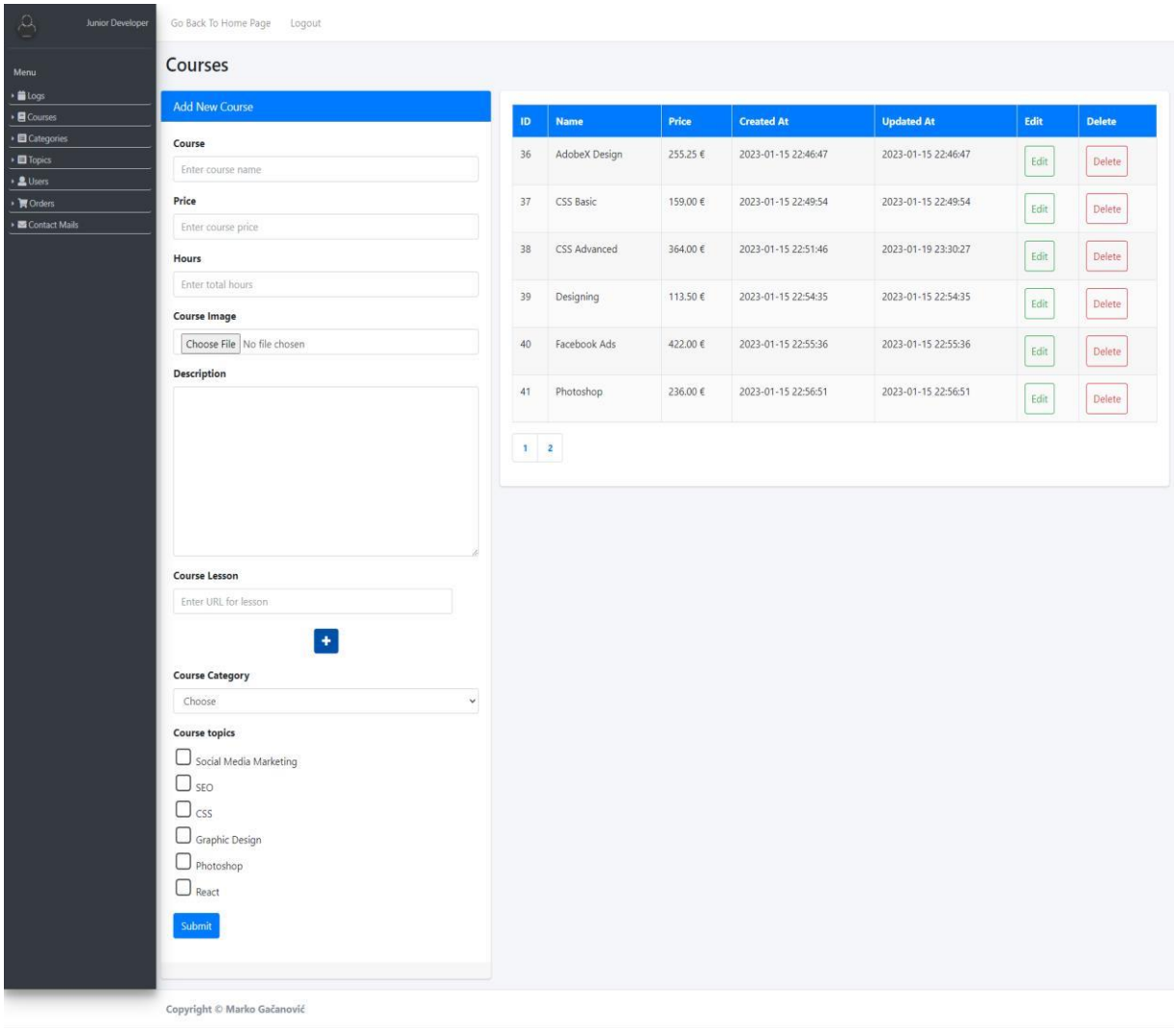
Ovo je početna stranica administratora (slika 20) kada se uloguje u sistem. Nalazi se tabela sa svim ključnim aktivnostima koje se dešavaju na aplikaciji.

Moguće je filtriranje po datumu, tipu loga i sadržaju.

Omogućena je paginacija i pretraga logova, kao i brisanje svih logova.

Svi logovi su smešteni u fajl ,,laravel.log“

3.1.14 Admin Courses Create



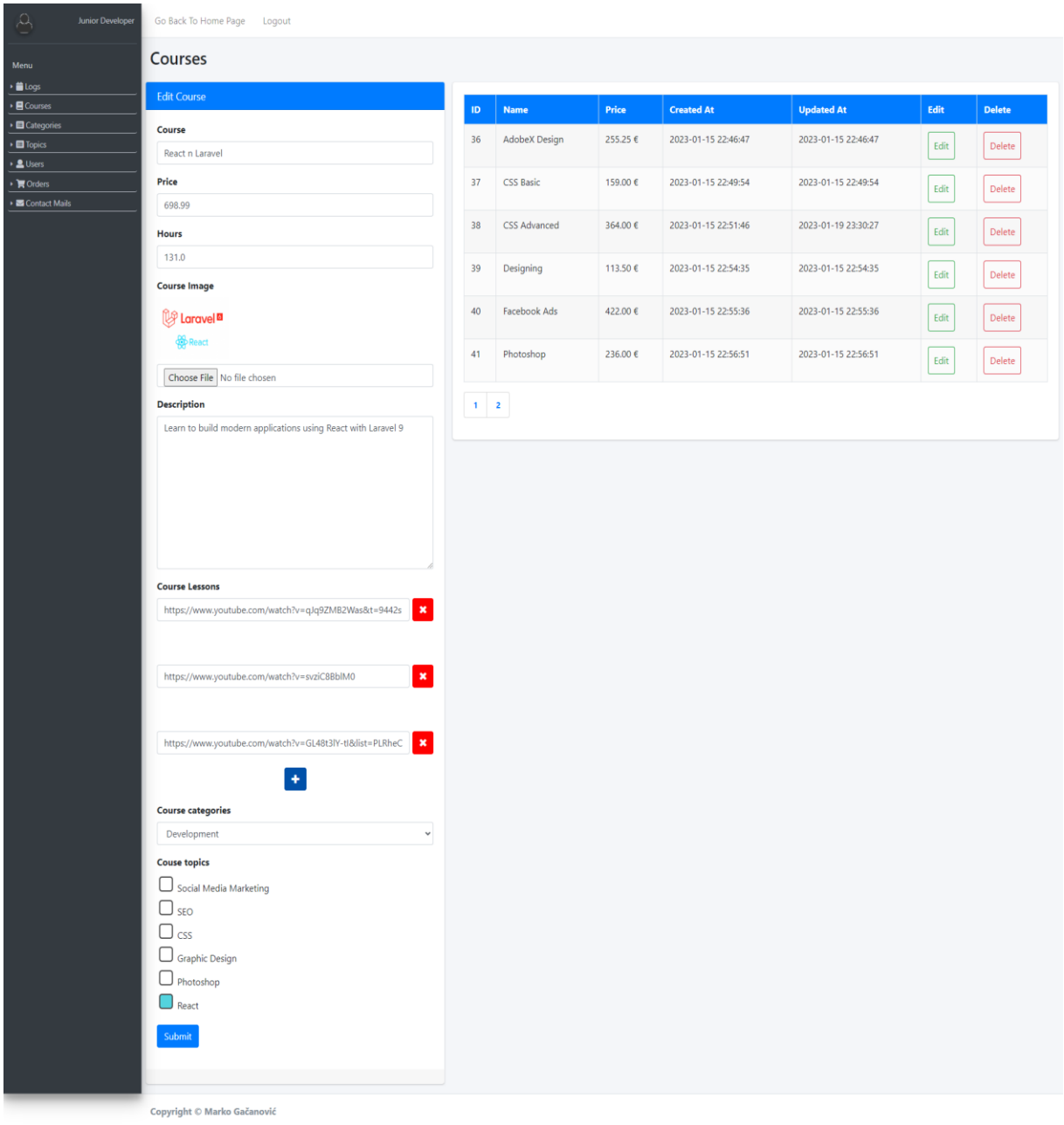
Slika 21. Admin courses create strana

Stranica namenjena za unos i brisanje svih e-kurseva (slika 21) na aplikaciji. Ovoj strani ima pristup samo administrator.

Prikazuju se svi e-kursevi dinamički putem AJAX-a, uz paginaciju.

Administrator unosi naziv e-kursa, cenu, broj sati, sliku, opis, lekcije kojih može biti više od jedne, kategoriju, i teme kojih takođe može biti više.

3.1.15 Admin Courses {id} Edit

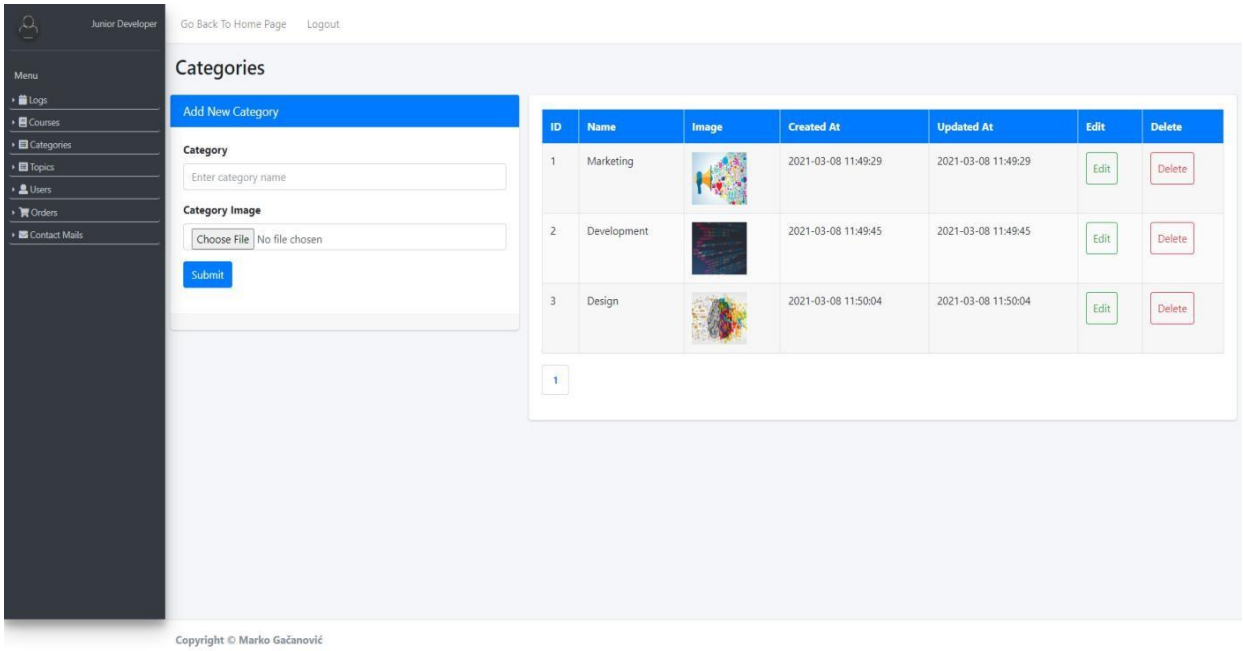


Slika 22. Admin courses edit strana

Strana za izmenu svih e-kurseva na admin layout-u (slika 22).

Kada se klikne na dugme ,,Edit“ otvara se posebna stranica koja sadrži jednu formu koja je identična kao i forma za unos e-kurseva.

3.1.16 Admin Categories Create



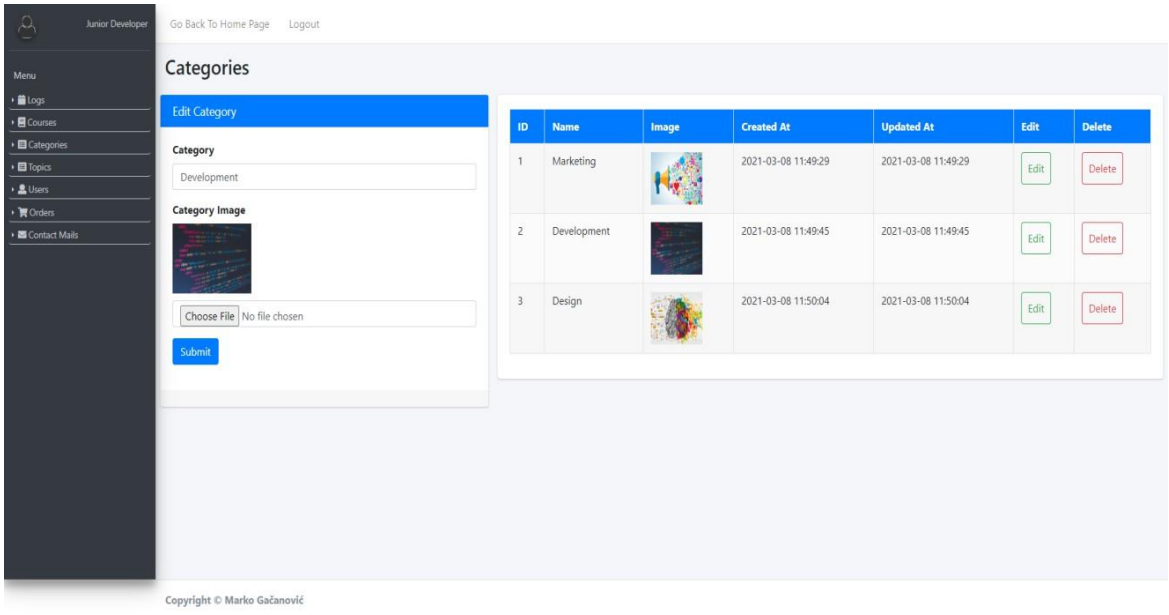
Slika 23. Admin categories strana

Stranica namenjena za brisanje i unos svih kategorija (slika 23) na aplikaciji. Ovoj strani ima pristup samo administrator.

Prikazuju se sve kategorije dinamički putem AJAX-a, uz paginaciju.

Administrator unosi naziv i sliku kategorije.

3.1.17 Admin Categories {id} Edit

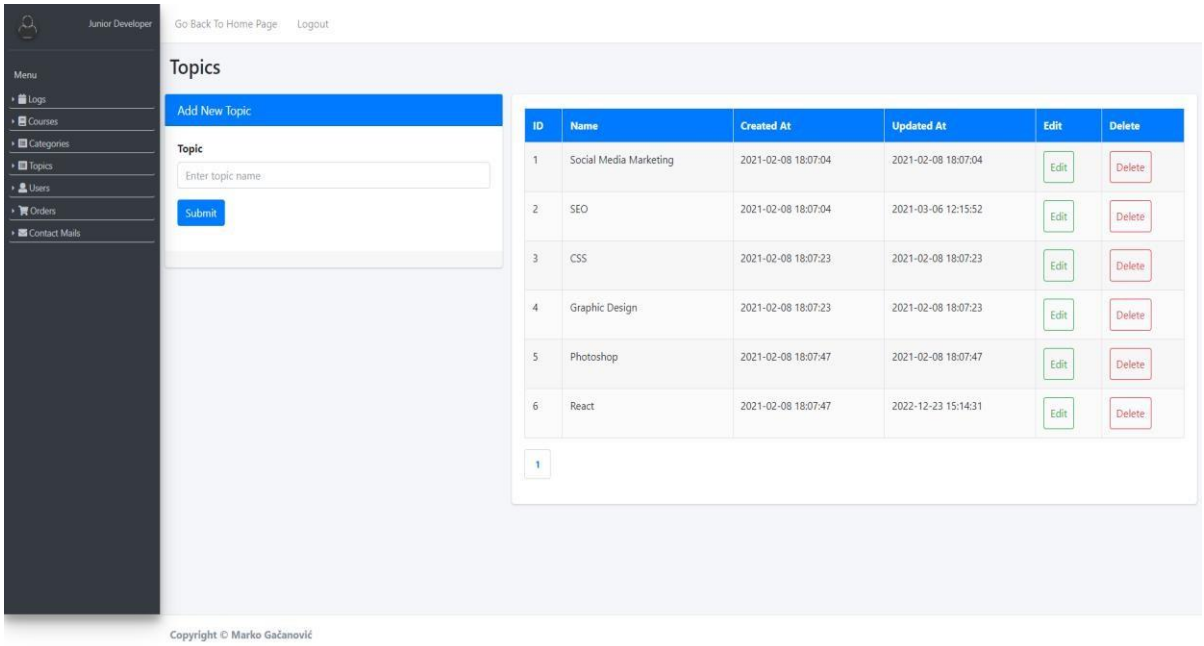


Slika 24. Admin categories edit strana

Strana za izmenu kategorija (slika 24).

Kada se klikne na dugme ,,Edit“ otvara se posebna stranica koja sadrži jednu formu koja je identična kao i forma za unos kategorija.

3.1.18 Admin Topics Create



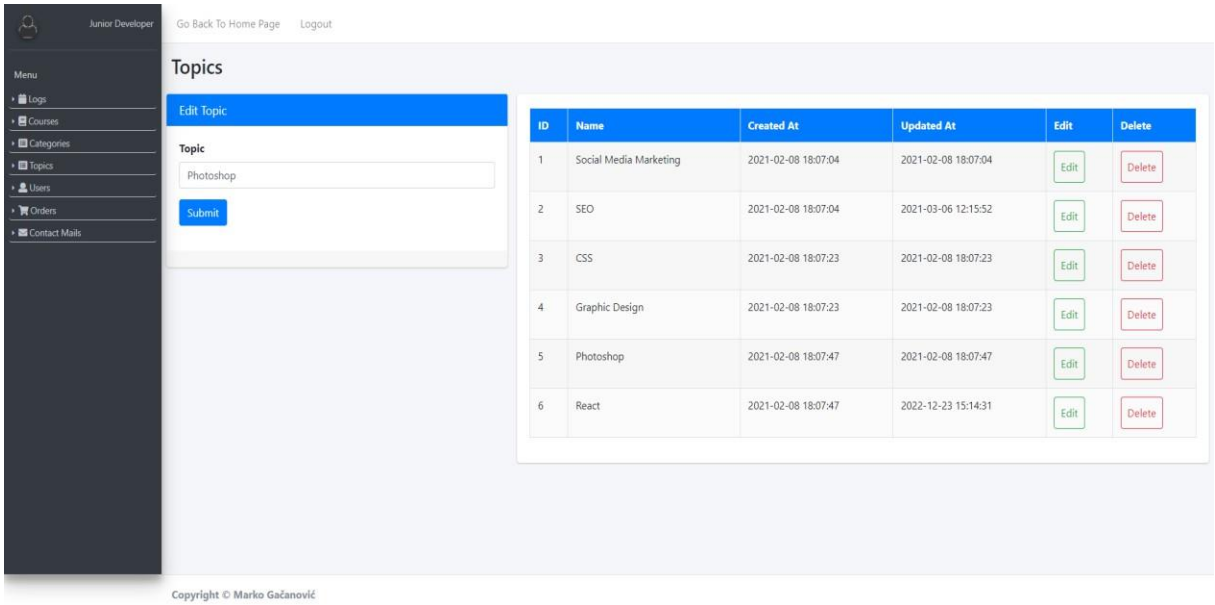
Slika 25. Admin topics create strana

Stranica namenjena za brisanje i unos svih tema (slika 25) na aplikaciji. Ovoj strani ima pristup samo administrator.

Prikazuju se sve teme dinamički putem AJAX-a, uz paginaciju.

Administrator unosi samo naziv teme. Jednoj temi može da pripada više e-kurseva.

3.1.19 Admin Topics {id} Edit

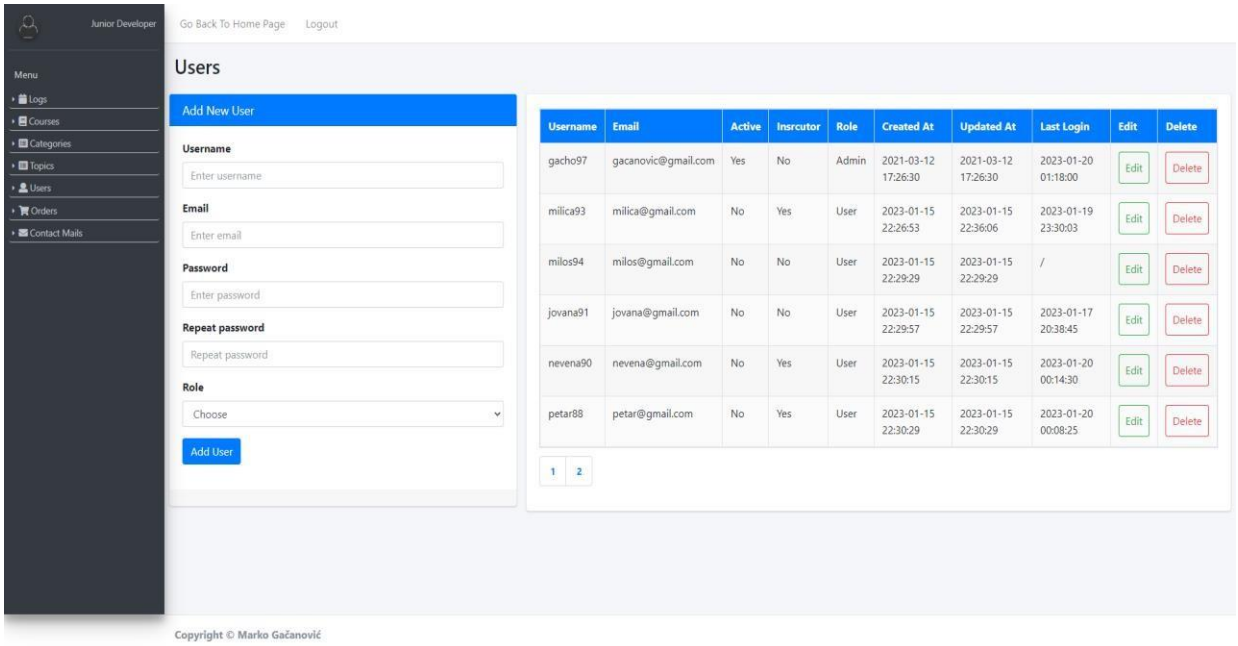


Slika 26. Admin topics edit strana

Strana za izmenu tema (slika 26) u aplikaciji.

Kada se klikne na dugme ,,Edit“ otvara se posebna stranica koja sadrži jednu formu koja je identična kao i forma za unos tema.

3.1.20 Admin Users Create



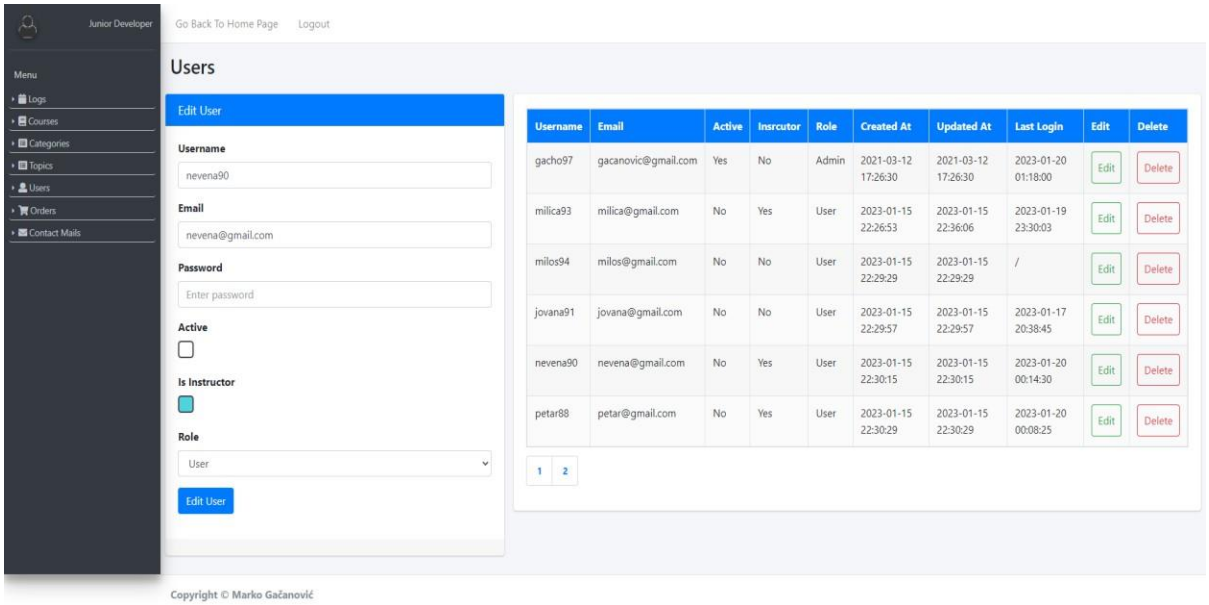
Slika 27. Admin users create strana

Stranica namenjena za brisanje i unos svih korisnika (slika 27) na aplikaciji. Ovoj strani ima pristup samo administrator.

Prikazuju se svi korisnici dinamički putem AJAX-a, uz paginaciju.

Administrator unosi username korisnika, email, lozinku i ulogu.

3.1.21 Admin Users {id} Edit



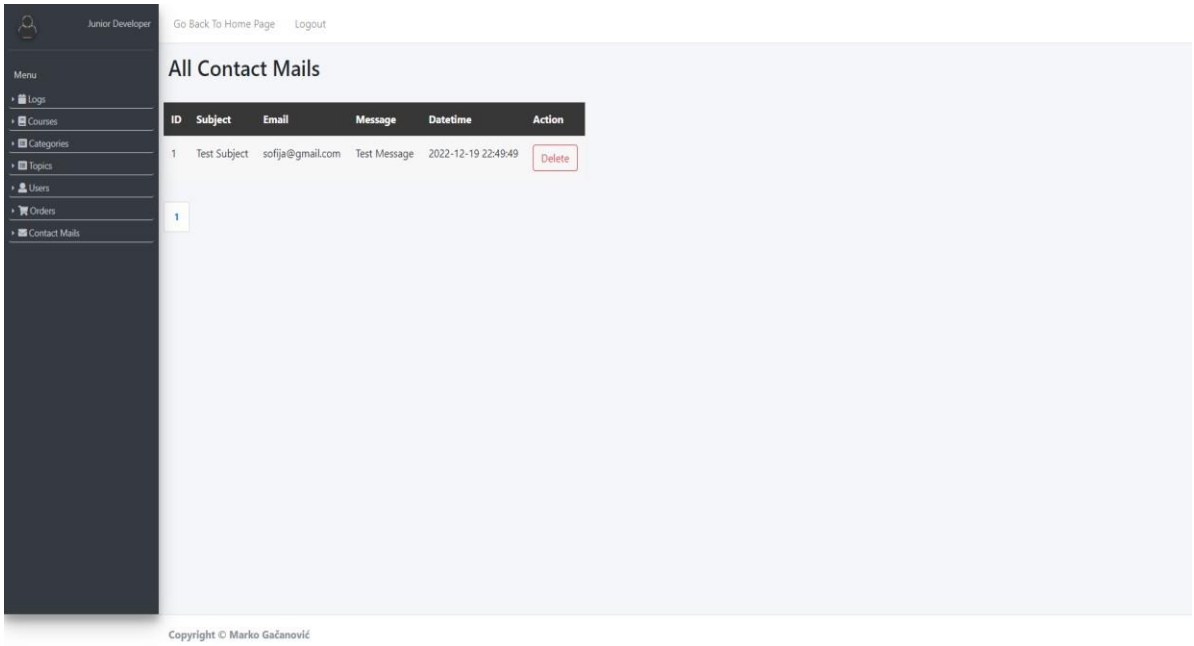
Slika 28. Admin users edit strana

Strana za izmenu svih korisnika (slika 28) iz admin layout-a.

Kada se klikne na dugme ,,Edit“ otvara se posebna stranica koja sadrži jednu formu za izmenu korisnika.

Administrator može da ukine privilegiju autorstva korisniku.

3.1.22 Admin Contact



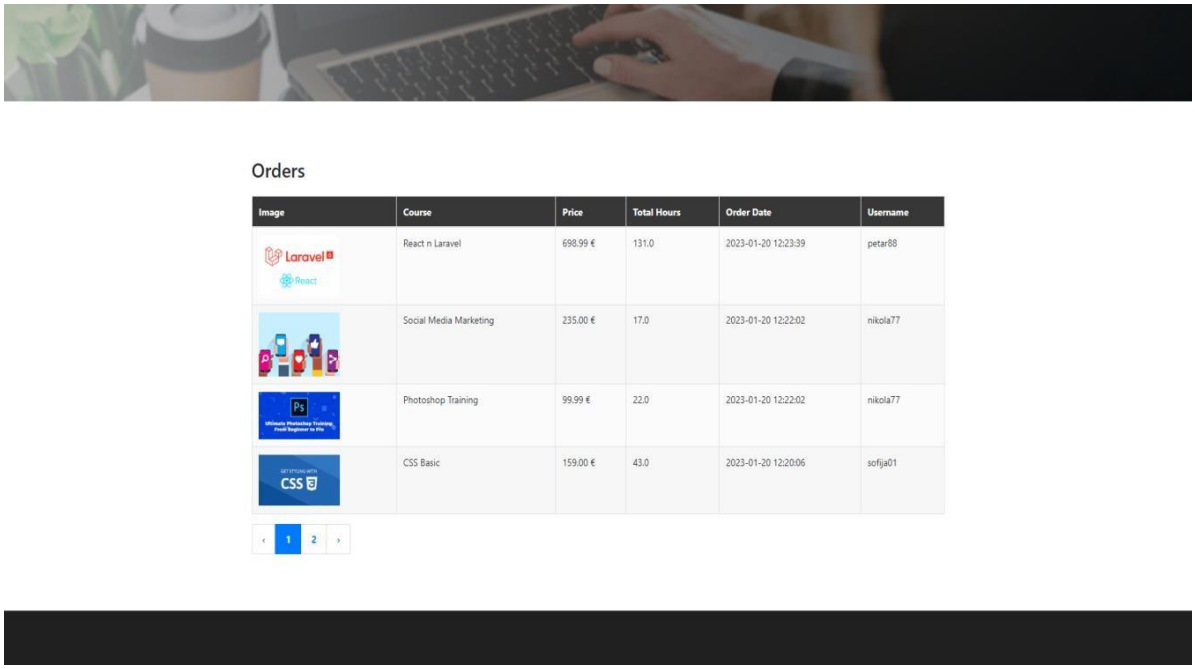
Slika 29. Admin contact strana

Stranica namenjena za prikaz i brisanje svih e-mailova (slika 29) koje je primio administrator.

Ovoj strani ima pristup samo administrator.

Prikaz je dinamički preko AJAX-a, uz paginaciju.

3.1.23 Admin Orders



Slika 30. Admin orders strana

Stranica na kojoj je pregled svih obavljenih porudžbina (slika 30), uz omogućenu serversku paginaciju. Stranici ima pritup samo administrator.

4 Kodovi

4.1 Models

4.1.1 Category.php

```php
<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Storage;

class Category extends Model
{
    public function getFeaturedCategories($limit)
    {
        return DB::table('category AS cat')
            ->join('course AS c', 'c.id_category', '=', 'cat.id_category')
            ->selectRaw('cat.id_category, cat.category_name, cat.category_image,
MIN(c.price) AS min_price')
            ->limit($limit)
            ->groupBy('c.price', 'cat.id_category', 'cat.category_name',
'cat.category_image')
            ->get();
    }

    public function getAllCateogries()
    {
        return DB::table('category')
            ->get();
    }

    public function getCategoriesForAdmin()
    {
        return DB::table('category')
            ->paginate(6);
    }

    public function addCategory($category, $image)
    {
        DB::table('category')
            ->insert([
                'category_name' => $category,
                'category_image' => $image,
                'created_at' => date('Y-m-d H-i-s', time()),
                'updated_at' => date('Y-m-d H-i-s', time())
            ]);
    }

    public function getSingleCategory($id)
    {
        return DB::table('category')
            ->where('id_category', $id)
```

```
            ->first();
    }

    public function updateCategoryWithoutImage($id, $name, $updatedAt)
    {
        DB::table('category')
            ->where('id_category', $id)
            ->update([
                'id_category' => $id,
                'category_name' => $name,
                'updated_at' => $updatedAt
            ]);
    }

    public function updateCategoryWithImage($id, $name, $image, $updatedAt)
    {
        DB::table('category')
            ->where('id_category', $id)
            ->update([
                'id_category' => $id,
                'category_name' => $name,
                'category_image' => $image,
                'updated_at' => $updatedAt
            ]);
    }

    public function deleteCategory($id)
    {
        DB::table('category')
            ->where('id_category', $id)
            ->delete();
    }
}
```

4.1.2 ContactMail.php

```
<?php

namespace App\Models;

use DB;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class ContactMail extends Model
{
    use HasFactory;

    public function getMailsForAdmin()
    {
        return DB::table('contact_mail')
            ->paginate(6);
```

```
    }

    public function deleteContactMail($id)
    {
        DB::table('contact_mail')
            ->where('id_contact_mail', $id)
            ->delete();
    }
}
```

4.1.3 Course.php

```
<?php

namespace App\Models;

use App\Http\Requests\AddCourseRequest;
use App\Http\Requests\UpdateCourseRequest;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Course extends Model
{
    public function insertCourse($courseName, $description, $price, $totalHours,
$author, $image_small, $image_big, $id_category)
    {
        return DB::table('course')
            ->insertGetId([
                'course_name' => $courseName,
                'description' => $description,
                'price' => $price,
                'total_hours' => $totalHours,
                'author' => $author,
                'image_small' => $image_small,
                'image_big' => $image_big,
                'id_category' => $id_category,
                'created_at' => date('Y-m-d H-i-s', time()),
                'updated_at' => date('Y-m-d H-i-s', time())
            ]);
    }

    public function getAllCourses()
    {
        return DB::table('course AS c')
            ->join('category AS cat', 'c.id_category', '=', 'cat.id_category')
            ->get();
    }
```

```
    public function getCoursesInCart($idsCourses)
    {
        return DB::table('course AS cou')
            ->join('category AS cat', 'cou.id_category', '=', 'cat.id_category')
            ->whereIn('cou.id_course', $idsCourses)
            ->get();
    }

    public function getCoursesForAdmin()
    {
        return DB::table('course')->paginate(6);
    }

    public function getCoursesForInstructor($currentAuthor)
    {
        return DB::table('course')
            ->where(['course.author' => $currentAuthor])
            ->paginate(6);
    }

    public function filter($search, $categories, $topic, $sort, $showing,
$currentAuthor)
    {
        $query = DB::table('course');

        if ($currentAuthor) {
            $query = $query->where('author', '!=', $currentAuthor);
        }

        if ($search) {
            $query = $query->where('course_name', 'like', '%' . $search . '%')
                ->orWhere('author', 'like', '%' . $search . '%');
        }
        $query = $query->join('category', 'course.id_category', '=',
'category.id_category');

        if ($categories) {
            $query = $query->whereIn('category.id_category', $categories);
        }

        if ($topic) {
            $query = $query->join('course_topic', 'course.id_course', '=',
'course_topic.id_course')
                ->join('topic', 'course_topic.id_topic', '=', 'topic.id_topic')
                ->whereIn('topic.id_topic', $topic);
        }

        switch ($sort) {
            case 'date':
                $query = $query->latest('course.created_at');
                break;
```

```
            case 'priceAsc':
                $query = $query->orderBy('course.price', 'ASC');
                break;
            case 'priceDesc':
                $query = $query->orderBy('course.price', 'DESC');
                break;
        }

        $query = $query->paginate($showing);

        return $query;
    }

    public function getSingleCourseTopics($id)
    {
        return DB::table('course')
            ->join('course_topic', 'course.id_course', '=', 'course_topic.id_course')
            ->join('topic', 'course_topic.id_topic', '=', 'topic.id_topic')
            ->select('topic.*')
            ->where('course.id_course', '=', $id)
            ->get();
    }

    public function getSingleCourse($idCourse)
    {
        $courseSingle = DB::table('course')
            ->join('category', 'category.id_category', '=', 'course.id_category')
            ->where('course.id_course', $idCourse)
            ->first();

         $courseSingle->topics = $this->getSingleCourseTopics($idCourse);

        return $courseSingle;
    }

    public function deleteCourse($id)
    {
        DB::table('course')
            ->where(['id_course' => $id])
            ->delete();
    }

    public function updateCourseWithoutImage($id, $name, $description, $price, $hours,
$category, $updatedAt)
    {
        return DB::table('course')
            ->where('id_course', $id)
            ->update([
                'course_name' => $name,
                'description' => $description,
                'price' => $price,
```

```
                'total_hours' => $hours,
                'id_category' => $category,
                'updated_at' => $updatedAt
            ]);
    }

    public function updateCourseWithImage($id, $name, $description, $price, $hours,
$smallImage, $bigImage, $category, $updatedAt)
    {
        return DB::table('course')
            ->where('id_course', $id)
            ->update([
                'course_name' => $name,
                'description' => $description,
                'price' => $price,
                'total_hours' => $hours,
                'image_small' => $smallImage,
                'image_big' => $bigImage,
                'id_category' => $category,
                'updated_at' => $updatedAt
            ]);
    }
}
```

4.1.4 CourseOrder.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;


class CourseOrder extends Model
{
    use HasFactory;

    public function insertCourseOrder($course, $order)
    {
        DB::table('courses_orders')
            ->insert([
                'id_course' => $course,
                'id_course_order' => $order
            ]);
    }

    public function deleteCourseFromOrder($id)
```

```
    {
        DB::table('courses_orders')
            ->where(['id_course' => $id])
            ->delete();
    }
}
```

4.1.5 CourseTopic.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class CourseTopic extends Model
{
    public function insertCourseTopic($course, $topic)
    {
        DB::table('course_topic')
            ->insert([
                'id_course' => $course,
                'id_topic' => $topic
            ]);
    }

    public function deleteCourse($id)
    {
        DB::table('course_topic')
            ->where(['id_course' => $id])
            ->delete();
    }
}
```

4.1.6 Instructor.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Instructor extends Model
```

```
{
    use HasFactory;

    public function getPollAnswers()
    {
        return DB::table('answer')
            ->get();
    }

    public function voting($answer, $idUser)
    {
        DB::table('voting')
            ->insert([
                'id_answer' => $answer,
                'id_user' => $idUser
            ]);
    }

    public function ifVoted($idUser)
    {
        return DB::table('voting')
            ->where('id_user', '=', $idUser)
            ->exists();
    }

    public function updateToInstructor($idUser)
    {
        return DB::table('user')
            ->where('id_user', $idUser)
            ->update([
                'is_instructor' => 1
            ]);
    }
}
```

4.1.7 Lesson.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Lesson extends Model
{
    use HasFactory;
```

```
    public function insertCourseLesson($lessonURL, $course)
    {
        DB::table('lesson')
            ->insert([
                'lesson' => $lessonURL,
                'id_course' => $course
            ]);
    }

    public function getAllLessons()
    {
        return DB::table('lesson')
            ->get();
    }

    public function getAllLessonsForCourse($id_course)
    {
        return DB::table('lesson AS l')
            ->join('course AS c', 'c.id_course', '=', 'l.id_course')
            ->where('l.id_course', '=', $id_course)
            ->get();
    }

    public function getAllLessonsForCourses($id_courses)
    {
        return DB::table('lesson AS l')
            ->join('course AS c', 'c.id_course', '=', 'l.id_course')
            ->whereIn('c.id_course', $id_courses)
            ->get();
    }

    public function deleteLesson($id_lesson)
    {
        DB::table('lesson')
            ->where('id_lesson', $id_lesson)
            ->delete();
    }

    public function deleteCourseLesson($id_course)
    {
        DB::table('lesson')
            ->where('id_course', $id_course)
            ->delete();
    }
}
```

4.1.8 Order.php

```
<?php
```

```php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Order extends Model
{
    use HasFactory;

    public function listLearningsForUser($idUser)
    {
        return DB::table('orders AS o')
            ->select('o.created_at AS bought_at', 'u.*', 'co.*', 'c.*')
            ->join('user AS u', 'u.id_user', '=', 'o.id_user')
            ->join('courses_orders AS co', 'co.id_course_order', '=',
'o.id_course_order')
            ->join('course AS c', 'c.id_course', '=', 'co.id_course')
            ->where('u.id_user', $idUser)
            ->get();
    }

    public function listOrdersAdmin()
    {
        return DB::table('orders AS o')
            ->select('c.*', 'u.*', 'o.created_at AS orderd_at')
            ->join('user AS u', 'u.id_user', '=', 'o.id_user')
            ->join('courses_orders AS co', 'co.id_course_order', '=',
'o.id_course_order')
            ->join('course AS c', 'c.id_course', '=', 'co.id_course')
            ->orderBy('o.created_at', 'DESC')
            ->paginate(4);
    }
}
```

4.1.9 Role.php

```php
<?php

namespace App\Models;

use DB;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Role extends Model
{
    use HasFactory;
```

```
    public function getAllRoles()
    {
        return DB::table('role')
            ->get();
    }
}
```

4.1.10 Topic.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Topic extends Model
{
    public function getAllTopics()
    {
        return DB::table('topic')
            ->get();
    }

    public function addTopic($topic)
    {
        DB::table('topic')
            ->insert([
                'topic_name' => $topic,
                'created_at' => date('Y-m-d H-i-s', time()),
                'updated_at' => date('Y-m-d H-i-s', time())
            ]);
    }

    public function getAdminTopics()
    {
        return DB::table('topic')
            ->paginate(6);
    }

    public function getSingleTopic($id)
    {
        return DB::table('topic')
            ->where('id_topic', $id)
            ->first();
    }
```

```
    public function updateTopic($id, $topic)
    {
        DB::table('topic')
            ->where('id_topic', $id)
            ->update([
                'topic_name' => $topic,
                'updated_at' => date('Y-m-d H-i-s', time())
            ]);
    }

    public function deleteTopic($id)
    {
        DB::table('topic')
            ->where('id_topic', $id)
            ->delete();
    }
}
```

4.1.11 User.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class User
{
    public function insert($obj)
    {
        DB::table('user')
            ->insert($obj);
    }

    public function getUsernameAndPassword($username, $password)
    {
        return DB::table('user')
            ->select('id_user', 'username', 'email', 'id_role')
            ->where([
                'username' => $username,
                'password' => md5($password)
            ])
            ->first();
    }

    public function updateActivityLogin($id_user, $active, $time)
    {
        return DB::table('user')
            ->where('id_user', $id_user)
```

```
            ->update([
                'active' => $active,
                'last_login' => $time
            ]);
    }

    public function updateActivityLogout($id_user, $active, $time)
    {
        return DB::table('user')
            ->where('id_user', $id_user)
            ->update([
                'active' => $active,
                'last_logout' => $time
            ]);
    }

    public function storeContactEmail($obj)
    {
        return DB::table('contact_mail')
            ->insert($obj);
    }

    public function getUsersForAdmin()
    {
        return DB::table('user')
            ->join('role', 'user.id_role', '=', 'role.id_role')
            ->paginate(6);
    }

    public function getSingleUser($id)
    {
        return DB::table('user')
            ->where('id_user', $id)
            ->first();
    }

    public function deleteUser($id)
    {
        DB::table('user')
            ->where('id_user', $id)
            ->delete();
    }

    public function deleteUserVoting($id)
    {
        DB::table('voting')
            ->where('id_user', $id)
            ->delete();
    }

    public function updateUser($obj, $id)
```

```
    {
        DB::table('user')
            ->where('id_user', $id)
            ->update($obj);
    }
}
```

4.1.12 Wish.php

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Wish extends Model
{
    use HasFactory;

    public function addWish($idUser, $idCourse)
    {
        DB::table('wish')
            ->insert([
                'id_user' => $idUser,
                'id_course' => $idCourse,
                'created_at' => date('Y-m-d H-i-s', time())
            ]);
    }

    public function checkIfAlereadyExists($idUser, $idCourse)
    {
        return DB::table('wish')
            ->where(['id_user' => $idUser, 'id_course' => $idCourse])
            ->exists();
    }

    public function numberOfWishes($idUser)
    {
        return DB::table('wish')
            ->where(['id_user' => $idUser])
            ->count();
    }

    public function getAllWishesForOneUser($idUser)
    {
        return DB::table('course')
            ->join('wish', 'course.id_course', '=', 'wish.id_course')
```

```
            ->join('user', 'wish.id_user', '=', 'user.id_user')
            ->where(['user.id_user' => $idUser])
            ->get();
    }

    public function deleteWish($idWish)
    {
        DB::table('wish')->where(['id_wish' => $idWish])->delete();
    }

    public function deleteCoursFromWish($id)
    {
        DB::table('wish')->where(['id_course' => $id])->delete();
    }
}
```

4.2 Controllers

4.2.1 AuthController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests\LoginRequest;
use App\Http\Requests\RegistrationRequest;
use App\Models\User;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use Exception;
use Illuminate\Http\Request;
use PDOException;

class AuthController extends FrontController
{
    private $model;

    public function __construct()
    {
        $this->model = new User();
    }

    public function doRegister(RegistrationRequest $request)
    {
        $created_at = date('Y-m-d H-i-s', time());
        $obj = [
            'username' => $request->input('username'),
            'email'  => $request->input('email'),
            'password' => md5($request->input('password')),
```

```
            'active' => 0,
            'is_instructor' => 0,
            'id_role' => 2,
            'created_at' => $created_at,
            'updated_at' => $created_at
        ];

        try {
            $this->model->insert($obj);

            Logs::loggingSuccess('User: ' . $obj['username'] . ', Action: Register');
            return response([], 201);
        } catch (Exception $ex) {
            Logs::logging($ex->getMessage(), '[AuthController::class, "doRegister"]');
            return Helper::returnGenericErrorAjax();
        }
    }

    public function doLogin(Request $request, LoginRequest $loginRequest)
    {
        $username = $request->input('username');
        $password = $request->input('password');

        try {
            $user = $this->model->getUsernameAndPassword($username, $password);

            if ($user) {
                $request->session()->put('user', $user);

                $time = Helper::getTime();
                $this->model->updateActivityLogin(session()->get('user')->id_user, 1,
$time);

                Logs::loggingSuccess('User: ' . $user->username . ', Action: Login');

                if (session()->get('user')->id_user == 1) {
                    return redirect()->route('logs');
                }

                return redirect('/');
            } else {
                return redirect()->route('login')->with('error', 'Wrong password or
account is inactive.');
            }
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[AuthController::class, "doLogin"]');
            return Helper::returnGenericError();
        }
    }

    public function doLogout(Request $request)
```

```
    {
        if ($request->session()->has('user')) {
            $time = Helper::getTime();
            try {
                $this->model->updateActivityLogout(session()->get('user')->id_user,  0,
$time);

                Logs::loggingSuccess('User: ' . session('user')->username . ', Action:
Logout');

                $request->session()->forget('user');

                return redirect('/');
            } catch (PDOException $ex) {
                Logs::logging($ex->getMessage(), '[AuthController::class,
"doLogout"]');
                return Helper::returnGenericError();
            }
        }
    }
}
```

4.2.2 CartController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Services\Logs;
use App\Models\Course;
use PDOException;

class CartController extends FrontController
{
    private $courseModel;

    public function __construct()
    {
        $this->courseModel = new Course();
    }

    public function getCoursesForCart()
    {
        try {
            $courses = $this->courseModel->getAllCourses();
            return response()->json($courses);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[CartController::class,
"getCoursesForCart"]');
```

```
        }
    }
}
```

4.2.3 CheckoutController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\CourseOrder;
use Illuminate\Http\Request;
use App\Models\Course;
use App\Models\Order;
use Symfony\Component\HttpKernel\Exception\NotFoundHttpException;


class CheckoutController extends FrontController
{
    private $courseModel;
    private $courseOrderModel;

    public function __construct()
    {
        $this->courseModel = new Course();
        $this->courseOrderModel = new CourseOrder();
    }

    public function checkout()
    {
        $stripe = new
\Stripe\StripeClient('sk_test_51M6FPJLWXnryLuL450g74dl6RCESGZXY7fFHSnsjMWoHSlJ8GWhJdeHP
uYcjuO7R8DDKzSJicMd71bkROT7EBvfo003W2WfyEx');

        $idUser = session()->get('user')->id_user;

        $idsForExplode = $_POST['cartItems'];
        $cartItems = explode(',', $idsForExplode);
        $courses = $this->courseModel->getCoursesInCart($cartItems);

        $totalPrice = 0;

        $lineItems = [];
        foreach ($courses as $course) {
            $totalPrice += $course->price;

            $lineItems[] = [
                'price_data' => [
                    'currency' => 'usd',
```

```
                    'product_data' => [
                        'name' => $course->course_name
                    ],
                    'unit_amount' => $course->price * 100 // usd cents
                ],
                'quantity' => 1
            ];
        }

        $courses_ids = [];
        foreach ($courses as $course) {
            $courses_ids[] = $course->id_course;
        }
        $imploded_courses = implode(',', $courses_ids);

        $checkout_session = $stripe->checkout->sessions->create([
            'line_items' => $lineItems,
            'mode' => 'payment',
            'success_url' => route('checkout.success', [],
true)."?id_session={CHECKOUT_SESSION_ID}&courses=$imploded_courses",
            'cancel_url' => route('checkout.cancel', [], true),
        ]);

        $order = new Order();
        $order->status = 'unpaid';
        $order->total_price = $totalPrice; // $totalPrice
        $order->id_session = $checkout_session->id;
        $order->id_user = $idUser;
        $order->save();

        header("HTTP/1.1 303 See Other");
        header("Location: " . $checkout_session->url);

        return redirect($checkout_session->url);
    }

    public function success(Request $request)
    {
        $stripe = new
\Stripe\StripeClient('sk_test_51M6FPJLWXnryLuL450g74dl6RCESGZXY7fFHSnsjMWoHSlJ8GWhJdeHP
uYcjuO7R8DDKzSJicMd71bkROT7EBvfo003W2WfyEx');
        $sessionId = $request->get('id_session');

        try {
            $session = $stripe->checkout->sessions->retrieve($sessionId);

            if (!$session) {
                throw new NotFoundHttpException;
            }

            $order = Order::where('id_session', $session->id)->first();
```

```
            if (!$order) {
                throw new NotFoundHttpException();
            }

            if ($order->status === "unpaid") {
                $courses = $_GET['courses'];

                $exploded_courses = explode(',', $courses);

                $order_last_id = $order->id_course_order;

                foreach ($exploded_courses as $course) {
                    $courses_ids = $course;
                    $this->courseOrderModel->insertCourseOrder($courses_ids,
$order_last_id);
                }

                $order = Order::where('id_session', $session->id)-
>update(['status'=>'paid']);
            }

            return view('pages.user.checkout-success');
        } catch (Exception $ex) {
            throw new NotFoundHttpException;
        }
    }

    public function cancel()
    {
        return view('pages.user.checkout-cancel');
    }

    public function webhook()
    {
        // This is your Stripe CLI webhook secret for testing your endpoint locally.
        $endpoint_secret =
'whsec_0fb25c7c9f0846d453948ad3651a63cb6b12f95076edae2e553b5b1ec3e56cd3';

        $payload = @file_get_contents('php://input');
        $sig_header = $_SERVER['HTTP_STRIPE_SIGNATURE'];
        $event = null;

        try {
            $event = \Stripe\Webhook::constructEvent($payload, $sig_header,
$endpoint_secret);
        } catch (\UnexpectedValueException $e) {
            // Invalid payload
            http_response_code(400);
        } catch (\Stripe\Exception\SignatureVerificationException $e) {
            // Invalid signature
```

```
            http_response_code(400);
        }

        // Handle the event
        switch ($event->type) {
            // jedini case mi treba 'checkout.session.completed' i onda radim isto sto
i u success.
            case 'checkout.session.completed':
                $session = $event->data->object;
                $success_url = $session->success_url;

                $order = Order::where('id_session', $session->id)->first();
                if (!$order) {
                    throw new NotFoundHttpException();
                }
                if ($order->status === "unpaid") {
                    $courses = substr($success_url, strrpos($success_url, '=') + 1);
                    $exploded_courses = explode(',', $courses);
                    $order_last_id = $order->id_course_order;

                    foreach ($exploded_courses as $course) {
                        $courses_ids = $course;
                        $this->courseOrderModel->insertCourseOrder($courses_ids,
$order_last_id);
                    }

                    $order = Order::where('id_session', $session->id)-
>update(['status'=>'paid']);
                }
            default:
                echo 'Received unknown event type ' . $event->type;
        }

        http_response_code(200);
    }
}
```

4.2.4 ContactController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests\ContactRequest;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use App\Mail\ContactMail;
use App\Models\User;
use Exception;
```

```php
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Mail;

class ContactController extends FrontController
{
    private $model;

    public function __construct()
    {
        $this->model = new User();
    }

    public function store(ContactRequest $request)
    {
        $obj = [
            'subject' => $request->input('subject'),
            'email' => $request->input('email'),
            'message' => $request->input('message'),
            'date' => date('Y-m-d H-i-s', time())
        ];

        try {
            Mail::to('markogacanovic111@gmail.com')->send(new ContactMail($obj));

            $this->model->storeContactEmail($obj);

            Logs::loggingSuccess('Successfully stored message');
            return response(['success' => 'Your message has been sent successfully!'],
201);
        } catch (Exception $ex) {
            Logs::logging($ex->getMessage(), 'Error.');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.5 FrontController.php

```php
<?php

namespace App\Http\Controllers;

use App\Http\Requests\FilterCoursesRequest;
use App\Models\Instructor;
use App\Models\Lesson;
use App\Models\Order;
use App\Models\Category;
use App\Models\Course;
use App\Models\Topic;
```

```php
use App\Models\User;
use Illuminate\Http\Request;

class FrontController extends Controller
{
    protected $data;
    protected $modelTopics;
    protected $modelCourses;
    protected $modelCategories;
    protected $modelLearnings;
    protected $modelLessons;
    protected $modelUser;
    protected $modelInstructor;

    public function __construct()
    {
        $this->modelTopics = new Topic();
        $this->modelCourses = new Course();
        $this->modelCategories = new Category();
        $this->modelLearnings = new Order();
        $this->modelLessons = new Lesson();
        $this->modelUser = new User();
        $this->modelInstructor = new Instructor();

        $categories = $this->modelCategories->getFeaturedCategories(3);
        $this->data['categories'] = $categories;
    }

    public function homePage()
    {
        return view('pages.user.home', $this->data);
    }

    public function wishesPage()
    {
        return view('pages.user.wishes', $this->data);
    }

    public function contactPage()
    {
        return view('pages.user.contact', $this->data);
    }

    public function authorPage()
    {
        return view('pages.user.author', $this->data);
    }

    public function loginPage()
    {
        return view('pages.user.login', $this->data);
```

```
    }

    public function cartPage()
    {
        return view('pages.user.cart', $this->data);
    }

    public function checkoutPage()
    {
        return view('pages.user.checkout', $this->data);
    }

    public function learningsPage()
    {
        $idUser = session()->get('user')->id_user;

        $learnings = ['myLearnings' => $this->modelLearnings-
>listLearningsForUser($idUser)];
        foreach ($learnings as $learning) {
            $courses = [];
            foreach ($learning as $l) {
                $courses[] = $l->id_course;
            }
        }

        return view('pages.user.learnings', ['myLearnings' => $this->modelLearnings-
>listLearningsForUser($idUser), 'categories' => $this->modelCategories-
>getFeaturedCategories(3), 'lessons' => $this->modelLessons-
>getAllLessonsForCourses($courses)]);
    }

    public function instructorPage()
    {
        $idUser = session()->get('user')->id_user;

        $isInstructor = $this->modelUser->getSingleUser($idUser)->is_instructor;
        if ($isInstructor === 1) {
            return view('pages.admin.courses', ['categories' => $this->modelCategories-
>getAllCateogries(), 'topics' => $this->modelTopics->getAllTopics(), 'lessons' =>
$this->modelLessons->getAllLessons()]);
        } else {
            return view('pages.instructor.instructor', ['user' => $this->modelUser-
>getSingleUser($idUser), 'answers' => $this->modelInstructor->getPollAnswers(),
'categories' => $this->modelCategories->getAllCateogries()]);
        }

    }

    public function ordersPage()
    {
```

```
        return view('pages.user.orders', ['ordersAdmin' => $this->modelLearnings-
>listOrdersAdmin(), 'categories' => $this->modelCategories->getFeaturedCategories(3)]);
    }

    public function coursesPage(FilterCoursesRequest $request)
    {
        $search = $request->input('search');
        $categoriesChb = $request->input('categories') ?? [];
        $sort = $request->input('sort') ?? 'date';
        $topicChb = $request->input('topic') ?? [];
        $showing = $request->input('showing') ?? 6;

        $this->data['search'] = $search;
        $this->data['categoriesChb'] = $categoriesChb;
        $this->data['sort'] = $sort;
        $this->data['topicChb'] = $topicChb;
        $this->data['showing'] = $showing;

        if (session()->has('user')) {
            $currentAuthor = session()->get('user')->username;
            $this->data['courses'] = $this->modelCourses->filter($search,
$categoriesChb, $topicChb, $sort, $showing, $currentAuthor);

            $idUser = session()->get('user')->id_user;
            $this->data['myLearnings'] = $this->modelLearnings-
>listLearningsForUser($idUser);
        } else {
            $currentAuthor = '';
            $this->data['courses'] = $this->modelCourses->filter($search,
$categoriesChb, $topicChb, $sort, $showing, $currentAuthor);
        }

        $this->data['topics'] = $this->modelTopics->getAllTopics();

        return view('pages.user.courses', $this->data);
    }

    public function singleCoursePage($id)
    {
        if (session()->has('user')) {
            $idUser = session()->get('user')->id_user;
            $this->data['myLearnings'] = $this->modelLearnings-
>listLearningsForUser($idUser);
        }

        $course = $this->modelCourses->getSingleCourse($id);
        $this->data['course'] = $course;

        return view('pages.user.singe_course', $this->data);
    }
}
```

4.2.6 InstructorController.php

```php
<?php

namespace App\Http\Controllers;

use App\Http\Requests\UpdateCourseRequest;
use App\Http\Services\Helper;
use App\Http\Services\ImageHelper;
use App\Http\Services\Logs;
use App\Models\Category;
use App\Models\Course;
use App\Models\CourseOrder;
use App\Models\CourseTopic;
use App\Models\Instructor;
use App\Models\Lesson;
use App\Models\Topic;
use App\Models\User;
use App\Models\Wish;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use PDOException;
use App\Http\Requests\AddCourseRequest;

class InstructorController extends Controller
{
    private $data;
    private $instructorModel;
    private $categories;
    private $topics;
    private $lessons;
    private $courses;
    private $courseTopics;
    private $users;
    private $courseOrder;
    private $wishes;

    public function __construct()
    {
        $this->instructorModel = new Instructor();
        $this->categories = new Category();
        $this->topics = new Topic();
        $this->courseTopics = new CourseTopic();
        $this->lessons = new Lesson();
        $this->courses = new Course();
        $this->users = new User();
        $this->courseOrder = new CourseOrder();
        $this->wishes = new Wish();
```

```
    }

    public function index()
    {
        $currentAuthor = session()->get('user')->username;
        return $this->courses->getCoursesForInstructor($currentAuthor);
    }

    public function vote(Request $request)
    {
        $answer = $request->input('answer');
        $idUser = session('user')->id_user;

        try {
            $this->instructorModel->voting($answer, $idUser);
            $this->instructorModel->updateToInstructor($idUser);
            return response([], 201);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[InstructorController::class,
"update"]');
            return Helper::returnGenericError();
        }
    }

    public function edit($id)
    {
        $this->data['categories'] = $this->categories->getAllCateogries();
        $this->data['topics'] = $this->topics->getAllTopics();
        $this->data['lessonsEdit'] = $this->lessons->getAllLessonsForCourse($id);
        $this->data['course'] = $this->courses->getSingleCourse($id);
        return view('pages.admin.courses_edit', $this->data);
    }

    public function update(UpdateCourseRequest $request, $id)
    {
        $courseName = $request->input('courseName');
        $description = $request->input('description');
        $price = $request->input('coursePrice');
        $totalHours = $request->input('courseHours');
        $idCategory = $request->input('category');
        $updatedAt = date('Y-m-d H-i-s', time());
        $image = $request->file('courseImage');

        DB::beginTransaction();
        if ($image != null) {
            $newImage = ImageHelper::insertImage($image);
            try {
                $this->courseTopics->deleteCourse($id);
                $this->courses->updateCourseWithImage($id, $courseName, $description,
$price, $totalHours, $newImage[0], $newImage[1], $idCategory, $updatedAt);
```

```
                // delete lessons of course
                $this->lessons->deleteCourseLesson($id);

                // insert lessons of course
                if (is_array($request->input('lesson'))) {
                    foreach ($request->input('lesson') as $lesson) {
                        $this->lessons->insertCourseLesson($lesson, $id);
                    }
                } else {
                    $this->lessons->insertCourseLesson($request->input('lesson'), $id);
                }

                foreach ($request->input('topicsChb') as $topic) {
                    $this->courseTopics->insertCourseTopic($id, $topic);
                }

                DB::commit();

                Logs::loggingSuccess('Author just updated a course.');
                return redirect()->route('insturctorEdit', ['id' => $id])->with('success', 'Course has been updated.');
            } catch (PDOException $ex) {
                DB::rollBack();
                Logs::logging($ex->getMessage(), '[InstructorController::class, "update"]');
                return Helper::returnGenericError();
            }
        } else {
            try {
                $this->courseTopics->deleteCourse($id);
                $this->courses->updateCourseWithoutImage($id, $courseName, $description, $price, $totalHours, $idCategory, $updatedAt);

                // delete lessons of course
                $this->lessons->deleteCourseLesson($id);

                // insert lessons of course
                if (is_array($request->input('lesson'))) {
                    foreach ($request->input('lesson') as $lesson) {
                        $this->lessons->insertCourseLesson($lesson, $id);
                    }
                } else {
                    $this->lessons->insertCourseLesson($request->input('lesson'), $id);
                }

                foreach ($request->input('topicsChb') as $topic) {
                    $this->courseTopics->insertCourseTopic($id, $topic);
                }

                DB::commit();
```

```
                Logs::loggingSuccess('Author just updated a course.');

                return redirect()->route('insturctorEdit', ['id' => $id])-
>with('success', 'Course has been updated.');
            } catch (PDOException $ex) {
                DB::rollBack();
                Logs::logging($ex->getMessage(), '[InstructorController::class,
"update"]');
                return Helper::returnGenericError();
            }
        }
    }

    public function create()
    {
        return view('pages.admin.courses', ['categories' => $this->categories-
>getAllCateogries(), 'topics' => $this->topics->getAllTopics(), 'lessons' => $this-
>lessons->getAllLessons()]);
    }

    public function store(Request $request, AddCourseRequest $addCourseRequest)
    {
        $courseName = $request->input('courseName');
        $description = $request->input('description');
        $price = $request->input('coursePrice');
        $totalHours = $request->input('courseHours');

        $idUser = session()->get('user')->id_user;
        $user = $this->users->getSingleUser($idUser);
        $author = $user->username;

        $imageSmall = $request->file('courseImage');
        $imageBig = ImageHelper::insertImage($imageSmall);

        $idCategory = $request->input('category');

        DB::beginTransaction();
        try {
            $idCourse = $this->courses->insertCourse($courseName, $description, $price,
$totalHours, $author, $imageBig[0], $imageBig[1], $idCategory);

            if (is_array($request->input('lesson'))) {
                foreach ($request->input('lesson') as $lesson) {
                    $this->lessons->insertCourseLesson($lesson, $idCourse);
                }
            } else {
                $this->lessons->insertCourseLesson($request->input('lesson'),
$idCourse);
            }

            foreach ($request->input('topicsChb') as $topic) {
```

```
                $this->courseTopics->insertCourseTopic($idCourse,  $topic);
            }

            DB::commit();

            Logs::loggingSuccess('Author just added a new course.');
            return redirect()->back()->with('success', 'Course has been added.');
        } catch (PDOException $ex) {
            DB::rollBack ();
            Logs::logging($ex->getMessage(), '[InstructorController::class, "store"]');
            return Helper::returnGenericError();
        }
    }

    public function destroy(Request $request, $id)
    {
        DB::beginTransaction();
        try {
            $this->courseTopics->deleteCourse($id);
            $this->lessons->deleteCourseLesson($id);
            $this->courseOrder->deleteCourseFromOrder($id);
            $this->wishes->deleteCoursFromWish($id);
            $this->courses->deleteCourse($id);

            DB::commit();

            Logs::loggingSuccess('Author just deleted a course.');
            return response([], 204);
        } catch (PDOException $ex) {
            DB::rollback();

            Logs::logging($ex->getMessage(), '[InstructorController::class,
"destroy"]');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.7 LessonController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Services\Helper;
use App\Models\Lesson;
use App\Http\Services\Logs;
use Illuminate\Http\Request;
use App\Http\Controllers\Exception;
```

```php
use PDOException;


class LessonController extends Controller
{
    private $lessonModel;

    public function __construct()
    {
        $this->lessonModel = new Lesson();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {

    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {

    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {

    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
```

```
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {

    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy(Request $request, $id)
    {
        try {
            $this->lessonModel->deleteLesson($id);
            Logs::loggingSuccess('Admin deleted a lesson');
            return response([], 204);
        } catch (\Exception $ex) {
            Logs::logging($ex->getMessage(), '[LessonController::class, "destroy"]');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.8 WishController.php

```
<?php

namespace App\Http\Controllers;
```

```php
use App\Http\Services\Logs;
use App\Models\Wish;
use Illuminate\Http\Request;
use PDOException;

class WishController extends Controller
{
    private $modelWish;

    public function __construct()
    {
        $this->modelWish = new Wish();
    }

    public function addNewWish(Request $request)
    {
        $idCourse = $request->input('idCourse');

        if (!session()->has('user')) {
            return response([], 401); //  unauthorized
        }

        $idUser = session('user')->id_user;

        try {
            $ifExists = $this->modelWish->checkIfAlereadyExists($idUser, $idCourse);
            if ($ifExists) {
                return response('Course is already in your wishlist', 400);
            }

            $this->modelWish->addWish($idUser, $idCourse);
            Logs::loggingSuccess('User: ' . session('user')->username . ', Added
Wish');
            return response('Successfully added course to wishlist', 201);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[WishController::class, "addNewWish"]');
            return response([], 500);
        }
    }

    public function numberOfWishes()
    {
        $idUser = session('user')->id_user ?? 0;
        try {
            $info = $this->modelWish->numberOfWishes($idUser);
            return response($info);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[WishController::class,
"numberOfWishes"]');
            return response([], 500);
        }
```

```
    }

    public function getAllWishesForOneUser()
    {
        $idUser = session('user')->id_user;

        try {
            $info = $this->modelWish->getAllWishesForOneUser($idUser);
            return response($info);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[WishController::class,
"getAllWishesForOneUser"]');
            return response([], 500);
        }
    }

    public function deleteWish(Request $request)
    {
        $idWish = $request->input('idWish');

        try {
            $this->modelWish->deleteWish($idWish);
            Logs::loggingSuccess('User: ' . session('user')->username . ', Removed
Wish');
            return response([], 204);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[WishController::class, "deleteWish"]');
            return response([], 500);
        }
    }
}
```

4.2.9 Admin/CategoryController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use App\Http\Requests\AddCategoryRequest;
use App\Http\Requests\UpdateCategoryRequest;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use App\Models\Category;
use Illuminate\Http\Request;
use PDOException;

class CategoryController extends Controller
{
```

```
    private $categoryModel;
    public function __construct()
    {
        $this->categoryModel = new Category();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return $this->categoryModel->getCategoriesForAdmin();
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('pages.admin.categories');
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request, AddCategoryRequest $AddCatRequest)
    {
        $name = $request->input('categoryName');

        $fileName = time() . '-category-' . $request->categoryImage-
>getClientOriginalName();
        $request->categoryImage->move(public_path('img/categories'),  $fileName);

        try {
            $this->categoryModel->addCategory($name, $fileName);
            Logs::loggingSuccess('Admin added a new category.');
            return redirect()->back()->with('success', 'Successfully added new
category.');
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[CategoryController::class, "store"]');
            return Helper::returnGenericError();
        }
    }

    /**
```

```
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('pages.admin.categories_edit', ['category' => $this->categoryModel-
>getSingleCategory($id)]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(UpdateCategoryRequest $request, $id)
    {
        $categoryName = $request->input('categoryName');
        $categoryImage = $request->file('categoryImage');
        $updatedAt = date('Y-m-d H-i-s', time());

        if ($categoryImage != null) {
            $fileName = time() . '-category-' . $request->categoryImage-
>getClientOriginalName();
            $request->categoryImage->move(public_path('img/categories'), $fileName);

            try {
                $this->categoryModel->updateCategoryWithImage($id, $categoryName,
$fileName, $updatedAt);
                Logs::loggingSuccess('Admin updated category.');
                return redirect()->route('categories.edit', ['category' => $id])-
>with('success', 'Successfully updated category.');
            } catch (PDOException $ex) {
                Logs::logging($ex->getMessage(), '[CategoryController::class,
"update"]');
                return Helper::returnGenericError();
            }
```

```
        } else {
            try {
                $this->categoryModel->updateCategoryWithoutImage($id, $categoryName,
$updatedAt);
                Logs::loggingSuccess('Admin updated category.');
                return redirect()->route('categories.edit', ['category' => $id])-
>with('success', 'Successfully updated category.');
            } catch (PDOException $ex) {
                Logs::logging($ex->getMessage(), '[CategoryController::class,
"update"]');
                return Helper::returnGenericError();
            }
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        try {
            $this->categoryModel->deleteCategory($id);
            Logs::loggingSuccess('Admin just deleted category.');
            return response([], 204);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[CategoryController::class, "destroy"]');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.10 Admin/ContactController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use App\Models\ContactMail;
use Illuminate\Http\Request;
use PDOException;

class ContactController extends Controller
{
```

```
    private $data;
    private  $modelContact;
    public function __construct()
    {
        $this->modelContact = new ContactMail();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return $this->modelContact->getMailsForAdmin();
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('pages.admin.contact_mails');
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        //
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
```

```
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        //
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        //
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        try {
            $this->modelContact->deleteContactMail($id);
            Logs::loggingSuccess('Admin just deleted mail.');
            return response([], 204);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[ContactController::class, "destroy"]');
            Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.11 Admin/CourseController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;

use App\Http\Requests\AddCourseRequest;
use App\Http\Requests\UpdateCourseRequest;
```

```php
use App\Http\Services\Helper;
use App\Http\Services\ImageHelper;
use App\Http\Services\Logs;

use App\Models\Category;
use App\Models\CourseOrder;
use App\Models\CourseTopic;
use App\Models\Lesson;
use App\Models\Topic;
use App\Models\Course;

use App\Models\User;
use App\Models\Wish;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

use PDOException;

class CourseController
{
    private $data;
    private $categories;
    private $topics;
    private $courses;
    private $courseTopics;
    private $lessons;
    private $courseOrder;
    private $wishes;
    private $users;

    public function __construct()
    {
        $this->categories = new Category();
        $this->topics = new Topic();
        $this->courses = new Course();
        $this->courseTopics = new CourseTopic();
        $this->lessons = new Lesson();
        $this->courseOrder = new CourseOrder();
        $this->wishes = new Wish();
        $this->users = new User();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return $this->courses->getCoursesForAdmin();
    }
```

```
    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('pages.admin.courses', ['categories' => $this->categories-
>getAllCateogries(), 'topics' => $this->topics->getAllTopics(), 'lessons' => $this-
>lessons->getAllLessons()]);
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request, AddCourseRequest $addCourseRequest)
    {
        $courseName = $request->input('courseName');
        $description = $request->input('description');
        $price = $request->input('coursePrice');
        $totalHours = $request->input('courseHours');

        $idUser = session()->get('user')->id_user;
        $user = $this->users->getSingleUser($idUser);
        $author = $user->username . ' - Admin';

        $imageSmall = $request->file('courseImage');
        $imageBig = ImageHelper::insertImage($imageSmall);

        $idCategory = $request->input('category');

        DB::beginTransaction();
        try {
            $idCourse = $this->courses->insertCourse($courseName, $description, $price,
$totalHours, $author, $imageBig[0], $imageBig[1], $idCategory);

            if (is_array($request->input('lesson'))) {
                foreach ($request->input('lesson') as $lesson) {
                    $this->lessons->insertCourseLesson($lesson, $idCourse);
                }
            } else {
                $this->lessons->insertCourseLesson($request->input('lesson'),
$idCourse);
            }


            foreach ($request->input('topicsChb') as $topic) {
                $this->courseTopics->insertCourseTopic($idCourse, $topic);
```

```
            }

            DB::commit();

            Logs::loggingSuccess('Admin just added a new course.');
            return redirect()->route('courses.create')->with('success', 'Course has
been added.');
        } catch (PDOException $ex) {
            DB::rollBack ();
            Logs::logging($ex->getMessage(), '[CourseController::class, "store"]');
            return Helper::returnGenericError();
        }
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $this->data['categories'] = $this->categories->getAllCateogries();
        $this->data['topics'] = $this->topics->getAllTopics();
        $this->data['lessonsEdit'] = $this->lessons->getAllLessonsForCourse($id);
        $this->data['course'] = $this->courses->getSingleCourse($id);
        return view('pages.admin.courses_edit', $this->data);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(UpdateCourseRequest $request, $id)
    {
        $courseName = $request->input('courseName');
        $description = $request->input('description');
```

```
        $price = $request->input('coursePrice');
        $totalHours = $request->input('courseHours');
        $idCategory = $request->input('category');
        $updatedAt = date('Y-m-d H-i-s', time());
        $image = $request->file('courseImage');

        DB::beginTransaction();
        if ($image != null) {
            $newImage = ImageHelper::insertImage($image);
            try {
                $this->courseTopics->deleteCourse($id);
                $this->courses->updateCourseWithImage($id, $courseName, $description,
$price, $totalHours, $newImage[0], $newImage[1], $idCategory, $updatedAt);

                // delete lessons of course
                $this->lessons->deleteCourseLesson($id);

                // insert lessons of course
                if (is_array($request->input('lesson'))) {
                    foreach ($request->input('lesson') as $lesson) {
                        $this->lessons->insertCourseLesson($lesson, $id);
                    }
                } else {
                    $this->lessons->insertCourseLesson($request->input('lesson'), $id);
                }

                foreach ($request->input('topicsChb') as $topic) {
                    $this->courseTopics->insertCourseTopic($id, $topic);
                }

                DB::commit();

                Logs::loggingSuccess('Admin just updated a course.');
                return redirect()->route('courses.edit', ['course' => $id])-
>with('success', 'Course has been updated.');
            } catch (PDOException $ex) {
                DB::rollBack();
                Logs::logging($ex->getMessage(), '[CourseController::class,
"update"]');
                return Helper::returnGenericError();
            }
        } else {
            try {
                $this->courseTopics->deleteCourse($id);
                $this->courses->updateCourseWithoutImage($id, $courseName,
$description, $price, $totalHours, $idCategory, $updatedAt);

                // delete lessons of course
                $this->lessons->deleteCourseLesson($id);

                // insert lessons of course
```

```
                if (is_array($request->input('lesson'))) {
                    foreach ($request->input('lesson') as $lesson) {
                        $this->lessons->insertCourseLesson($lesson, $id);
                    }
                } else {
                    $this->lessons->insertCourseLesson($request->input('lesson'), $id);
                }

                foreach ($request->input('topicsChb') as $topic) {
                    $this->courseTopics->insertCourseTopic($id, $topic);
                }

                DB::commit();

                Logs::loggingSuccess('Admin just updated a course.');

                return redirect()->route('courses.edit', ['course' => $id])-
>with('success', 'Course has been updated.');
            } catch (PDOException $ex) {
                DB::rollBack();
                Logs::logging($ex->getMessage(), '[CourseController::class,
"update"]');
                return Helper::returnGenericError();
            }
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy(Request $request, $id)
    {
        DB::beginTransaction();
        try {
            $this->courseTopics->deleteCourse($id);
            $this->lessons->deleteCourseLesson($id);
            $this->courseOrder->deleteCourseFromOrder($id);
            $this->wishes->deleteCoursFromWish($id);
            $this->courses->deleteCourse($id);

            DB::commit();

            Logs::loggingSuccess('Admin just deleted a course.');
            return response([], 204);
        } catch (PDOException $ex) {
            DB::rollback();

            Logs::logging($ex->getMessage(), '[CourseController::class, "destroy"]');
```

```
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.12 Admin/TopicController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use App\Http\Requests\AddUpdateTopicRequest;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use App\Models\Topic;
use Illuminate\Http\Request;
use PDOException;

class TopicController extends Controller
{
    private $topicModel;
    public function __construct()
    {
        $this->topicModel = new Topic();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return $this->topicModel->getAdminTopics();
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('pages.admin.topics');
    }

    /**
     * Store a newly created resource in storage.
     *
```

```
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request, AddUpdateTopicRequest $topicRequest)
    {
        $topicName = $request->input('topicName');

        try {
            $this->topicModel->addTopic($topicName);
            Logs::loggingSuccess('Admin added a new topic.');
            return redirect()->back()->with('success', 'Successfully added new
topic.');
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[TopicController::class, "store"]');
            return Helper::returnGenericError();
        }
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('pages.admin.topics_edit', ['topic' => $this->topicModel-
>getSingleTopic($id)]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(AddUpdateTopicRequest $request, $id)
    {
        $topic = $request->input('topicName');
```

```
        try {
            $this->topicModel->updateTopic($id, $topic);
            Logs::loggingSuccess('Admin updated topic.');
            return redirect()->route('topics.edit', ['topic' => $id])->with('success',
'Successfully updated topic');
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[TopicController::class, "update"]');
            return Helper::returnGenericError();
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        try {
            $this->topicModel->deleteTopic($id);
            Logs::loggingSuccess('Admin deleted topic');
            return response([], 204);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[TopicController::class, "destroy"]');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.2.13 Admin/UserController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use App\Http\Requests\AddUserRequest;
use App\Http\Requests\UpdateUserRequest;
use App\Http\Services\Helper;
use App\Http\Services\Logs;
use App\Http\Services\UserService;
use App\Models\CourseOrder;
use App\Models\Order;
use App\Models\Role;
use App\Models\User;
use Exception;
use Illuminate\Http\Request;
use PDOException;
```

```php
class UserController extends Controller
{
    private $userModel;
    private $roleModel;
    private $orderModel;
    private $orderCourseModel;
    public function __construct()
    {
        $this->userModel = new User();
        $this->roleModel = new Role();
        $this->orderModel = new Order();
        $this->orderCourseModel = new CourseOrder();
    }
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return $this->userModel->getUsersForAdmin();
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('pages.admin.users', ['roles' => $this->roleModel->getAllRoles()]);
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(AddUserRequest $request)
    {
        $created_at = date('Y-m-d H-i-s', time());
        $obj = [
            'username' => $request->input('username'),
            'email' => $request->input('email'),
            'password' => md5($request->input('password')),
            'active' => 0,
            'is_instructor' => 0,
            'id_role' => $request->input('role'),
            'created_at' => $created_at,
```

```
            'updated_at' => $created_at
        ];

        try {
            $this->userModel->insert($obj);
            Logs::loggingSuccess('Admin added a user');
            return response([], 201);
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[UserController::class, "store"]');
            return Helper::returnGenericErrorAjax();
        }
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        return view('pages.admin.users_edit', ['user' => $this->userModel-
>getSingleUser($id), 'roles' => $this->roleModel->getAllRoles()]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(UpdateUserRequest $request, $id)
    {
        $service = new UserService();
        try {
            $service->updateUser($request, $id);
            Logs::loggingSuccess('Admin just updated a user.');
            return redirect()->route('users.edit', ['user' => $id])->with('success',
'Successfully updated user.');
```

```
        } catch (PDOException $ex) {
            Logs::logging($ex->getMessage(), '[UserController::class, "update"]');
            return Helper::returnGenericError();
        }
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        try {
            $this->userModel->deleteUserVoting($id);
            $this->userModel->deleteUser($id);
            Logs::loggingSuccess('Admin deleted a user');
            return response([], 204);
        } catch (Exception $ex) {
            Logs::logging($ex->getMessage(), '[UserController::class, "destroy"]');
            return Helper::returnGenericErrorAjax();
        }
    }
}
```

4.3 Middlewares

4.3.1 AdminMiddleware.php

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;

class AdminMiddleware
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        if (session()->has('user') && session('user')->id_role == 1) {
```

```
            return $next($request);
        }

        return abort(404);
    }
}
```

4.3.2 AuthoriseMiddleware404.php

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;

class AuthoriseMiddleware404
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        if (session('user')) {
            return $next($request);
        }
        return abort(404);
    }
}
```

4.3.3 AuthoriseMiddleware.php

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;

class AuthoriseMiddleware
{
    /**
```

```
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        if (session('user')) {
            return $next($request);
        }
        return redirect()->route('login');
    }
}
```

4.3.4 RecordAccessToPage.php

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;

class RecordAccessToPage
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        $date = date('d-m-y H:i:s');

        if (session()->has('user')) {
            $writeIn = ($date . "\t" . $request->ip() . "\t" . 'Authorized: ' .
session()->get('user')->email . "\t" . $request->url() . "\n");
            $openFile = fopen(storage_path() . '/app/file_access.txt', 'a');

            if ($openFile) {
                $date;
                fwrite($openFile, $writeIn);
                fclose($openFile);
            }
        } else {
```

```
            $writeIn = ($date . "\t" . $request->ip() . "\t" . 'Not Authorized: ' .
"\t" . $request->url() . "\n");
            $openFile = fopen(storage_path() . '/app/file_access.txt', 'a');

            if ($openFile) {
                fwrite($openFile, $writeIn);
                fclose($openFile);
            }
        }

        return $next($request);
    }
}
```

4.4 Services

4.4.1 Helper.php

```
<?php

namespace App\Http\Services;

class Helper
{
    public static function returnGenericError()
    {
        return redirect()->back()->with('error', 'Error on server! Try again later.');
    }

    public static function returnGenericErrorAjax()
    {
        return response(['error' => 'Error on server, please try later.'], 500);
    }

    public static function getTime()
    {
        return date('Y-m-d H:i:s', time());
    }
}
```

4.4.2 ImageHelper.php

```
<?php

namespace App\Http\Services;


use Intervention\Image\ImageManagerStatic as Image;
```

```php
class ImageHelper
{
    public static function insertImage($image)
    {
        $dir = public_path() . '/img/courses/';

        $imgSmall = time() . '-uploaded-' . $image->getClientOriginalName();

        $image->move($dir, $imgSmall);

        $imgNew = Image::make($dir . $imgSmall);
        $imgNew->backup();
        $imgNew->resize(400, null, function ($constraint) {
            $constraint->aspectRatio();
        });
        $imgNew->save($dir . $imgSmall);
        $imgNew->reset();

        $imgNew = Image::make($dir . $imgSmall);
        $imgNew->resize(750, null, function ($constraint) {
            $constraint->aspectRatio();
        });
        $imgBig = 'big-' . $imgSmall;
        $imgNew->save($dir . $imgBig);

        return [$imgSmall, $imgBig];
    }
}
```

4.4.3 Logs.php

```php
<?php

namespace App\Http\Services;

use Illuminate\Support\Facades\Log;

class Logs
{
    public static function logging($ex, $msg)
    {
        Log::channel('single')->error($msg . ', With a bug: ' . $ex);
    }

    public static function loggingSuccess($msg)
    {
        Log::channel('single')->notice($msg);
    }
}
```

4.4.4 UserService.php

```php
<?php

namespace App\Http\Services;

use App\Models\User;
use Illuminate\Http\Request;

class UserService
{
    public function updateUser(Request $request, $id)
    {
        $user = new User();

        $obj = [
            'username' => $request->input('username'),
            'email' => $request->input('email'),
            'updated_at' => date('Y-m-d H-i-s', time())
        ];

        if ($request->has('role')) {
            $obj['id_role'] = $request->input('role');
        }

        $active = $request->input('active');
        if ($active != null) {
            $obj['active'] = $active;
        } else {
            $obj['active'] = 0;
        }

        $isInstructor = $request->input('isInstructor');
        if ($isInstructor != null) {
            $obj['is_instructor'] = $isInstructor;
        } else {
            $obj['is_instructor'] = 0;
        }

        $password = $request->input('password');
        if ($password != null) {
            $obj['password'] = md5($password);
        }

        $user->updateUser($obj, $id);
    }
}
```

4.5 Requests

4.5.1 AddCategoryRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class AddCategoryRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'categoryName' => 'required|string',
            'categoryImage' => 'required|image|mimes:png,jpg,jpeg,gif,svg|max:2048'
        ];
    }
}
```

4.5.2 AddCourseRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Support\Facades\Validator;
class AddCourseRequest extends FormRequest
```

```
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'courseName' => 'required|string',
            'description' => 'required|string',
            'coursePrice' => 'required|min:1|numeric',
            'courseHours' => 'required|min:1|numeric',
            'courseImage' => 'required|image|mimes:png,jpg,jpeg,gif,svg|max:2048',
            'category' => 'exists:category,id_category',
            'lesson' => 'required|array',
            'lesson.*' => 'required|string',
            'topicsChb' => 'required|array'
        ];
    }

    public function messages()
    {
        return [
            'courseName.required' => 'Course name is required.',
            'description.required' => 'Description is required.',
            'coursePrice.required' => 'Price is required.',
            'coursePrice.numeric' => 'Price must be integer',
            'courseHours.required' => 'Total Hours is required.',
            'courseHours.numeric' => 'Total Hours must be integer.',
            'courseImage.required' => 'Image is required.',
            'courseImage.image' => 'File must be image format.',
            'courseImage.max' => 'Image size is 8000kb maximum.',
            'lesson.required' => 'Lesson is required.',
            'category.exists' => 'You must choose category.',
            'topicsChb.required' => 'You must choose topic.',
        ];
    }
}
```

4.5.3 AddUpdateTopicRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class AddUpdateTopicRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'topicName' => 'required|string'
        ];
    }
}
```

4.5.4 AddUserRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class AddUserRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
```

```
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'username' => 'required|regex:/^[\d\w\_\-\.]{6,30}$/|unique:user,username',
            'email' => 'required|email|unique:user,email',
            'password' => ['required', 'min:4', 'regex:/^[A-z]{3,}[0-9]{1,}$/'],
            'confirmPassword' => 'required|same:password',
            'active' => 'nullable',
            'role' => 'required|exists:role,id_role',
        ];
    }
}
```

4.5.5 ContactRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class ContactRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
```

```
            'email' => 'required|email',
            'subject' => 'required|regex:/[A-z0-9]+/',
            'message' => 'required|regex:/[A-z0-9]+/'
        ];
    }

    public function messages()
    {
        return [
            'email.required' => 'Email is required.',
            'email.email' => 'Email is not in good format.',
            'subject.required' => 'Subject is required.',
            'subject.regex' => 'Subject is not in good format.',
            'message.required' => 'Message is required.',
            'message.regex' => 'Message is not in good format.'
        ];
    }
}
```

4.5.6 FilterCoursesRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class FilterCoursesRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'search' => 'nullable|string',
            'categories' => 'nullable|array',
```

```
            'topicChb' => 'nullable|array',
            'sort' => 'nullable|alpha',
            'showing' => 'nullable|numeric'
        ];
    }
}
```

4.5.7 LoginRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class LoginRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'username' => 'required|regex:/^[\d\w\_\-\.]{6,30}$/|exists:user,username',
            'password' => 'required|regex:[^[A-z0-9.?!]{6,}]'
        ];
    }

    public function message()
    {
        return [
            'username.required' => 'Username is required.',
            'username.regex' => 'Username is not in good format.',
            'username.exists' => 'Username doesn"t exists in our system.',
```

```
            'password.required' => 'Password is required.',
            'password.regex' => 'Password is not in good format.'
        ];
    }
}
```

4.5.8 RegistrationRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class RegistrationRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'username' => 'required|regex:/^[\d\w\_\-\.]{6,30}$/|unique:user,username',
            'email' => 'required|email|unique:user,email',
            'password' => 'required|string|min:4|regex:/^[A-z]{3,}[0-9]{1,}$/',
            'passwordConfirm' => 'required|same:password'
        ];
    }
```

```
    public function messages()
    {
        return [
            'username.required' => 'Username is required.',
            'username.regex' => 'Username must have min. 6 characters.',
            'username.unique' => 'This username already exists.',
            'email.required' => 'Email is required.',
            'email.email' => 'Email is not in propper format.',
            'email.unique' => 'This email is already taken.',
            'password.require' => 'Password is required.',
            'password.regex' => 'Password must have 3 letters and min. 1 number.',
            'passwordConfirm.same' => 'Passwords do not match.'
        ];
    }
}
```

4.5.9 UpdateCategoryRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class UpdateCategoryRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true ;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'categoryName' => 'required|string',
            'categoryImage' => 'nullable|image|mimes:png,jpg,jpeg,gif,svg|max:2048'
        ];
    }
}
```

4.5.10 UpdateCourseRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class UpdateCourseRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'courseName' => 'required|string',
            'description' => 'required|string',
            'coursePrice' => 'required|min:1|numeric',
            'courseHours' => 'required|min:1|numeric',
            'courseImage' => 'nullable|image|mimes:png,jpg,jpeg,gif,svg|max:2048',
            'lesson' => 'required|array',
            'lesson.*' => 'required|string',
            'category' => 'exists:category,id_category',
            'topicsChb' => 'required|array'
        ];
    }

    public function messages()
    {
        return [
            'topicsChb.required' => 'You must choose topic.',
            'courseImage.image' => 'File must be image',
            'courseImage.max' => 'Image must be less than 2048 kb.',
            'lesson.required' => 'Lesson URL is required',
            'category.exists' => 'You must choose category.'
```

```
        ];
    }
}
```

4.5.11 UpdateUserRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Validation\Rule;

class UpdateUserRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        $id = $this->input('hiddenUserField');
        return [
            'username' => ['required', 'regex:/^[\d\w\_\-\.]{6,30}$/',
Rule::unique('user', 'username')->ignore($id, 'id_user')],
            'email' => ['required', 'email', Rule::unique('user', 'email')->ignore($id,
'id_user')],
            'password' => 'nullable|min:4|regex:/^[A-z]{3,}[0-9]{1,}$/',
            'active' => 'nullable',
            'role' => 'required|exists:role,id_role'
        ];
    }

    public function messages()
    {
        return [
            'username.required' => 'Username is required...',
            'username.regex' => 'Username must have min. 6 characters.',
```

```
            'username.unique' => 'This username already exists',
            'email.required' => 'Email is required.',
            'email.email' => 'Email is not in good format.',
            'email.unique' => 'This email already exists',
            'password.regex' => 'Password must have 3 letters and min. 1 number.',
            'role.required' => 'Role is required.',
            'role.exists' => 'Role does not exists in system.'
        ];
    }
}
```

4.6 Views

4.6.1 components

4.6.1.1 admin/form.blade.php

```
<section class="content-header">
    <div class="container-fluid">
        <div class="row mb-2">
            <div class="col-sm-6">
                <h1>{{ $headTitle }}</h1>
            </div>
        </div>
    </div>
</section>

<section class="content">
    <div class="row">
        <div class="col-md-4">
            <div class="card card-primary">
                <div class="card-header">
                    <h3 class="card-title">{{ $title }}</h3>
                </div>

                <div class="card-body">
                    @if (isset($id))
                        <form action="{{ route($action, $id) }}" method="POST"
enctype="multipart/form-data">
                            <input type="hidden" name="idCourseHdn" value="{{ $id }}">
                    @else
                        <form action="{{ route($action) }}" method="POST"
enctype="multipart/form-data">
                    @endif
```

```
                    @method($method)

                    @csrf

                    @if (isset($arrayInputs))
                        @foreach ($arrayInputs as $input)
                            <div class="form-group">
                                @if (!isset($input['noLabel']))
                                    <label>{{ $input['label'] ?? '' }}</label>
                                @endif

                                @if (isset($input['image']))
                                    <img src="{{ asset("/img/".$input['folder'] .
$input['image'] ?? '') }}" alt="{{ $input['name'] ?? '' }}" class="small-img" />
                                @endif

                                @if (($input['type']) == 'textarea')
                                    <textarea name="{{ $input['name'] }}" value=""
placeholder="" class="form-control" cols="25" rows="10">{{ $input['value'] ?? '' }}
</textarea>
                                @endif

                                @if (($input['type']) !== 'textarea')
                                    <input type="{{ $input['type'] ?? 'text' }}"
class="form-control"
                                        id="{{ $input['id'] ?? '' }}" value="{{
$input['value'] ?? '' }}"
                                        name="{{ $input['name'] ?? '' }}"
                                        step=".01"
                                        placeholder="{{ $input['placeholder'] ?? '' }}"
/>
                                @endif
                            </div>
                        @endforeach
                    @endif



                    @if (isset($arrayLessons))
                        @foreach ($arrayLessons as $lesson)
                            <div class="form-group lessonRow row" id="lesson">
                                <div class="col-11">
                                    <label>{{ $lesson->label ?? '' }}</label>

                                    <input type="{{ $lesson->type ?? 'text' }}"
class="form-control mb-8"
                                        id="{{ $lesson->value ?? '' }}" value="{{
$lesson->text ?? '' }}"
                                        name="{{ $lesson->name ?? '' }}"
                                        placeholder="{{ $lesson->placeholder ?? '' }}"
/>
                                </div>
```

```
                            </div>

                            <div class="center-align centeredBtn">
                                <button type="button" class="btn blue"
id="btnAddLesson"><i class="fa fa-plus"></i></button>
                            </div>
                        @endforeach
                    @endif

                    @if (isset($arrayLessonsEdit))
                        @foreach ($arrayLessonsEdit as $arl)
                        <label>{{ $arl->label ?? '' }}</label>

                            @if (isset($arl->option))
                                @foreach ($arl->option as $option)
                                    <div class="form-group lessonRow row" id="{{
$option['value'] }}">
                                        <div class="col-11">
                                            <input type="text" class="form-control mb-
8"
                                                id="{{ $option['value'] ?? '' }}"
                                                value="{{ $option['text'] ?? '' }}"
                                                step=".01"
                                                name="{{ $arl->name ?? '' }}" />
                                        </div>

                                        <div class="col-1">
                                            <button type="button" class="btn
btnRemoveLessonFromEdit" data-id='{{ $option['value'] }}'><i class="fa fa-
times"></i></button>
                                        </div>
                                    </div>
                                @endforeach
                            @endif
                        @endforeach
                        <div class="center-align centeredBtn">
                            <button type="button" class="btn blue" id="btnAddLesson"><i
class="fa fa-plus"></i></button>
                        </div>
                    @endif

                    @if (isset($arrayDropdowns))
                        @foreach ($arrayDropdowns as $dd)
                            <div class="form-group">
                                <label>{{ $dd->label ?? '' }}</label>

                                <select name="{{ $dd->name }}"  class="form-control">
                                    <option value="0">Choose</option>

                                    @foreach ($dd->option as $option)
                                        @if (isset($dd->selected))
```

```
                                        @if ($option['value'] == $dd->selected)
                                            <option selected value="{{
$option['value'] ?? '' }}">{{ $option['text'] }}</option>
                                        @else
                                            <option value="{{ $option['value'] ??
'' }}">{{ $option['text'] }}</option>
                                        @endif
                                    @else
                                        <option value="{{ $option['value'] ?? ''
}}">{{ $option['text'] }}</option>
                                    @endif
                                @endforeach
                            </select>
                        </div>
                    @endforeach
                @endif



                @if (isset($arrayCheckboxes))
                    @foreach ($arrayCheckboxes as $chb)
                        <div class="form-group">
                            <label>{{ $chb->label ?? '' }}</label>
                            <div class="d-flex justify-content-around flex-wrap">
                                @foreach ($chb->option as $input)
                                    @if (in_array($input['value'], $chb->selected
?? []))
                                        <span class="w-100 m-1"><input
type="checkbox"
                                                name="{{ $chb->name ?? '' }}"
                                                value="{{ $input['value'] ?? '' }}"
class="mr-2"
                                                checked="checked" />{{
$input['text'] ?? '' }}</span>
                                    @else
                                        <span class="w-100 m-1"><input
type="checkbox"
                                                name="{{ $chb->name ?? '' }}"
                                                value="{{ $input['value'] ?? '' }}"
                                                class="mr-2" />{{ $input['text'] ??
'' }}</span>
                                    @endif
                                @endforeach
                            </div>
                        </div>
                    @endforeach
                @endif

                @if (isset($button))
                    <div class="form-group">
```

```
                            <button type="{{ $button['type'] ?? 'submit' }}" name="{{
$button['name'] ?? '' }}"
                                    id="{{ $button['id'] ?? '' }}"
                                    class="btn btn-primary {{ $button['class'] ?? '' }}">{{
$button['value'] ?? 'Submit' }}</button>
                        </div>
                    @endif
                    </form>
                </div>
                <div class="card-footer">
                    <div class="js-notification"></div>
                    @if ($errors->any())
                        <div class="alert alert-danger pr mt-3">
                            <ul>
                                @foreach ($errors->all() as $error)
                                    <li>{{ $error }}</li>
                                @endforeach
                            </ul>
                        </div>
                    @endif
                    @if (session('success'))
                        <div class="alert alert-success mt-3">
                            {{ session('success') }}
                        </div>
                    @endif
                    @if (session('error'))
                        <div class="alert alert-danger mt-3">
                            {{ session('error') }}
                        </div>
                    @endif
                </div>
            </div>
        </div>

        <div class="col-md-8">
            <div class="card">
                <div class="card-body">
                    <table id="table" class="table table-bordered table-hover">

                    </table>
                    <div class="dataTables_paginate paging_simple_numbers pt-3"
id="table_paginate">

                    </div>
                    <div class="msgCrud">
                        @if (session()->has('updateMsg'))
                            {{ session()->get('updateMsg') }}
                        @endif
                    </div>
                </div>
            </div>
```

```
        </div>
    </div>
</section>
```

4.6.1.2 *instructor/form.blade.php*

```
<div class="header-instructor">
    <div class="row">
        <div class="col-6">
            <img src="{{ asset('/img/logo-learnings.png') }}" class="img-fluid" />
        </div>
        <div class="col-6 flex-link">
            <a class="exit-url" href="{{ url('/') }}">Exit</a>
        </div>
    </div>
</div>

<section class="content-header">
    <div class="container-fluid">
        <div class="row mb-2 centerText">
            <div class="col-sm-6 centerTitle">
                <h1>{{ $pageTitleInstructor }}</h1>
            </div>
        </div>
    </div>
</section>

<section class="content content-wrapper-instructor">
    <div class="row">
        <div class="col-md-4">
            <div class="card card-primary">
                @if (isset($id))
                    <a href="{{ url('/instructor') }}" class="go-back-link">&#x2190; Go
Back To Create Course</a>
                @endif
                <div class="card-header bckg">
                    <h3 class="card-title centerTitle">{{ $formTitleInstructor }}</h3>
                </div>

                <div class="card-body">
                    @if (isset($id))
                        <form action="{{ route($actionInstructor, $id) }}"
method="POST" enctype="multipart/form-data">
                            <input type="hidden" name="idCourseHdn" value="{{ $id }}">
                    @else
                        <form action="{{ route($actionInstructor) }}" method="POST"
enctype="multipart/form-data">
                    @endif

                    @method($methodInstructor)
```

```
                    @csrf

                    @if (isset($arrayInputsInstructor))
                        @foreach ($arrayInputsInstructor as $input)
                            <div class="form-group">
                                @if (!isset($input['noLabel']))
                                    <label>{{ $input['label'] ?? '' }}</label>
                                @endif

                                @if (isset($input['image']))
                                    <img src="{{ asset("/img/".$input['folder'] .
 $input['image'] ?? '') }}" alt="{{ $input['name'] ?? '' }}" class="small-img" />
                                @endif

                                @if (($input['type']) == 'textarea')
                                    <textarea name="{{ $input['name'] }}" value=""
placeholder="" class="form-control" cols="25" rows="10">{{ $input['value'] ?? '' }}
</textarea>
                                @endif

                                @if (($input['type']) !== 'textarea')
                                    <input type="{{ $input['type'] ?? 'text' }}"
class="form-control"
                                        id="{{ $input['id'] ?? '' }}" value="{{
$input['value'] ?? '' }}"
                                        name="{{ $input['name'] ?? '' }}"
                                        step=".01"
                                        placeholder="{{ $input['placeholder'] ?? '' }}"
/>
                                @endif
                            </div>
                        @endforeach
                    @endif


                    @if (isset($arrayLessonsInstructor))
                        @foreach ($arrayLessonsInstructor as $lesson)
                            <div class="form-group lessonRow row" id="lesson">
                                <div class="col-11">
                                    <label>{{ $lesson->label ?? '' }}</label>

                                    <input type="{{ $lesson->type ?? 'text' }}"
class="form-control mb-8"
                                        id="{{ $lesson->value ?? '' }}" value="{{
$lesson->text ?? '' }}"
                                        name="{{ $lesson->name ?? '' }}"
                                        placeholder="{{ $lesson->placeholder ?? '' }}"
/>
                                </div>
                            </div>
```

```
                            <div class="center-align centeredBtn">
                                <button type="button" class="btn blue"
id="btnAddLesson"><i class="fa fa-plus"></i></button>
                            </div>
                        @endforeach
                    @endif

                    @if (isset($arrayLessonsEdit))
                        @foreach ($arrayLessonsEdit as $arl)
                        <label>{{ $arl->label ?? '' }}</label>

                            @if (isset($arl->option))
                                @foreach ($arl->option as $option)
                                    <div class="form-group lessonRow row" id="{{
$option['value'] }}">
                                        <div class="col-11">
                                            <input type="text" class="form-control mb-
8"
                                                id="{{ $option['value'] ?? '' }}"
                                                value="{{ $option['text'] ?? '' }}"
                                                step=".01"
                                                name="{{ $arl->name ?? '' }}" />
                                        </div>

                                        <div class="col-1">
                                            <button type="button" class="btn
btnRemoveLessonFromEdit" data-id='{{ $option['value'] }}'><i class="fa fa-
times"></i></button>
                                        </div>
                                    </div>
                                @endforeach
                            @endif
                        @endforeach
                        <div class="center-align centeredBtn">
                            <button type="button" class="btn blue" id="btnAddLesson"><i
class="fa fa-plus"></i></button>
                        </div>
                    @endif

                    @if (isset($arrayDropdownsInstructor))
                        @foreach ($arrayDropdownsInstructor as $dd)
                            <div class="form-group">
                                <label>{{ $dd->label ?? '' }}</label>

                                <select name="{{ $dd->name }}"  class="form-control">
                                    <option value="0">Choose</option>

                                    @foreach ($dd->option as $option)
                                        @if (isset($dd->selected))
                                            @if ($option['value'] == $dd->selected)
```

```
                                                <option selected value="{{
$option['value'] ?? '' }}">{{ $option['text'] }}</option>
                                            @else
                                                <option value="{{ $option['value'] ??
'' }}">{{ $option['text'] }}</option>
                                            @endif
                                        @else
                                            <option value="{{ $option['value'] ?? ''
}}">{{ $option['text'] }}</option>
                                        @endif
                                    @endforeach
                                </select>
                            </div>
                        @endforeach
                    @endif

                    @if (isset($arrayCheckboxesInstructor))
                        @foreach ($arrayCheckboxesInstructor as $chb)
                            <div class="form-group chb-display">
                                <label>{{ $chb->label ?? '' }}</label>
                                <div class="d-flex justify-content-around flex-wrap">
                                    @foreach ($chb->option as $input)
                                        @if (in_array($input['value'], $chb->selected
?? []))
                                            <span class="w-100 m-1"><input
type="checkbox"
                                                    name="{{ $chb->name ?? '' }}"
                                                    value="{{ $input['value'] ?? '' }}"
class="mr-2"
                                                    checked="checked" />{{
$input['text'] ?? '' }}</span>
                                        @else
                                            <span class="w-100 m-1"><input
type="checkbox"
                                                    name="{{ $chb->name ?? '' }}"
                                                    value="{{ $input['value'] ?? '' }}"
                                                    class="mr-2" />{{ $input['text'] ??
'' }}</span>
                                        @endif
                                    @endforeach
                                </div>
                            </div>
                        @endforeach
                    @endif

                    @if (isset($buttonInstructor))
                        <div class="form-group">
                            <button type="{{ $buttonInstructor['type'] ?? 'submit' }}"
name="{{ $buttonInstructor['name'] ?? '' }}"
                                id="{{ $buttonInstructor['id'] ?? '' }}"
```

```
                                class="btn btn-primary {{ $buttonInstructor['class'] ??
'' }}">{{ $buttonInstructor['value'] ?? 'Submit' }}</button>
                        </div>
                    @endif
                    </form>
                </div>

                <div class="card-footer">
                    <div class="js-notification"></div>
                    @if ($errors->any())
                        <div class="alert alert-danger pr mt-3">
                            <ul>
                                @foreach ($errors->all() as $error)
                                    <li>{{ $error }}</li>
                                @endforeach
                            </ul>
                        </div>
                    @endif
                    @if (session('success'))
                        <div class="alert alert-success mt-3">
                            {{ session('success') }}
                        </div>
                    @endif
                    @if (session('error'))
                        <div class="alert alert-danger mt-3">
                            {{ session('error') }}
                        </div>
                    @endif
                </div>
            </div>
        </div>

        <div class="col-md-8">
            <div class="card">
                <div class="card-body">
                    <table id="table" class="table table-bordered table-hover">

                    </table>
                    <div class="dataTables_paginate paging_simple_numbers pt-3"
id="table_paginate">

                    </div>
                    <div class="msgCrud">
                        @if (session()->has('updateMsg'))
                            {{ session()->get('updateMsg') }}
                        @endif
                    </div>
                </div>
            </div>
        </div>
```

```
    </div>
</section>
```

4.6.1.3 single_course.blade.php

```
<div class="col-xl-4 col-md-6 col-12 pb-30 pt-10 pr-25">
    <div class="product {{ $course->id_course }}">
        <img src="{{ asset('/img/courses/' . $course->image_small) }}" alt="webdev"
class="img-fluid" />

        <div class="content">
            <div class="text-desc">
                <h4 class="title">
                    <a href="{{ url('courses', ['id' => $course->id_course]) }}">{{
$course->course_name }}</a>
                </h4>
                <a href="{{ route('courses', ['categories[]' => $course->id_category])
}}"
                    class="category">{{ $course->category_name }}</a>
            </div>

            <div class="course-price">
                <p class="pricee">&euro; {{ $course->price }} </p>
                <p>{{ floatval($course->total_hours) }} hours</p>
                <p>Author: {{ $course->author }}</p>

                <div class="iconsFa">
                    @if (session()->has('user'))
                        @if (isset($ids_learning_courses))
                            @if (session()->has('user') && in_array($course->id_course,
$ids_learning_courses))
                                <p>You already bought this course.</p>
                            @else
                                <a href="javascript:void(0)" class="add-course-to-cart"
                                    data-idcourse="{{ $course->id_course }}">
                                    <i class="fas fa-shopping-cart fa-2x clr"></i>
                                </a>
                                <a href="javascript:void(0)" class="wishhhh" data-
idcourse="{{ $course->id_course }}">
                                    <i class="fas fa-heart fa-2x clr"></i>
                                </a>
                            @endif
                        @else
                            <a href="javascript:void(0)" class="add-course-to-cart"
                                data-idcourse="{{ $course->id_course }}">
                                <i class="fas fa-shopping-cart fa-2x clr"></i>
                            </a>
                            <a href="javascript:void(0)" class="wishhhh" data-
idcourse="{{ $course->id_course }}">
                                <i class="fas fa-heart fa-2x clr"></i>
```

```
                        </a>
                    @endif
                @else
                    <a href="javascript:void(0)" class="alert-item-cart">
                        <i class="fas fa-shopping-cart fa-2x clr"></i>
                    </a>
                    <a href="javascript:void(0)" class="alert-item-wish">
                        <i class="fas fa-heart fa-2x clr"></i>
                    </a>
                @endif
            </div>
        </div>
    </div>
</div>
```

4.6.2 email

4.6.2.1 contact.blade.php

```
@component('mail::message')
    <strong>Subject</strong>
    {{ $subject }}
    <br />

    <strong>Email</strong>
    {{ $email }}

    <strong>Message</strong>
    {{ $message }}
@endcomponent
```

4.6.3 fixed

4.6.3.1 admin

4.6.3.1.1 header.blade.php

```
<nav class="main-header navbar navbar-expand navbar-white navbar-light">
    <ul class="navbar-nav">
        <li class="nav-item d-none d-sm-inline-block">
            <a href="{{ url('/') }}" class="nav-link">Go Back To Home Page</a>
        </li>
        <li class="nav-item d-none d-sm-inline-block">
```

```
            <a href="{{ url('/logout') }}" class="nav-link">Logout</a>
        </li>
    </ul>
</nav>
```

4.6.3.1.2 head.blade.php

```
<head>
    <title>@yield('title')</title>

    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-
scale=1.0, user-scalable=no" />
    <meta name="csrf-token" content="{{ csrf_token() }}" />

    <link href="{{ asset('css/style.css') }}" rel="stylesheet" type="text/css"
media="all" />
    <link rel="stylesheet" href="{{ asset('css/bootstrap.min.css') }}" />
    <link rel="stylesheet" href="{{ asset('css/admin_lite.min.css') }}" />
    <link rel="stylesheet" href="{{ asset('css/data_tables.min.css') }}" />
    <link href="{{ asset('fontawesome/css/fontawesome-all.min.css') }}"
rel="stylesheet" type="text/css" />
    <link href="https://fonts.googleapis.com/css2?family=Lato&display=swap"
rel="stylesheet" type="text/css" />

    <script src="{{ asset('js/jquery.min.js') }}"></script>
</head>
```

4.6.3.1.3 footer.blade.php

```
<footer class="main-footer">
    <strong>Copyright &copy; Marko Gačanović</strong>
</footer>

<aside class="control-sidebar control-sidebar-dark"></aside>

<script src="{{ asset('js/main.js') }}"></script>
```

4.6.3.1.4 sidebar.blade.php

```
<aside class="main-sidebar sidebar-dark-primary elevation-4">
    <div class="sidebar">
        <div class="user-panel mt-3 pb-3 mb-3 d-flex">
            <div class="image">
                <img src="{{ asset('img/avatar.png') }}" class="img-circle elevation-2"
alt="User Image">
```

```
            </div>
            <div class="info">
                <a href="#" class="d-block">Junior Developer</a>
            </div>
        </div>

        <nav class="mt-2">
            <ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview"
role="menu" data-accordion="false">
                <li class="nav-header">Menu</li>

                <li class="nav-item">
                    <a href="{{ route('logs') }}" class="nav-link"><i class="fas fa-
calendar"></i> Logs</a>
                </li>

                <li class="nav-item">
                    <a href="{{ route('courses.create') }}" class="nav-link"><i
class="fas fa-book"></i> Courses</a>
                </li>

                <li class="nav-item">
                    <a href="{{ route('categories.create') }}" class="nav-link"><i
class="fa fa-list-alt"></i>
                        Categories</a>
                </li>

                <li class="nav-item">
                    <a href="{{ route('topics.create') }}" class="nav-link"><i
class="fa fa-list-alt"></i>
                        Topics</a>
                </li>

                <li class="nav-item">
                    <a href="{{ route('users.create') }}" class="nav-link"><i class="fa
fa-user"></i>
                        Users</a>
                </li>

                <li class="nav-item">
                    <a href="{{ url('/orders') }}" class="nav-link"><i class="fas fa-
shopping-cart"></i> Orders</a>
                </li>

                <li class="nav-item">
                    <a href="{{ route('contact.create') }}" class="nav-link"><i
class="fas fa-envelope"></i> Contact Mails</a>
                </li>
            </ul>
        </nav>
    </div>
```

```
</aside>
```

4.6.3.2 user

4.6.3.2.1 banner.blade.php

```
<div class="wrapper bgded overlay gradient" style="background-image:url('{{
asset('img/background.jpg') }}');">
    <div id="pageintro" class="hoc clear">
        <article class="pageintro-wrap">
            <p>Knowledge is the key.</p>

            <footer>
                <form class="example" action="{{ route('courses') }}" method="GET">
                    <input type="text" placeholder="Search For Courses" aria-
label="true" name="search"
                        class="form-control" value="">
                    <button type="submit" class="buttonS"><i class="fa fa-
search"></i></button>
                </form>
            </footer>
        </article>
    </div>
</div>
```

4.6.3.2.2 footer.blade.php

```
<div class="wrapper row4">
    <footer id="footer" class="hoc clear">

        <div class="one_half first">
            <h6 class="heading">CATEGORIES</h6>
            <ul class="nospace linklist">
                @foreach ($categories as $c)
                    <li>
                        <article>
                            <p class="nospace btmspace-10">
                                <a href="{{ route('courses', ['categories[]' => $c-
>id_category]) }}">{{ $c->category_name }}</a>
                            </p>
                        </article>
                    </li>
                @endforeach
            </ul>
        </div>
        <div class="one_half">
            <h6 class="heading">PAGES</h6>
            <ul class="nospace linklist">
                <li><a href="{{ route('home') }}">Home</a></li>
```

```
                <li><a href="{{ route('courses') }}">Courses</a></li>
                <li><a href="{{ route('contactPage') }}">Contact</a></li>
                @if (!session()->has('user'))
                    <li><a href="{{ route('login') }}">Login</a></li>
                @else
                    @if (session()->has('user') && session()->get('user')->id_role ==
1)
                        <li><a href="{{ route('logs') }}">Admin</a></li>
                    @endif
                    @if (session()->has('user') && session()->get('user')->id_role !==
1)
                        <li><a href="{{ url('/learnings') }}">My Learnings</a></li>
                        <li><a href="{{ url('/instructor') }}">Instructor
Dashboard</a></li>
                    @endif
                    <li><a href="{{ route('logout') }}">Logout</a></li>
                @endif
            </ul>
        </div>

    </footer>
</div>

<div class="wrapper row5">
    <div id="copyright" class="hoc clear">
        <p class="fl_left">&copy; Copyright, <a href="{{ route('author') }}">Marko
Gačanović 38 / 17</a>   &   <a href="{{ asset('dokumentacija.pdf')
}}">Documentation</a> </p>

        <p class="fl_right">Template by <a target="_blank" href="https://www.os-
templates.com/" title="Free Website Templates">OS Templates</a></p>
    </div>
</div>

{{-- <a id="backtotop" href="#top"><i class="fas fa-chevron-up"></i></a> --}}
```

4.6.3.2.3 head.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>@yield('title')</title>

    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-
scale=1.0, user-scalable=no" />
    <meta name="description" content="@yield('description')" />
    <meta name="keywords" content="@yield('keywords')" />
```

```
    <link href="{{ asset('css/template_framework.css') }}" rel="stylesheet"
type="text/css" media="all" />
    <link href="{{ asset('css/style.css') }}" rel="stylesheet" type="text/css"
media="all" />
    <link href="{{ asset('css/bootstrap.min.css') }}" rel="stylesheet" type="text/css"
/>
    <link href="{{ asset('fontawesome/css/fontawesome-all.min.css') }}"
rel="stylesheet" type="text/css" />
    <link href="https://fonts.googleapis.com/css2?family=Lato&display=swap"
rel="stylesheet" type="text/css" />
    <link rel="icon" href="{{ url('favicon.png') }}">
</head>
```

4.6.3.2.4 header.blade.php

```
<body id="top">

    <div class="wrapper row0">
        <div id="topbar" class="hoc clear">
            <div class="fl_left">

                <ul class="nospace">
                    <li><a href="#"><img src="{{ asset('img/avatar.png') }}"
alt="avatar" /></a></li>
                    <p>My Account</p>
                    @if (session()->has('user'))
                        <p>{{ session()->get('user')->username }}</p>
                    @endif
                </ul>

            </div>
            <div class="fl_right">

                <ul class="nospace">
                    @if (session()->has('user') && session()->get('user')->id_role ==
1)
                        <h3>ADMIN</h3>
                    @endif
                    <li>
                        @if (session()->has('user') && session()->get('user')->id_role
== 2)
                            <a href="{{ url('/wishlist') }}" title="Your Whishlist">
                                <i class="fas fa-heart fa-2x"></i>
                                <span class="number" id="numberOfWishes">0</span>
                            </a>
                        @endif
                    </li>
                    <li>
                        @if (session()->has('user') && session()->get('user')->id_role
== 2)
```

```
                            <a href="{{ url('/cart') }}" title="Your Cart">
                                <i class="fas fa-shopping-cart fa-2x"></i>
                                <span class="number" id="numberInCart">0</span>
                            </a>
                        @endif
                    </li>
                </ul>

            </div>
        </div>
    </div>

    <div class="wrapper row1">
        <header id="header" class="hoc clear">
            <div id="logo" class="fl_left">
                <h1><a href="{{ route('home') }}">Gačanović Academy</a></h1>
            </div>
            <nav id="mainav" class="fl_right">
                <ul class="clear">
                    <li><a href="{{ route('home') }}">Home</a></li>
                    <li><a href="{{ route('courses') }}">Courses</a></li>
                    <li><a href="{{ route('contactPage') }}">Contact</a></li>

                    @if (!session()->has('user'))
                        <li><a href="{{ route('login') }}">Login</a></li>
                    @else
                        @if (session()->has('user') && session()->get('user')->id_role
== 1)
                            <li><a href="{{ route('logs') }}">Admin</a></li>
                            <li><a href="{{ url('/orders') }}">Orders</a></li>
                        @endif
                        @if (session()->has('user') && session()->get('user')->id_role
!== 1)
                            <li><a href="{{ url('/learnings') }}">My Learnings</a></li>
                            <li><a href="{{ url('/instructor') }}">Instructor
Dashboard</a></li>

                        @endif
                        <li><a href="{{ route('logout') }}">Logout</a></li>
                    @endif
                </ul>
            </nav>
        </header>
    </div>
```

4.6.3.2.5 scripts.blade.php

```
<script src="{{ asset('js/jquery.min.js') }}"></script>
<script src="{{ asset('js/jquery.mobilemenu.js') }}"></script>
```

```
<script src="{{ asset('js/main.js') }}"></script>

@if (session()->has('user') && session()->get('user')->id_role == 2)
    <script src="{{ asset('js/cart.js') }}"></script>
    <script src="{{ asset('js/wish.js') }}"></script>
@endif

</body>

</html>
```

4.6.4 layout

4.6.4.1 admin.blade.php

```
<!DOCTYPE html>
<html lang="en">

@include('fixed.admin.head')

<body class="hold-transition sidebar-mini">

    <div class="wrapper">

        @include('fixed.admin.header')
        @include('fixed.admin.sidebar')

        <div class="content-wrapper">
            @yield('content')
        </div>

    </div>

    @include('fixed.admin.footer')

    @yield('scripts')
</body>

</html>
```

4.6.4.2 instructor.blade.php

```
@include('fixed.user.head')

<div class="wrapper">
    <div class="content-wrapper">
```

```
        @yield('content')
    </div>
</div>

@include('fixed.user.footer')

@include('fixed.user.scripts')
```

4.6.4.3 user.blade.php

```
@include('fixed.user.head')

@include('fixed.user.header')
@include('fixed.user.banner')

@yield('content')

@include('fixed.user.footer')

<?php if (strpos($_SERVER['REQUEST_URI'], 'contact') !== false) :?>
    <script src="{{ asset('js/contact.js') }}"></script>
<?php endif ?>

@include('fixed.user.scripts')
```

4.6.5 pages

4.6.5.1 admin

4.6.5.1.1 categories_edit.blade.php

```
@extends('layout.admin')

@php
$arrayInputs = [
    [
        'label' => 'Category',
        'type' => 'text',
        'name' => 'categoryName',
        'placeholder' => 'Enter category name',
        'value' => $category->category_name,
    ],
    [
        'label' => 'Category Image',
        'type' => 'file',
```

```
        'name' => 'categoryImage',
        'folder' => 'categories/',
        'image' => $category->category_image,
    ],
];
@endphp

@section('content')
    @component('components.admin.form', [
        'headTitle' => 'Categories',
        'title' => 'Edit Category',
        'method' => 'PUT',
        'action' => 'categories.update',
        'arrayInputs' => $arrayInputs,
        'id' => $category->id_category,
        'button' => '',
    ])@endcomponent
@endsection
```

4.6.5.1.2 categories_blade.php

```
@extends('layout.admin')

@php
$arrayInputs = [
    [
        'label' => 'Category',
        'type' => 'text',
        'name'  => 'categoryName',
        'placeholder' => 'Enter category name',
    ],
    [
        'label' => 'Category Image',
        'type' => 'file',
        'name' => 'categoryImage'
    ],
];
@endphp

@section('content')
    @component('components.admin.form', [
        'headTitle' => 'Categories',
        'title' => 'Add New Category',
        'method' => 'POST',
        'action' => 'categories.store',
        'arrayInputs' => $arrayInputs,
        'button' => '',
    ]) @endcomponent
@endsection
```

4.6.5.1.3 contact_mails.blade.php

```
@extends('layout.admin')

@section('content')
    <section class="content-header">
        <div class="container-fluid">
            <div class="row mb-2">
                <div class="col-sm-6">
                    <h2>All Contact Mails</h2>
                </div>
            </div>
        </div>
    </section>

    <section class="content">
        <div class="row">
            <div class="col-md-12">
                <table class="table table-responsive" id="table">

                </table>
            </div>
            <div class="col-md-12">
                <div class="js-notification"></div>
                <div class="dataTables_paginate paging_simple_numbers pt-3"
id="table_paginate">

                </div>
            </div>
        </div>
    </section>
@endsection
```

4.6.5.1.4 courses_edit.blade.php

```
@extends(session()->get('user')->id_role == 2 ? 'layout.instructor' : 'layout.admin')

@php
    $arrayInputs = [
        [
            'label' => 'Course',
            'type' => 'text',
            'name' => 'courseName',
            'placeholder' => 'Enter course name',
            'value' => $course->course_name,
```

```
        ],
        [
            'label' => 'Price',
            'type' => 'number',
            'name' => 'coursePrice',
            'placeholder' => 'Enter course price',
            'value' => $course->price,
        ],
        [
            'label' => 'Hours',
            'type' => 'number',
            'name' => 'courseHours',
            'placeholder' => 'Enter total hours',
            'value' => $course->total_hours,
        ],
        [
            'label' => 'Course Image',
            'type' => 'file',
            'name' => 'courseImage',
            'folder' => 'courses/',
            'image' => $course->image_small,
        ],
        [
            'label' => 'Description',
            'type' => 'textarea',
            'name' => 'description',
            'placeholder' => 'Add description',
            'value' => $course->description,
        ],
    ];

    $lessonObject = new stdClass();
    if (!$lessonsEdit->count() == 0) {
        foreach ($lessonsEdit as $l) {
            $lesson[] = ['value' => $l->id_lesson, 'text' => $l->lesson];
        }
        $lessonObject->option = $lesson;
    }


    $lessonObject->type = 'text';
    $lessonObject->name = 'lesson[]';
    $lessonObject->placeholder = 'Enter URL for lesson';
    $lessonObject->label = 'Course Lessons';

    $arrayLessonsEdit = [$lessonObject];

    foreach ($categories as $c) {
        $cat[] = ['value' => $c->id_category, 'text' => $c->category_name];
    }
    $categoryObject = new stdClass();
```

```
    $categoryObject->option = $cat;
    $categoryObject->name = 'category';
    $categoryObject->label = 'Course categories';
    $categoryObject->selected = $course->id_category;

    $arrayDropdowns = [$categoryObject];

    foreach ($topics as $t) {
        $topicsChb[] = ['value' => $t->id_topic, 'text' => $t->topic_name];
    }

    foreach ($course->topics as $ct) {
        $selectedTopics[] = $ct->id_topic;
    }

    $topicObject = new stdClass();
    $topicObject->option = $topicsChb;
    $topicObject->name = 'topicsChb[]';
    $topicObject->label = 'Couse topics';
    $topicObject->selected = $selectedTopics;

    $arrayCheckboxes = [$topicObject];
@endphp

@section('content')
    @if (session()->get('user')->id_role == 1)
        @component('components.admin.form',
            [
                'headTitle' => 'Courses',
                'title' => 'Edit Course',
                'method' => 'PUT',
                'action' => 'courses.update',
                'arrayInputs' => $arrayInputs,
                'arrayLessonsEdit' => $arrayLessonsEdit,
                'arrayDropdowns' => $arrayDropdowns,
                'arrayCheckboxes' => $arrayCheckboxes,
                'id' => $course->id_course,
                'button' => '',
            ])
        @endcomponent
    @else
        @component('components.instructor.form',
            [
                'pageTitleInstructor' => 'Courses - Instructor',
                'formTitleInstructor' => 'Edit Course',
                'methodInstructor' => 'PUT',
                'actionInstructor' => 'instructorUpdateCourse',
                'arrayInputsInstructor' => $arrayInputs,
                'arrayLessonsEdit' => $arrayLessonsEdit,
                'arrayDropdownsInstructor' => $arrayDropdowns,
                'arrayCheckboxesInstructor' => $arrayCheckboxes,
```

```
                'id' => $course->id_course,
                'buttonInstructor' => '',
            ])
        @endcomponent
    @endif
@endsection
```

4.6.5.1.5 courses.blade.php

```
@extends(session()->get('user')->id_role == 2 ? 'layout.instructor' : 'layout.admin')

@php
    $arrayInputs = [
        [
            'label' => 'Course',
            'type' => 'text',
            'name' => 'courseName',
            'placeholder' => 'Enter course name',
        ],
        [
            'label' => 'Price',
            'type' => 'number',
            'name' => 'coursePrice',
            'placeholder' => 'Enter course price',
        ],
        [
            'label' => 'Hours',
            'type' => 'number',
            'name' => 'courseHours',
            'placeholder' => 'Enter total hours',
        ],
        [
            'label' => 'Course Image',
            'type' => 'file',
            'name' => 'courseImage',
        ],
        [
            'label' => 'Description',
            'type' => 'textarea',
            'name' => 'description',
            'placeholder' => 'Enter description',
        ],
    ];

    $lessonObject = new stdClass();
    if (!$lessons->count() == 0) {
        foreach ($lessons as $l) {
            $lesson[] = ['value' => $l->id_lesson, 'text' => $l->lesson];
        }
        $lessonObject->option = $lesson;
```

```
    }
    $lessonObject->type = 'text';
    $lessonObject->name = 'lesson[]';
    $lessonObject->placeholder = 'Enter URL for lesson';
    $lessonObject->label = 'Course Lesson';
    $arrayLessons = [$lessonObject];

    foreach ($categories as $c) {
        $category[] = ['value' => $c->id_category, 'text' => $c->category_name];
    }
    $catObject = new stdClass();
    $catObject->option = $category;
    $catObject->name = 'category';
    $catObject->label = 'Course Category';

    $arrayDropdowns = [$catObject];

    foreach ($topics as $t) {
        $topicsChb[] = ['value' => $t->id_topic, 'text' => $t->topic_name];
    }
    $topicObject = new stdClass();
    $topicObject->option = $topicsChb;
    $topicObject->name = 'topicsChb[]';
    $topicObject->label = 'Course topics';

    $arrayCheckboxes = [$topicObject];

@endphp

@section('content')
    @if (session()->get('user')->id_role == 1)
        @component('components.admin.form',
            [
                'headTitle' => 'Courses',
                'title' => 'Add New Course',
                'method'  =>  'POST',
                'action' => 'courses.store',
                'arrayInputs' => $arrayInputs,
                'arrayLessons' => $arrayLessons,
                'arrayDropdowns' => $arrayDropdowns,
                'arrayCheckboxes' => $arrayCheckboxes,
                'button' => '',
            ])
        @endcomponent
    @else
        @component('components.instructor.form',
            [
                'pageTitleInstructor' => 'Courses - Instructor',
                'formTitleInstructor' => 'Add New Course',
                'methodInstructor'  =>  'POST',
                'actionInstructor' => 'instructorStoreCourse',
```

```
                'arrayInputsInstructor' => $arrayInputs,
                'arrayLessonsInstructor' => $arrayLessons,
                'arrayDropdownsInstructor' => $arrayDropdowns,
                'arrayCheckboxesInstructor' => $arrayCheckboxes,
                'buttonInstructor' => '',
            ])
        @endcomponent
    @endif
@endsection
```

4.6.5.1.6 log.blade.php

```
@extends('layout.admin')

@section('content')
    <div class="container-fluid">
        <div class="row mt-4">
            <div class="col sidebar mb-3">
                <h4 class="mt-1"><i class="fa fa-calendar" aria-hidden="true"></i>
Laravel Log Viewer</h4>
                <p class="text-muted"><i>by Rap2h</i></p>

                <div class="list-group div-scroll">
                    @foreach ($folders as $folder)
                        <div class="list-group-item">
                            <a href="?f={{
\Illuminate\Support\Facades\Crypt::encrypt($folder)  }}">
                                <span class="fa fa-folder"></span> {{ $folder }}
                            </a>
                            @if ($current_folder == $folder)
                                <div class="list-group folder">
                                    @foreach ($folder_files as $file)
                                        <a href="?l={{
\Illuminate\Support\Facades\Crypt::encrypt($file)  }}&f={{
\Illuminate\Support\Facades\Crypt::encrypt($folder)  }}"
                                            class="list-group-item @if
($current_file==$file) llv-active @endif">
                                            {{ $file }}
                                        </a>
                                    @endforeach
                                </div>
                            @endif
                        </div>
                    @endforeach
                    @foreach ($files as $file)
                        <a href="?l={{
\Illuminate\Support\Facades\Crypt::encrypt($file)  }}"
                            class="list-group-item @if ($current_file==$file) llv-
active @endif">
                            {{ $file }}
```

```
                        </a>
                    @endforeach
                </div>
            </div>
            <div class="col-10 table-container">
                @if ($logs === null)
                    <div>
                        Log file >50M, please download it.
                    </div>
                @else
                    <table id="table-log" class="table table-striped" data-ordering-
index="{{ $standardFormat ? 2 : 0 }}">
                        <thead>
                            <tr>
                                @if ($standardFormat)
                                    <th>Level</th>
                                    <th>Context</th>
                                    <th>Date</th>
                                @else
                                    <th>Line number</th>
                                @endif
                                <th>Content</th>
                            </tr>
                        </thead>
                        <tbody>
                            @foreach ($logs as $key => $log)
                                <tr data-display="stack{{ $key }}">
                                    @if ($standardFormat)
                                        <td class="nowrap text-{{ $log['level_class']
}}">
                                            <span class="fa fa-{{ $log['level_img'] }}"
                                                aria-
hidden="true"></span>  {{ $log['level'] }}
                                        </td>
                                        <td class="text">{{ $log['context'] }}</td>
                                    @endif
                                    <td class="date">{{ $log['date'] }}</td>
                                    <td class="text">
                                        @if ($log['stack'])
                                            <button type="button"
                                                class="float-right expand btn btn-
outline-dark btn-sm mb-2 ml-2"
                                                data-display="stack{{ $key }}">
                                                <span class="fa fa-search"></span>
                                            </button>
                                        @endif
                                        {{ $log['text'] }}
                                        @if (isset($log['in_file']))
                                            <br />{{ $log['in_file'] }}
                                        @endif
                                        @if ($log['stack'])
```

```
                                        <div class="stack" id="stack{{ $key }}"
                                            style="display: none; white-space: pre-
wrap;">{{ trim($log['stack']) }}
                                        </div>
                                    @endif
                                </td>
                            </tr>
                        @endforeach
                    </tbody>
                </table>
            @endif
            <div class="p-3">
                @if ($current_file)
                    <a
                        href="?dl={{
\Illuminate\Support\Facades\Crypt::encrypt($current_file) }}{{ $current_folder ? '&f='
. \Illuminate\Support\Facades\Crypt::encrypt($current_folder) : '' }}">
                        <span class="fa fa-download"></span> Download file
                    </a>
                    -
                    <a id="clean-log"
                        href="?clean={{
\Illuminate\Support\Facades\Crypt::encrypt($current_file) }}{{ $current_folder ? '&f='
. \Illuminate\Support\Facades\Crypt::encrypt($current_folder) : '' }}">
                        <span class="fa fa-sync"></span> Clean file
                    </a>
                    -
                    <a id="delete-log"
                        href="?del={{
\Illuminate\Support\Facades\Crypt::encrypt($current_file) }}{{ $current_folder ? '&f='
. \Illuminate\Support\Facades\Crypt::encrypt($current_folder) : '' }}">
                        <span class="fa fa-trash"></span> Delete file
                    </a>
                    @if (count($files) > 1)
                        -
                        <a id="delete-all-log"
                            href="?delall=true{{ $current_folder ? '&f=' .
\Illuminate\Support\Facades\Crypt::encrypt($current_folder) : '' }}">
                            <span class="fa fa-trash-alt"></span> Delete all files
                        </a>
                    @endif
                @endif
            </div>
        </div>
    </div>
    </div>

@endsection

@section('scripts')
```

```
    <script type="text/javascript"
src="https://cdn.datatables.net/1.10.16/js/jquery.dataTables.min.js"></script>
    <script type="text/javascript"
src="https://cdn.datatables.net/1.10.16/js/dataTables.bootstrap4.min.js"></script>
    <script>
        $(document).ready(function() {
            $('.table-container tr').on('click', function() {
                $('#' + $(this).data('display')).toggle();
            });
            $('#table-log').DataTable({
                "order": [$('#table-log').data('orderingIndex'), 'desc'],
                "stateSave": true,
                "stateSaveCallback": function(settings, data) {
                    window.localStorage.setItem("datatable", JSON.stringify(data));
                },
                "stateLoadCallback": function(settings) {
                    var data = JSON.parse(window.localStorage.getItem("datatable"));
                    if (data) data.start = 0;
                    return data;
                }
            });
            $('#delete-log, #clean-log, #delete-all-log').click(function() {
                return confirm('Are you sure?');
            });
        });

        localStorage.setItem('courses', JSON.stringify([]));
    </script>
@endsection
```

4.6.5.1.7 topics_edit.blade.php

```
@extends('layout.admin')

@php
$arrayInputs = [
    [
        'label' => 'Topic',
        'type' => 'text',
        'name' => 'topicName',
        'placeholder' => 'Enter topic name',
        'value' => $topic->topic_name,
    ],
];
@endphp

@section('content')
    @component('components.admin.form', [
        'headTitle' => 'Topics',
        'title' => 'Edit Topic',
```

```
        'method' => 'PUT',
        'action' => 'topics.update',
        'arrayInputs' => $arrayInputs,
        'id' => $topic->id_topic,
        'button' => '',
    ])@endcomponent
@endsection
```

4.6.5.1.8 topics.blade.php

```
@extends('layout.admin')

@php
    $arrayInputs = [
        [
            'label' => 'Topic',
            'type' => 'text',
            'name' => 'topicName',
            'placeholder' => 'Enter topic name'
        ],
    ];
@endphp

@section('content')
    @component('components.admin.form', [
        'headTitle' => 'Topics',
        'title' => 'Add New Topic',
        'method' => 'POST',
        'action' => 'topics.store',
        'arrayInputs' => $arrayInputs,
        'button' => ''
    ])@endcomponent
@endsection
```

4.6.5.1.9 users_edit.blade.php

```
@extends('layout.admin')

@section('content')
     <section class="content-header">
        <div class="container-fluid">
            <div class="row mb-2">
                <div class="col-sm-6">
                    <h1>Users</h1>
                </div>
            </div>
        </div>
```

```
    </section>

    <section class="content">
        <div class="row">
            <div class="col-md-4">
                <div class="card card-primary">
                    <div class="card-header">
                        <h3 class="card-title">Edit User</h3>
                    </div>

                    <div class="card-body">
                        <form action="{{ url("/admin/users/{$user->id_user}") }}"
method="POST" role="form">
                            @csrf
                            @method('PUT')

                            <div class="form-group">
                                <label for="username">Username</label>
                                <input type="text" name="username" id="username"
class="form-control" placeholder="Enter username" value="{{ $user->username }}" />
                            </div>
                            <div class="form-group">
                                <label for="email">Email</label>
                                <input type="text" name="email" id="email" class="form-
control" placeholder="Enter email" value="{{ $user->email }}" />
                            </div>
                            <div class="form-group">
                                <label for="password">Password</label>
                                <input type="text" name="password" id="password"
class="form-control" placeholder="Enter password" />
                            </div>
                            <div class="form-group">
                                <label for="active">Active</label> <br/>
                                <input type="checkbox" name="active" id="active"
value="1" @if ($user->active == 1) checked @endif>
                            </div>
                            <div class="form-group">
                                <label for="isInstructor">Is Instructor</label> <br/>
                                <input type="checkbox" name="isInstructor"
id="isInstructor" value="1" @if ($user->is_instructor == 1) checked @endif>
                            </div>
                            <div class="form-group">
                                <label for="role">Role</label>
                                <select class="form-control" name="role" id="role">
                                    <option value="0">Choose</option>
                                    @foreach ($roles as $role)
                                        @if ($user->id_role == $role->id_role)
                                            <option selected value="{{ $role->id_role
}}">{{ $role->role_name }}</option>
                                        @else
```

```
                                        <option value="{{ $role->id_role }}">{{
$role->role_name }}</option>
                                    @endif
                                @endforeach
                            </select>
                        </div>

                        <input type="hidden" name="hiddenUserField" value="{{
$user->id_user }}">

                        <div class="form-group">
                            <button type="submit" class="btn btn-primary"
id="EditUser">Edit User</button>
                        </div>
                    </form>
                </div>
                <div class="card-footer">
                    <div class="js-notification"></div>
                    @if ($errors->any())
                        <div class="alert alert-danger pr mt-3">
                            <ul>
                                @foreach ($errors->all() as $error)
                                    <li>{{ $error }}</li>
                                @endforeach
                            </ul>
                        </div>
                    @endif
                    @if (session('success'))
                        <div class="alert alert-success mt-3">
                            {{ session('success') }}
                        </div>
                    @endif
                    @if (session('error'))
                        <div class="alert alert-danger mt-3">
                            {{ session('error') }}
                        </div>
                    @endif
                </div>
            </div>
        </div>

        <div class="col-md-8">
            <div class="card">
                <div class="card-body">
                    <table id="table" class="table table-bordered table-hover
table-responsive">

                    </table>
                    <div class="dataTables_paginate paging_simple_numbers"
id="table_paginate">
```

```
                        </div>
                        <div class="msgCrud">
                            @if (session()->has('updateMsg'))
                                {{ session()->get('updateMsg') }}
                            @endif
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>
@endsection
```

4.6.5.1.10 users.blade.php

```
@extends('layout.admin')
@section('content')
    <section class="content-header">
        <div class="container-fluid">
            <div class="row mb-2">
                <div class="col-sm-6">
                    <h1>Users</h1>
                </div>
            </div>
        </div>
    </section>

    <section class="content">
        <div class="row">
            <div class="col-md-4">
                <div class="card card-primary">
                    <div class="card-header">
                        <h3 class="card-title">Add New User</h3>
                    </div>

                    <div class="card-body">
                        <form action="{{ url('/admin/users') }}" method="POST"
role="form">
                            @csrf

                            <div class="form-group">
                                <label for="username">Username</label>
                                <input type="text" name="username" id="username"
class="form-control" placeholder="Enter username" />
                            </div>
                            <div class="form-group">
                                <label for="email">Email</label>
                                <input type="text" name="email" id="email" class="form-
control" placeholder="Enter email" />
                            </div>
```

```
                            <div class="form-group">
                                <label for="password">Password</label>
                                <input type="password" name="password" id="password"
class="form-control" placeholder="Enter password" />
                            </div>
                            <div class="form-group">
                                <label for="confirmPassword">Repeat password</label>
                                <input type="password" name="confirmPassword"
id="confirmPassword" class="form-control" placeholder="Repeat password" />
                            </div>
                            <div class="form-group">
                                <label for="role">Role</label>
                                <select class="form-control" name="role" id="role">
                                    <option value="0">Choose</option>
                                    @foreach ($roles as $role)
                                        <option value="{{ $role->id_role }}">{{ $role-
>role_name }}</option>
                                    @endforeach
                                </select>
                            </div>
                            <div class="form-group">
                                <button type="button" class="btn btn-primary"
id="AddUser">Add User</button>
                            </div>
                        </form>
                    </div>
                    <div class="card-footer">
                        <div class="js-notification"></div>
                        @if ($errors->any())
                            <div class="alert alert-danger pr mt-3">
                                <ul>
                                    @foreach ($errors->all() as $error)
                                        <li>{{ $error }}</li>
                                    @endforeach
                                </ul>
                            </div>
                        @endif
                        @if (session('success'))
                            <div class="alert alert-success mt-3">
                                {{ session('success') }}
                            </div>
                        @endif
                        @if (session('error'))
                            <div class="alert alert-danger mt-3">
                                {{ session('error') }}
                            </div>
                        @endif
                    </div>
                </div>
            </div>
```

```
            <div class="col-md-8">
                <div class="card">
                    <div class="card-body">
                        <table id="table" class="table table-bordered table-hover
table-responsive">

                        </table>
                        <div class="dataTables_paginate paging_simple_numbers"
id="table_paginate">

                        </div>
                        <div class="msgCrud">
                            @if (session()->has('updateMsg'))
                                {{ session()->get('updateMsg') }}
                            @endif
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>
@endsection
```

4.6.5.2 Intructor

4.6.5.2.1 instructor.blade.php

```
@extends('layout.instructor')

@section('content')
    <div class="header-instructor">
        <div class="row">
            <div class="col-6">
                <img src="{{ asset('/img/logo-learnings.png') }}" class="img-fluid" />
            </div>
            <div class="col-6 flex-link">
                <a class="exit-url" href="{{ url('/') }}">Exit</a>
            </div>
        </div>
    </div>
    <div class="container">
        <div class="row">
            <div class="col-12 center-text-instructor">
                <h2 class="section-title mb-4" id="poll-heading">Please Select Answer
to Start Teaching.</h2>
            </div>
        </div>
    </div>
```

```
    <div class="container wp">
        <div class="row">
            <div class="col-12">
                <form action="" method="PUT" role="form" id="poll-instructor">
                    <p>What kind of teaching have you done before?</p>

                    @foreach ($answers as $ans)
                        <label class="form-control">
                            <input type="radio" name="teachingAns" value="{{ $ans-
>id_answer }}" /> {{ $ans->answer }}
                        </label>
                    @endforeach

                    <input type="hidden" name="idUser" id="hdnIdUsr" value="{{ $user-
>id_user }}">
                    <button type="button" id="btnVote" class="btn btn-success mt-30 mb-
50">Become a Instructor</button>
                </form>

                <div id="votingRes">

                </div>

                <a href="" id="btnCreateCourse" class="btn btn-primary mt-30 mb-50">Go
To Create Your Course!</a>
            </div>
        </div>
    </div>
@endsection
```

4.6.5.3 user

4.6.5.3.1 author.blade.php

```
@extends('layout.user')

@section('title')
    Author
@endsection

@section('content')
    <div class="container">
        <h2 class="section-title mb-4">Author</h2>
        <div class="row">
            <div class="col-6">
                <div id="author_image">
```

```
                    <img src="{{ asset('img/marko.jpg') }}" alt="author" />
                </div>
            </div>
            <div class="col-6">
                <div id="author_info">
                    <p class="textAurhor">My name is Marko Gačanović, 25 years old</p>
                    <p class="college">
                        Student of ICT College of Vocational Studies, Information
Technology course. <br/><br/>
                        Index number : 38 / 17
                    </p>
                </div>
            </div>
        </div>
    </div>
@endsection
```

4.6.5.3.2 cart.blade.php

```
@extends('layout.user')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-12">
                <h2 class="section-title mb-4">Cart</h2>
                <form action="#">
                    <table class="table table-bordered table-cart">
                        <thead>
                            <tr>
                                <th class="pro-thumbnail">Image</th>
                                <th class="pro-title">Course</th>
                                <th class="pro-price">Price</th>
                                <th class="pro-hours">Total Hours</th>
                                <th class="">Remove</th>
                            </tr>
                        </thead>
                        <tbody id="cart">

                        </tbody>
                        <tfoot>

                        </tfoot>
                    </table>
                </form>
            </div>
        </div>

        <div class="row">
            <div class="col-10">
```

```
                <form action="{{ route('checkout') }}" method="POST">
                    @csrf

                    <input type="hidden" name="cartItems" id="cartItems">

                    <button class="checkout-btn mt-90">Checkout</button>
                </form>
            </div>
        </div>
    </div>
@endsection
```

4.6.5.3.3 checkout-cancel.blade.php

```
<html>

<head>
    <link
href="https://fonts.googleapis.com/css?family=Nunito+Sans:400,400i,700,900&display=swap
" rel="stylesheet">
    <link rel='stylesheet' href='https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css'>
</head>
<style>
    body {
        text-align: center;
        padding: 40px 0;
        background: #EBF0F5;
    }

    h1 {
        color: #d12323;
        font-family: "Nunito Sans", "Helvetica Neue", sans-serif;
        font-weight: 900;
        font-size: 40px;
        margin-bottom: 10px;
    }

    p {
        color: #404F5E;
        font-family: "Nunito Sans", "Helvetica Neue", sans-serif;
        font-size: 20px;
        margin-bottom: 20px;
    }

    i {
        color: #d12323;
        font-size: 100px;
        line-height: 200px;
        margin-left: -15px;
```

```
    }

    .card {
        background: white;
        padding: 60px;
        border-radius: 4px;
        box-shadow: 0 2px 3px #C8D0D8;
        display: inline-block;
        margin: 0 auto;
    }

    .btn-go-home {
        outline: none;
        font-size: 18px;
        height: auto;
        background: #88B04B;
        color: #fff;
        text-decoration: none;
        font-weight: normal;
        padding: 14px;
        border-radius: 2px;
    }
    .btn-go-home:hover {
        background: #b7d48c;
    }

</style>

<body>
    <div class="card">
        <div style="border-radius:200px; height:200px; width:200px; background:
#F8FAF5; margin:0 auto;">
            <i style="font-size: 70px; margin-top: 57px; margin-left: -2px;" class="fa
fa-close"></i>
        </div>
        <h1>Cancelled</h1>
        <p>We coudn't receive your order!</p>

        <a class="btn-go-home" href="{{ route('home') }}">Go Back To Home Page</a>
    </div>
</body>

</html>
```

4.6.5.3.4 checkout-success.blade.php

```
<html>

<head>
```

```
    <link
href="https://fonts.googleapis.com/css?family=Nunito+Sans:400,400i,700,900&display=swap
" rel="stylesheet">
</head>
<style>
    body {
        text-align: center;
        padding: 40px 0;
        background: #EBF0F5;
    }

    h1 {
        color: #88B04B;
        font-family: "Nunito Sans", "Helvetica Neue", sans-serif;
        font-weight: 900;
        font-size: 40px;
        margin-bottom: 10px;
    }

    p {
        color: #404F5E;
        font-family: "Nunito Sans", "Helvetica Neue", sans-serif;
        font-size: 20px;
        margin-bottom: 20px;
    }

    i {
        color: #9ABC66;
        font-size: 100px;
        line-height: 200px;
        margin-left: -15px;
    }

    .card {
        background: white;
        padding: 60px;
        border-radius: 4px;
        box-shadow: 0 2px 3px #C8D0D8;
        display: inline-block;
        margin: 0 auto;
    }

    .btn-go-home {
        outline: none;
        font-size: 18px;
        height: auto;
        background: #88B04B;
        color: #fff;
        text-decoration: none;
        font-weight: normal;
        padding: 14px;
```

```
        border-radius: 2px;
    }
    .btn-go-home:hover {
        background: #b7d48c;
    }

</style>

<body>
    <div class="card">
        <div style="border-radius:200px; height:200px; width:200px; background:
#F8FAF5; margin:0 auto;">
            <i class="checkmark">✓</i>
        </div>
        <h1>Success</h1>
        <p>We received your purchase request;<br /> thanks for learning from us!</p>
        <a class="btn-go-home" href="{{ route('home') }}">Go Back To Home Page</a>
        <a class="btn-go-home" href="{{ url('/learnings') }}">Check My Courses</a>
    </div>
</body>

<script>
    localStorage.setItem('courses', JSON.stringify([]));
</script>
</html>
```

4.6.5.3.5 contact.blade.php

```
@extends('layout.user')

@section('title')
    Contact
@endsection

@section('content')
    <div class="wrapper row3">
        <main class="hoc container clear">
            <div class="content">
                <div id="comments">

                    <h2>Write To Us</h2>
                    <div id="contact_form">
                        @csrf
                        <div class="one_third first">
                            <label for="email">Mail <span>*</span></label>
                            @if (session()->has('user'))
                                <input type="email" name="email" id="email" value="{{
session()->get('user')->email }}" size="22" required>
                                <small id="emailHelp" class="form-text text-
danger"></small>
```

```
                            @else
                                <input type="email" name="email" id="email" value=""
size="22" required>
                                <small id="emailHelp" class="form-text text-
danger"></small>
                            @endif
                        </div>
                        <div class="one_third">
                            <label for="subject">Subject <span>*</span></label>
                            <input type="text" name="subject" id="subject" value=""
size="22" required>
                            <small id="subjectHelp" class="form-text text-
danger"></small>
                        </div>
                        <div class="block clear">
                            <label for="message">Your Message</label>
                            <textarea name="message" id="message" cols="25"
rows="10"></textarea>
                            <small id="messageHelp" class="form-text text-
danger"></small>
                        </div>
                        <div>
                            <input type="submit" name="submit" id="submitContact"
value="Submit Form">
                             
                            <input type="reset" name="reset" id="resetContact"
value="Reset Form">
                        </div>
                    </div>

                    <div id="successMessage" class="alert invisible" role="alert">
                        <p class="text-center font-weight-light" id="msg"></p>
                    </div>

                    @if ($errors->any())
                        <div class="alert alert-danger">
                            <ul>
                                @foreach ($errors->all() as $error)
                                    <li>{{ $error }}</li>
                                @endforeach
                            </ul>
                        </div>
                    @endif

                    @if (session()->has('error'))
                        {{ session()->get('error') }}
                    @endif

                    @if (session()->has('message'))
                        {{ session()->get('message') }}
                    @endif
```

```
                </div>
            </div>
            <div class="clear"></div>
        </main>
    </div>
@endsection
```

4.6.5.3.6 courses.blade.php

```
@extends('layout.user')

@section('title')
    Courses
@endsection

@section('content')
    <section class="product-section section mt-30 mb-40">
        <div class="container">
            <form action="{{ route('courses') }}" method="GET">
                @csrf
                <div class="row">
                    <div class="col-xl-9 col-lg-8 col-12 mb-50 order-lg-2 ">
                        <div class="row mb-50 pl-br">
                            <div class="col pl-br">
                                <div class="shop-top-bar with-sidebar ">
                                    <div class="product-sort">
                                        <p>Sort by:</p>
                                        <select name="sort" class="form-control">
                                            @php
                                                $sort = [['value' => 'date', 'title' =>
'Newest Courses'], ['value' => 'priceAsc', 'title' => 'Price: low to high'], ['value'
=> 'priceDesc', 'title' => 'Price: high to low']];
                                            @endphp
                                            @foreach ($sort as $s)
                                                @if ($s['value'] == $sort)
                                                    <option selected value="{{
$s['value'] }}">{{ $s['title'] }}

                                                    </option>
                                                @else
                                                    <option value="{{ $s['value']
}}">{{ $s['title'] }}</option>

                                                @endif
                                            @endforeach
                                        </select>
                                    </div>
                                    <div class="product-search">
                                        <p>Search:</p>
                                        <footer>
                                            <form class="example" action="">
```

```
                                        <input type="text" placeholder="Search
For Courses" aria-label="true"
                                            name="search" class="form-control"
value="{{ $search }}">
                                    </form>
                                </footer>
                            </div>
                            <div class="product-showing">
                                <p>Showing:</p>
                                <select name="showing" id="showing"
class="form-control">
                                    @php
                                        $show = [['value' => '4'], ['value' =>
'6'], ['value' => '9'], ['value' => '12'], ['value' => '15']];
                                    @endphp
                                    @foreach ($show as $s)
                                        @if ($s['value'] == $showing)
                                            <option selected value="{{
$s['value'] }}">{{ $s['value'] }}
                                            </option>
                                        @else
                                            <option value="{{ $s['value']
}}">{{ $s['value'] }}</option>
                                        @endif
                                    @endforeach
                                </select>

                            </div>
                            <div class="">
                                <p>Pages: {{ $courses->currentPage() }} of {{
$courses->lastPage() }}</p>

                                <button type="submit" class="buttonS"><i
class="fa fa-search"></i></button>
                            </div>
                        </div>
                    </div>
                </div>

                <div class="shop-product-wrap grid with-sidebar row">
                        @foreach ($courses as $course)
                            @if (session()->has('user') && count($myLearnings)
> 0)
                                @foreach ($myLearnings as $learning)
                                    <?php
                                    $ids_learning_courses[] = $learning-
>id_course;

                                    ?>
                                @endforeach
                            @endif
```

```
                                    @if (session()->has('user') && session()-
>get('user')->id_role == 2 && count($myLearnings) > 0)
                                        @component('components.single_course',
                                            [
                                                'course' => $course,
                                                'ids_learning_courses' =>
$ids_learning_courses,
                                            ])
                                        @endcomponent
                                    @else
                                        @component('components.single_course',
                                            [
                                                'course' => $course,
                                            ])
                                        @endcomponent
                                    @endif
                                @endforeach


                            @if (!count($courses))
                                <h2>No courses for specific request. Try another.</h2>
                            @endif
                        </div>
                        <div class="row mt-30">
                            <div class="col">
                                {{ $courses->links('pagination::bootstrap-4') }}
                            </div>
                        </div>
                    </div>

                    <div class="shop-sidebar-wrap col-xl-3 col-lg-4 col-12 order-lg-1">
                        <div class="shop-sidebar">
                            <h4 class="text-center pb-5 pt-2">TOPIC</h4>
                            @foreach ($topics as $topic)
                                <div class="col-12 mb-15 position-relative">
                                    @if (in_array($topic->id_topic, $topicChb ?? []))
                                        <input type="checkbox" name="topic[]"
id="remember_me"
                                            value="{{ $topic->id_topic }}" checked />
                                        <label for="remember_me">{{ $topic->topic_name
}}</label>
                                    @else
                                        <input type="checkbox" name="topic[]"
id="remember_me"
                                            value="{{ $topic->id_topic }}" />
                                        <label for="remember_me">{{ $topic->topic_name
}}</label>
                                    @endif
                                </div>
                            @endforeach
                        </div>
```

```
                        <div class="shop-sidebar">
                            <h4 class="text-center pb-5 pt-5">CATEGORY</h4>
                            @foreach ($categories as $c)
                                <div class="col-12 mb-15 position-relative">
                                    @if (in_array($c->id_category, $categoriesChb ??
[]))
                                        <input type="checkbox" name="categories[]"
id="remember_me" checked="checked"
                                            value="{{ $c->id_category }}">
                                        <label for="remember_me">{{ $c->category_name
}}</label>
                                    @else
                                        <input type="checkbox" name="categories[]"
id="remember_me"
                                            value="{{ $c->id_category }}">
                                        <label for="remember_me">{{ $c->category_name
}}</label>
                                    @endif
                                </div>
                            @endforeach
                        </div>
                    </div>
                </div>
            </form>
        </div>
    </section>
@endsection
```

4.6.5.3.7 home_blade.php

```
@extends('layout.user')

@section('title')
    Home
@endsection

@section('content')
    <div class="wrapper row2">
        <section class="hoc container clear">
            <div class="sectiontitle fl_left">
                <h2 class="subtitle">Course Categories</h2>
            </div>
            <ul class="nospace group prices">
                @foreach ($categories as $c)
                    <li class="one_third">
                        <article>
                            <div class="image-cat">
                                <img src="{{ asset("img/categories/" . $c-
>category_image) }}" alt="{{ $c->category_name }}" />
```

```
                            </div>
                            <h6 class="heading">
                                <a href="{{ route('courses', ['categories[]' => $c-
>id_category]) }}">{{ $c->category_name }}</a>
                            </h6>
                            <p>From: <sup></sup><strong>{{ $c->min_price
}}</strong><em>&euro;</em></p>
                        </article>
                    </li>
                @endforeach
            </ul>
        </section>
    </div>

    <div class="wrapper coloured">
        <article class="hoc cta clear">
            <h6 class="three_quarter first subscribeTitle">SUBSCRIBE OUR
NEWSLETTER</h6>
            <footer class="one_quarter">
                <form class="example" action="">
                    <input class="enterEmailPlaceholder" type="text" placeholder="Enter
Your Email" aria-label="true" name="search">
                    <button type="submit">SUB</button>
                </form>
            </footer>
        </article>
    </div>
@endsection
```

4.6.5.3.8 learnings.blade.php

```
@extends('layout.user')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-12">
                <h2 class="section-title mb-4">My Learnings</h2>

                @if (session()->get('user')->id_role == 2)
                    @if ($myLearnings->isEmpty())
                        <h2 class="text-danger">No courses in your learning
section.</h2>
                    @else
                        <div class="row">
                            @foreach ($myLearnings as $learning)
                                <div class="col-3">
                                    <input type="hidden" name="hiddenIdCourse" id="{{
$learning->id_course }}" value="{{ $learning->id_course }}">
```

```
                                    <div class="image-course-info">
                                        <img src="{{ asset('/img/courses/' . $learning-
>image_small) }}" class="learning-img"/>
                                    </div>
                                    <div class="headings-course-info">
                                        <p><span>Bought At: </span> {{ $learning-
>bought_at }}</p>
                                        <p>{{ $learning->description }}</p>
                                        <p><span>Price: </span> {{ $learning->price }}
&euro;</p>
                                        <p><span>Author: </span> {{ $learning->author
}}</p>
                                    </div>
                                    <div class="stats-course-info">
                                        <p class="lessons-heading"><span>Lessons:
</span> </p>
                                        @foreach ($lessons as $lesson)
                                            @if ($lesson->id_course == $learning-
>id_course)
                                                <a href="{{ $lesson->lesson }}"
target="_blank"> <p>- {{ $lesson->lesson }}</p> </a>
                                            @endif
                                        @endforeach
                                    </div>
                                </div>
                            @endforeach
                        </div>
                    @endif
                @endif

            </div>
        </div>
    </div>
@endsection
```

4.6.5.3.9 login.blade.php

```
@extends('layout.user')

@section('title')
Login
@endsection

@section('content')
    <div class="row">
        <div class="col-xs-12 col-lg-6 mx-auto" id="loginReg">
            <div class="panel panel-login">
                <div class="panel-heading">
                    <div class="row">
                        <div class="col-xs-6">
```

```
                            <a href="#" class="active" id="login-form-link">Login</a>
                        </div>
                        <div class="col-xs-6">
                            <a href="#" id="register-form-link">Register</a>
                        </div>
                    </div>

                    <hr/>
                </div>

                <div class="panel-body">
                    <div class="row">
                        <div class="col-lg-12">
                            <form id="login-form" action="{{ route('doLogin') }}"
method="post">
                                @csrf

                                <div class="form-group">
                                    <input type="text" name="username" id="username"
class="form-control" placeholder="Username">
                                </div>
                                <div class="form-group">
                                    <input type="password" name="password"
id="password" class="form-control" placeholder="Password">
                                </div>

                                <div class="form-group">
                                    <div class="row">
                                        <div class="col-sm-6 col-sm-offset-3 mx-auto">
                                            <input type="submit" name="login-submit"
id="login-submit" class="form-control authBtn btn-login" value="Login">
                                        </div>
                                    </div>
                                </div>
                            </form>

                            <form id="register-form" action="" method="post"
style="display: none;">
                                @csrf

                                <div class="form-group">
                                    <input type="email" name="emailReg" id="emailReg"
class="form-control" placeholder="Email">
                                </div>
                                <div class="form-group">
                                    <input type="text" name="usernameReg"
id="usernameReg" class="form-control" placeholder="Username">
                                </div>
                                <div class="form-group">
                                    <input type="password" name="passwordReg"
id="passwordReg" class="form-control" placeholder="Password">
```

```
                                </div>
                                <div class="form-group">
                                    <input type="password" name="passwordRegConf"
id="passwordRegConf" class="form-control" placeholder="Repeat password">
                                </div>

                                <div class="form-group">
                                    <div class="row">
                                        <div class="col-sm-6 col-sm-offset-3 mx-auto ">
                                            <input type="button" name="register-submit"
id="register-submit" class="form-control authBtn btn-register" value="Register">
                                        </div>
                                    </div>
                                </div>
                            </form>

                            <div id="notification">
                                @if ($errors->any())
                                    <div class="alert alert-danger">
                                        <ul>
                                            @foreach($errors->all() as $error)
                                                <li>{{ $error }}</li>
                                            @endforeach
                                        </ul>
                                    </div>
                                @endif

                                @if(session()->has('error'))
                                     {{ session()->get('error') }}
                                @endif
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
@endsection
```

4.6.5.3.10 orders_blade.php

```
@extends('layout.user')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-12">
                <h2 class="section-title mb-4">Orders</h2>
                <form action="#">
                    <table class="table table-bordered">
```

```
                        @if (session()->get('user')->id_role == 1)
                            @if ($ordersAdmin->isEmpty())
                                <h2 class="text-danger">No orders.</h2>
                            @else
                                <thead>
                                    <tr>
                                        <th class="pro-thumbnail">Image</th>
                                        <th class="pro-title">Course</th>
                                        <th class="pro-price">Price</th>
                                        <th class="pro-hours">Total Hours</th>
                                        <th class="pro-date">Order Date</th>
                                        <th class="pro-username">Username</th>
                                    </tr>
                                </thead>
                                <tbody id="order-table">
                                    @foreach ($ordersAdmin as $oa)
                                        <tr>
                                            <td><img src="{{ asset('img/courses/' .
$oa->image_small) }}" alt="{{ $oa->course_name }}"></td>
                                            <td>{{ $oa->course_name }}</td>
                                            <td>{{ $oa->price }} &euro;</td>
                                            <td>{{ $oa->total_hours }}</td>
                                            <td>{{ $oa->orderd_at }}</td>
                                            <td>{{ $oa->username }}</td>
                                        </tr>
                                    @endforeach
                                </tbody>
                            @endif
                        @endif
                    </table>
                </form>
                @if (session()->has('user') && session()->get('user')->id_role == 1)
                    {{ $ordersAdmin->links('pagination::bootstrap-4') }}
                @endif
            </div>
        </div>
    </div>
@endsection
```

4.6.5.3.11 single_course.blade.php

```
@extends('layout.user')

@section('title')
@endsection

@section('content')
    @if (session()->has('user'))
        @foreach ($myLearnings as $learning)
            <?php
```

```
                $ids_learning_courses[] = $learning->id_course;
            ?>
        @endforeach
    @endif
    <section class="content">
        <div class="row">
            <div class="col-md-6 col-12 text-center mx-auto my-5">
                <div class="single-image">
                    <img src="{{ asset('/img/courses/' . $course->image_big) }}"
class="img-fluid" alt="{{ $course->course_name }}">
                </div>
            </div>
            <div class="col-md-6 col-12 text-md-left text-center">
                <div class="contentT">
                    <div class="titles">
                        <p><span>Category :</span> <span class="cat">{{ $course-
>category_name }}</span></p>
                        <h2>{{ $course->course_name }}</h2>
                    </div>
                    <div class="numbers py-5">
                        <h4 class="price">Price: {{ $course->price }} &euro;</h4>
                        <h4 class="hours">Total hours: {{ floatval($course-
>total_hours) }}</h4>
                        <h4 class="py-2">Author: {{ $course->author }}</h4>
                    </div>

                    <div class="actions">
                        @if (session()->has('user'))
                            @if (session()->get('user')->id_user == 1)
                                <h4 class="text-warning">You must be logged in as User
if You want to buy course.</h4>
                            @else
                                @if (isset($ids_learning_courses))
                                    @if (in_array($course->id_course,
$ids_learning_courses))
                                        <h4>You already bought this course.</h4>
                                    @else
                                        <a href="javascript:void(0)" class="add-course-
to-cart" data-idcourse="{{ $course->id_course }}">
                                            <i class="fas fa-shopping-cart fa-2x clr
"></i><span>Add To Cart</span>
                                        </a>
                                        <a href="javascript:void(0)" class="wishhhh"
data-idcourse="{{ $course->id_course }}">
                                            <i class="fas fa-heart fa-2x clr
"></i><span>Add To Whishlist</span>
                                        </a>
                                    @endif
                                @else
                                    <a href="javascript:void(0)" class="add-course-to-
cart" data-idcourse="{{ $course->id_course }}">
```

```
                                        <i class="fas fa-shopping-cart fa-2x clr
"></i><span>Add To Cart</span>
                                    </a>
                                    <a href="javascript:void(0)" class="wishhhh" data-
idcourse="{{ $course->id_course }}">
                                        <i class="fas fa-heart fa-2x clr
"></i><span>Add To Whishlist</span>
                                    </a>
                                @endif
                            @endif
                        @else
                            <h4 class="text-warning">You must be logged in if You want
to buy course.</h4>
                        @endif
                    </div>
                </div>
            </div>
        </div>
    </section>
@endsection
```

4.6.5.3.12 wishes.blade.php

```
@extends('layout.user')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-12">
                 <h2 class="section-title mb-4">Wishes</h2>
                <form action="#">
                    <table class="table table-bordered table-wish">
                        <thead>
                            <tr>
                                <th class="pro-thumbnail">Image</th>
                                <th class="pro-title">Course</th>
                                <th class="pro-price">Price</th>
                                <th class="pro-subtotal">Visit</th>
                                <th class="pro-remove">Remove</th>
                            </tr>
                        </thead>
                        <tbody id="wishlist">

                        </tbody>
                        <tfoot>

                        </tfoot>
                    </table>
                </form>
            </div>
```

```
        </div>
    </div>
@endsection
```

4.7 Routes

4.7.1 web.php

```
<?php

use App\Http\Controllers\Admin\CategoryController;
use App\Http\Controllers\Admin\CourseController;
use App\Http\Controllers\Admin\TopicController;
use App\Http\Controllers\Admin\UserController;
use App\Http\Controllers\Admin\ContactController as AdminContactController;

use App\Http\Controllers\AuthController;
use App\Http\Controllers\CartController;
use App\Http\Controllers\ContactController;
use App\Http\Controllers\FrontController;
use App\Http\Controllers\InstructorController;
use App\Http\Controllers\LessonController;
use App\Http\Controllers\WishController;
use App\Http\Controllers\CheckoutController;

use Illuminate\Support\Facades\Route;
use Rap2hpoutre\LaravelLogViewer\LogViewerController;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/

Route::pattern('id', '^[0-9]+$');

Route::get('/', [FrontController::class, 'homePage'])->name('home')-
>middleware('RecordAccessToPage');
Route::get('/courses', [FrontController::class, 'coursesPage'])->name('courses')-
>middleware('RecordAccessToPage');
Route::get('/courses/{id}', [FrontController::class, 'singleCoursePage'])-
>middleware('RecordAccessToPage');
```

```
Route::get('/login', [FrontController::class, 'loginPage'])->name('login')-
>middleware('RecordAccessToPage');
Route::post('/login', [AuthController::class, 'doLogin'])->name('doLogin');
Route::get('/logout', [AuthController::class, 'doLogout'])->name('logout');
Route::post('/register', [AuthController::class, 'doRegister'])->name('register');

Route::get('/contact', [FrontController::class, 'contactPage'])->name('contactPage')-
>middleware('RecordAccessToPage');
Route::post('/contact', [ContactController::class, 'store']);

Route::get('/author', [FrontController::class, 'authorPage'])->name('author')-
>middleware('RecordAccessToPage');

Route::get('/orders', [FrontController::class, 'ordersPage'])-
>middleware('RecordAccessToPage');


Route::group(['middleware' => ['AuthoriseLogin']], function () {
    Route::get('/wishlist', [FrontController::class, 'wishesPage'])-
>middleware('RecordAccessToPage');

    Route::get('/cart', [FrontController::class, 'cartPage'])-
>middleware('RecordAccessToPage');

    Route::get('/checkout', [FrontController::class, 'checkoutPage'])-
>middleware('RecordAccessToPage');
    Route::post('/checkout', [CheckoutController::class, 'checkout'])-
>middleware('RecordAccessToPage')->name('checkout');
    Route::get('/success', [CheckoutController::class, 'success'])-
>middleware('RecordAccessToPage')->name('checkout.success');
    Route::get('/cancel', [CheckoutController::class, 'cancel'])-
>middleware('RecordAccessToPage')->name('checkout.cancel');

    Route::get('/learnings', [FrontController::class, 'learningsPage'])-
>middleware('RecordAccessToPage');

    Route::get('/instructor', [FrontController::class, 'instructorPage'])-
>middleware('RecordAccessToPage');

    Route::get('/create', [InstructorController::class, 'instructorPage']);
    Route::get('/index', [InstructorController::class, 'index'])-
>name('istructorIndex');
    Route::post('/store', [InstructorController::class, 'store'])-
>name('instructorStoreCourse');
    Route::get('instructor/{id}/edit', [InstructorController::class, 'edit'])-
>name('insturctorEdit');
    Route::put('/update/{id}', [InstructorController::class, 'update'])-
>name('instructorUpdateCourse');
    Route::delete('/destroy/{id}', [InstructorController::class, 'destroy']);
});
```

```
Route::post('/webhook', [CheckoutController::class, 'webhook'])-
>name('checkout.webhook');

Route::get('/cart/showCourses', [CartController::class, 'getCoursesForCart']);

Route::group(['middleware' => ['Authorise404']], function () {
    Route::prefix('/api')->group(function () {
        Route::get('numberOfWishes', [WishController::class, 'numberOfWishes']);
        Route::get('/wishlist', [WishController::class, 'getAllWishesForOneUser']);
        Route::post('/addWish', [WishController::class, 'addNewWish']);
        Route::delete('/deleteWish', [WishController::class, 'deleteWish']);

        Route::post('/vote', [InstructorController::class, 'vote']);
    });
});

Route::group(['middleware' => ['Admin']], function () {
    Route::prefix('/admin')->group(function () {
        Route::resources([
            '/courses' => CourseController::class,
            '/categories' => CategoryController::class,
            '/lesson' => LessonController::class,
            '/topics' => TopicController::class,
            '/users' => UserController::class,
            '/contact' => AdminContactController::class,
        ]);

        Route::get('/logs', [LogViewerController::class, 'index'])->name('logs');
    });
});
```

4.8 JavaScript

4.8.1 main.js

```
const baseURL = 'http://localhost:8000/'; // doesn't work for 127.0.0.1
// const baseURL = 'http://127.0.0.1:8000/'

const regExpLoginRegister = new RegExp(`^${baseURL}login[\#]?$`);

const regExpAdminCourses = new RegExp(`^${baseURL}admin/courses/create[\#]?$`);
const regExpAdminCategories = new RegExp(`^${baseURL}admin/categories/create[\#]?$`);
const regExpAdminTopics = new RegExp(`^${baseURL}admin/topics/create[\#]?$`);
const regExpAdminUsers = new RegExp(`^${baseURL}admin/users/create[\#]?$`);
const regExpAdminMails = new RegExp(`^${baseURL}admin/contact/create[\#]?$`);
```

```
const regExpInstructor = new RegExp(`^${baseURL}instructor[\#]?$`);
const regExpInstructorEditCourse = new RegExp(`^${baseURL}instructor\/[0-
9]+\/edit[\#]?$`);

const regExpEditTopics = new RegExp(`^${baseURL}admin\/topics\/[0-9]+\/edit[\#]?$`);
const regExpEditCategories = new RegExp(`^${baseURL}admin\/categories\/[0-
9]+\/edit[\#]?$`);

const regExpEditCourses = new RegExp(`^${baseURL}admin\/courses\/[0-9]+\/edit[\#]?$`);

const regExpEditUsers = new RegExp(`^${baseURL}admin\/users\/[0-9]+\/edit[\#]?$`);

$.ajaxSetup({
    headers: {
        'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
    }
});

$(document).ready(function() {
    $(document).on('click', '.js-pagination', function(e) {
        e.preventDefault();

        let page = $(this).data('page');

        let pageURL = window.location.href;
        let idFromURL = pageURL.split('/');
        let id = idFromURL[4];
        let idAdmin = idFromURL[5];

        $.ajax({
            url: page,
            dataType: 'json',
            method: 'GET',
            success: function(data) {
                switch (window.location.href) {
                    case baseURL + 'admin/courses/create':
                    case baseURL + 'admin/courses/create#':
                    case baseURL + 'instructor':
                    case baseURL + 'instructor#':
                    case baseURL + 'instructor/' + id + '/edit':
                    case baseURL + 'instructor/' + id + '/edit#':
                    case baseURL + 'admin/courses/' + idAdmin + '/edit':
                    case baseURL + 'admin/courses/' + idAdmin + '/edit#':
                        printAllCoursesTable(data.data);
                    case baseURL + 'admin/categories/create':
                    case baseURL + 'admin/categories/create#':
                        printAllCategoriesTable(data.data);
                        break;
                    case baseURL + 'admin/topics/create':
                    case baseURL + 'admin/topics/create#':
                        printAllTopicsTable(data.data);
```

```
                        break;
                    case baseURL + 'admin/users/create':
                    case baseURL + 'admin/users/create#':
                    case baseURL + 'admin/users/' + idAdmin + '/edit':
                    case baseURL + 'admin/users/' + idAdmin + '/edit#':
                        printAllUsersTable(data.data);
                        break;
                    case baseURL + 'admin/contact/create':
                    case baseURL + 'admin/contact/create#':
                        printAllMailsTable(data.data);
                        break;
                }
            },
            error: function(xhr, status, error) {
                alert(xhr.status);
            }
        });
    });

    if (regExpAdminMails.test(window.location.href)) {
        $('#table').attr('id', 'mailsTable');
        $('#table_paginate').attr('id', 'mailsTable_paginate');

        loadAdminMails();

        $(document).on('click', '.delete-mail', function() {
            let id = $(this).data('id');

            $.ajax({
                url: '/admin/contact/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    if (xhr.status == 204) {
                        loadAdminMails();
                        $('.js-notification').html('Successfully deleted');
                    }
                },
                error: function(xhr, status, error) {
                    $('.js-notification').html(error);
                }
            });
        });
    }

    if (regExpEditCourses.test(window.location.href)) {
        //$('.btnRemoveLessonFromEdit').hide();
        // $('#btnAddLesson').hide();
    }

    if (regExpInstructor.test(window.location.href) ||
regExpInstructorEditCourse.test(window.location.href)) {
```

```
        $('#table').attr('id', 'coursesTable');
        $('#table_paginate').attr('id', 'coursesTable_paginate');

        loadInstructorCourses();

        $(document).on('click', '.delete_course_instructor', function() {
            let id = $(this).data('id');

            $.ajax({
                url: '/destroy/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    if (xhr.status == 204) {
                        loadInstructorCourses();
                        $('.msgCrud').html('Successfully deleted.');
                    }
                },
                error: function(xhr, status, error) {
                    $('.msgCrud').html(error);
                }
            });
        });
    }

    if (regExpAdminCourses.test(window.location.href) ||
regExpEditCourses.test(window.location.href)) {
        $('#table').attr('id', 'coursesTable');
        $('#table_paginate').attr('id', 'coursesTable_paginate');

        loadAdminCourses();

        $(document).on('click', '.delete_course', function() {
            let id = $(this).data('id');

            $.ajax({
                url: '/admin/courses/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    if (xhr.status == 204) {
                        loadAdminCourses();
                        $('.msgCrud').html('Successfully deleted.');
                    }
                },
                error: function(xhr, status, error) {
                    $('.msgCrud').html(error);
                }
            });
        });
    }
```

```
    if (regExpAdminCategories.test(window.location.href) ||
regExpEditCategories.test(window.location.href)) {
        $('#table').attr('id', 'categoriesTable');
        $('#table_paginate').attr('id', 'categoriesTable_paginate');

        loadAdminCategories();

        $(document).on('click', '.delete-category', function() {
            let id = $(this).data('id');

            $.ajax({
                url: '/admin/categories/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    if (xhr.status == 204) {
                        loadAdminCategories();
                        $('.msgCrud').html('Successfully deleted.');
                    }
                },
                error: function(xhr, status, error) {
                    $('.msgCrud').html(error);
                }
            });
        });
    }

    if (regExpAdminTopics.test(window.location.href) ||
regExpEditTopics.test(window.location.href)) {
        $('#table').attr('id', 'topicsTable');
        $('#table_paginate').attr('id', 'topicsTable_paginate');

        loadAdminTopics();

        $(document).on('click', '.delete-topic', function() {
            let id = $(this).data('id');

            $.ajax({
                url: '/admin/topics/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    if (xhr.status == 204) {
                        loadAdminTopics();
                        $('.msgCrud').html('Successfully deleted');
                    }
                },
                error: function(xhr, status, error) {
                    $('.msgCrud').html(error);
                }
            });
        });
    }
```

```
    $('.btnRemoveLessonFromEdit').click(function() {
        let id = $(this).data('id');
        if (confirm('Are you sure you want to delete this lesson?')) {
            $.ajax({
                url: '/admin/lesson/' + id,
                method: 'DELETE',
                success: function(data, status, xhr) {
                    alert('Successfully deleted lesson.');
                    location.reload();
                },
                error: function(xhr, status, error) {
                    $('.msgCrud').html(error);
                    $('.msgCrud').html(xhr.status);
                }
            });
        }
    });

    if (regExpAdminUsers.test(window.location.href) ||
regExpEditUsers.test(window.location.href)) {
        $('#table').attr('id', 'usersTable');
        $('#table_paginate').attr('id', 'usersTable_paginate');

        $('#usersTable_paginate').css('margin-top', '12px');

        loadAdminUsers();

        $(document).on('click', '#AddUser', function() {
            data = {
                username: $('#username').val().trim(),
                email: $('#email').val().trim(),
                password: $('#password').val().trim(),
                confirmPassword: $('#confirmPassword').val().trim(),
                role: $('#role').val()
            };

            $.ajax({
                url: '/admin/users',
                dataType: 'json',
                method: 'POST',
                headers: {
                    Accept: 'application/json'
                },
                data: data,
                success: function(data, status, xhr) {
                    if (xhr.status == 201) {
                        $('.js-notification').html('Successfully added a user.');
                        resetUserForm();
                        loadAdminUsers();
                    }
```

```
            },
            error: function(xhr, status, error) {
                if (xhr.status == 422) {
                    let printErrors = handleParametersException(xhr);
                    $('.js-notification').html(printErrors);
                }
                if (xhr.status == 500) {
                    let error = JSON.parse(xhr.responseText);
                    $('.js-notification').html(error.error);
                }
            }
        });
    });

    $(document).on('click', '.delete-user', function() {
        let id = $(this).data('id');

        $.ajax({
            url: '/admin/users/' + id,
            method: 'DELETE',
            success: function(data, status, xhr) {
                if (xhr.status == 204) {
                    loadAdminUsers();
                    $('.msgCrud').html('Successfully deleted');
                }
            },
            error: function(xhr, status, error) {
                $('.msgCrud').html(error);
            }
        });
    });
}

if (regExpLoginRegister.test(window.location.href)) {
    $('#login-form-link').click(function(e) {
        e.preventDefault();

        $('#login-form').delay(100).fadeIn(100);

        $('#register-form').fadeOut(100);
        $('#register-form-link').removeClass('active');

        $(this).addClass('active');
    });

    $('#register-form-link').click(function(e) {
        e.preventDefault();

        $('#register-form').delay(100).fadeIn(100);

        $('#login-form').fadeOut(100);
```

```
            $('#login-form-link').removeClass('active');

            $(this).addClass('active');
        });

        $('#register-submit').click(function() {
            let data = {
                username: $('#usernameReg').val(),
                email: $('#emailReg').val(),
                password: $('#passwordReg').val(),
                passwordConfirm: $('#passwordRegConf').val()
            };

            let errors = regexForRegistration(data);

            if (errors != null) {
                $('#notification').html(errors);
            } else {
                $.ajax({
                    url: '/register',
                    dataType: 'json',
                    method: 'POST',
                    headers: {
                        Accept: 'application/json'
                    },
                    data: data,
                    success: function(data, status, xhr) {
                        if (xhr.status == 201) {
                            $('#notification').html('Successfully Registered!');
                            resetRegistrationForm();
                        }
                    },
                    error: function(xhr, status, error) {
                        if (xhr.status == 422) {
                            let printErrors = handleParametersException(xhr);
                            $('#notification').html(printErrors);
                        }
                        if (xhr.status == 500) {
                            let error = JSON.parse(xhr.responseText);
                            $('#notification').html(error.error);
                        }
                    }
                });
            }
        });
    }

    let counter = 1;
    let maxLessons = 3;
    $(document).on('click', '#btnAddLesson', function() {
        if (maxLessons > counter) {
```

```
            $('#lesson input:text').attr('name', 'lesson[]');

            counter++;

            html = `
            <div class="form-group row lessonRow" id="${counter}">
                <input type="hidden" name="hiddenEmptyFieldLesson"/>
                <div class="col-11">
                    <input type="text" class="form-control mb-8"
                        id="${counter}"
                        name="lesson[]"
                        placeholder="Enter URL for lesson" />
                </div>
            `;
            if (counter > 1) {
                html += `
                <div class="col-1">
                    <button type="button" class="btn" id="btnRemoveLesson"><i class="fa
fa-times"></i></button>
                </div>
                `;
            }
            `</div>`;

            let btnAddLesson = document.getElementById('btnAddLesson');
            btnAddLesson.insertAdjacentHTML('beforebegin', html);
        }

        if (counter === 3) {
            $('#btnAddLesson').fadeOut();
        }
    });

    $(document).on('click', '#btnRemoveLesson', function(e) {
        e.preventDefault();

        $(this).parent('div').parent('div').remove();
        counter--;

        if (counter < 3) {
            $('#btnAddLesson').fadeIn();
        }
    });

    $('.alert-item-cart').click(function() {
        alert('You must login first to add course in cart.');
    });

    $('.alert-item-wish').click(function() {
        alert('You must login first to add this course in wishlist.');
    });
```

```
}); // END of Document

function printPagination(links, url) {
    let print = `<ul class='pagination'>`;
    for (let i = 1; i <= links; i++) {
        print += `<li class='page-item'><a href='#' class='page-link js-pagination'
data-page='${url +
            '?page=' +
            i}'>${i}</a></li>`;
    }
    print += `</ul>`;

    let pageURL = window.location.href;
    let idFromURL = pageURL.split('/');
    let id = idFromURL[4];
    let idAdmin = idFromURL[5];

    switch (window.location.href) {
        case baseURL + 'admin/courses/create':
        case baseURL + 'admin/courses/create#':
        case baseURL + 'instructor':
        case baseURL + 'instructor#':
        case baseURL + 'instructor/' + id + '/edit':
        case baseURL + 'instructor/' + id + '/edit#':
        case baseURL + 'admin/courses/' + idAdmin + '/edit':
        case baseURL + 'admin/courses/' + idAdmin + '/edit#':
            $('#coursesTable_paginate').html(print);
            break;
        case baseURL + 'admin/categories/create':
        case baseURL + 'admin/categories/create#':
            $('#categoriesTable_paginate').html(print);
            break;
        case baseURL + 'admin/topics/create':
        case baseURL + 'admin/topics/create#':
            $('#topicsTable_paginate').html(print);
            break;
        case baseURL + 'admin/users/create':
        case baseURL + 'admin/users/create#':
        case baseURL + 'admin/users/' + idAdmin + '/edit':
        case baseURL + 'admin/users/' + idAdmin + '/edit#':
            $('#usersTable_paginate').html(print);
            break;
        case baseURL + 'admin/contact/create':
        case baseURL + 'admin/contact/create#':
            $('#mailsTable_paginate').html(print);
            break;
    }
}

function loadAdminMails() {
    $.ajax({
```

```
        url: '/admin/contact',
        dataType: 'json',
        method: 'GET',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
            printAllMailsTable(data.data);
        },
        error: function(xhr, status, error) {
            console.log(xhr.status);
            console.log(xhr.responseText);
            console.log(error);
        }
    });
}

function printAllMailsTable(data) {
    let print = `<tr>
            <th>ID</th>
            <th>Subject</th>
            <th>Email</th>
            <th>Message</th>
            <th>Datetime</th>
            <th>Action</th>
        </tr>`;
    for (let i of data) {
        print += `<tr>
                <td>${i.id_contact_mail}</td>
                <td>${i.subject}</td>
                <td>${i.email}</td>
                <td>${i.message}</td>
                <td>${i.date}</td>
                <td><button class="btn btn-outline-danger delete-mail" data-
id="${i.id_contact_mail}">Delete</button></td>
            </tr>`;
    }

    $('#mailsTable').html(print);
}

function loadInstructorCourses() {
    $.ajax({
        url: '/index',
        dataType: 'json',
        method: 'GET',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
            printAllCoursesTable(data.data);
        },
        error: function(xhr, status, error) {
```

```
            alert(xhr.status);
        }
    });
}

function loadAdminCourses() {
    $.ajax({
        url: '/admin/courses',
        dataType: 'json',
        method: 'GET',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
            printAllCoursesTable(data.data);
        },
        error: function(xhr, status, error) {
            alert(xhr.status);
        }
    });
}

function printAllCoursesTable(data) {
    let print = `<tr>
            <th>ID</th>
            <th>Name</th>
            <th>Price</th>
            <th>Created At</th>
            <th>Updated At</th>
            <th>Edit</th>
            <th>Delete</th>
        </tr>`;
    for (let i of data) {
        let id_course = i.id_course;
        print += `<tr>
                <td>${i.id_course}</td>
                <td>${i.course_name}</td>
                <td>${i.price} &euro;</td>
                <td>${i.created_at}</td>
                <td>${i.updated_at}</td>
                <td><a href="${regExpInstructor.test(window.location.href) ||
                regExpInstructorEditCourse.test(window.location.href)
                    ? '/instructor/' + id_course + '/edit'
                    : '/admin/courses/' +
                        id_course +
                        '/edit'}"  class='btn btn-outline-success' data-
id='${i.id_course}'>Edit</a></td>
                <td><a href='#' class='btn btn-outline-danger
${regExpInstructor.test(window.location.href)  ||
                regExpInstructorEditCourse.test(window.location.href)
                    ? 'delete_course_instructor'
```

```
                    : 'delete_course'} delete_course' data-
id='${i.id_course}'>Delete</a></td>
            </tr>`;
    }

    $('#coursesTable').html(print);
}

function loadAdminCategories() {
    $.ajax({
        url: '/admin/categories',
        dataType: 'json',
        method: 'GET',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
            printAllCategoriesTable(data.data);
        },
        error: function(xhr, status, error) {
            console.log(error);
            alert(xhr.status);
        }
    });
}

function printAllCategoriesTable(data) {
    let print = `<tr>
        <th>ID</th>
        <th>Name</th>
        <th>Image</th>
        <th>Created At</th>
        <th>Updated At</th>
        <th>Edit</th>
        <th>Delete</th>
    </tr>`;
    for (let i of data) {
        print += `<tr>
            <td>${i.id_category}</td>
            <td>${i.category_name}</td>
            <td><img src='/img/categories/${i.category_image}' alt='${i.category_name}'
width='85' height='75'/></td>
            <td>${i.created_at}</td>
            <td>${i.updated_at}</td>
            <td><a href='/admin/categories/${i.id_category}/edit' class='btn btn-
outline-success'>Edit</a></td>
            <td><a href='#' class='btn btn-outline-danger delete-category' data-
id='${i.id_category}'>Delete</a></td>
        </tr>`;
    }

    $('#categoriesTable').html(print);
```

```
}

function loadAdminTopics() {
    $.ajax({
        url: '/admin/topics',
        method: 'GET',
        dataType: 'json',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
            printAllTopicsTable(data.data);
        },
        error: function(xhr, status, data) {
            console.log(error);
        }
    });
}

function printAllTopicsTable(data) {
    let print = `<tr>
        <th>ID</th>
        <th>Name</th>
        <th>Created At</th>
        <th>Updated At</th>
        <th>Edit</th>
        <th>Delete</th>
    </tr>`;
    for (let i of data) {
        print += `<tr>
            <td>${i.id_topic}</td>
            <td>${i.topic_name}</td>
            <td>${i.created_at}</td>
            <td>${i.updated_at}</td>
            <td><a href='/admin/topics/${i.id_topic}/edit' class='btn btn-outline-
success'>Edit</a></td>
            <td><a href='#' class='btn btn-outline-danger delete-topic' data-
id='${i.id_topic}'>Delete</a></td>
        </tr>`;
    }

    $('#topicsTable').html(print);
}

function loadAdminUsers() {
    $.ajax({
        url: '/admin/users',
        method: 'GET',
        dataType: 'json',
        success: function(data) {
            let paginationLinks = Math.ceil(data.total / data.per_page);
            printPagination(paginationLinks, data.path);
```

```
            printAllUsersTable(data.data);
        },
        error: function(xhr, status, error) {
            console.log(error);
        }
    });
}

function printAllUsersTable(data) {
    let print = `<tr>
        <th>Username</th>
        <th>Email</th>
        <th>Active</th>
        <th>Insrcutor</th>
        <th>Role</th>
        <th>Created At</th>
        <th>Updated At</th>
        <th>Last Login</th>
        <th>Edit</th>
        <th>Delete</th>
    </tr>`;
    for (let i of data) {
        if (i.last_login == null) {
            i.last_login = '/';
        }
        if (i.active == 1) {
            i.active = 'Yes';
        } else {
            i.active = 'No';
        }
        if (i.is_instructor == 1) {
            i.is_instructor = 'Yes';
        } else {
            i.is_instructor = 'No';
        }
        print += `<tr>
            <td>${i.username}</td>
            <td>${i.email}</td>
            <td>${i.active}</td>
            <td>${i.is_instructor}</td>
            <td>${i.role_name}</td>
            <td>${i.created_at}</td>
            <td>${i.updated_at}</td>
            <td>${i.last_login}</td>
            <td><a href='/admin/users/${i.id_user}/edit' class='btn btn-outline-
success'>Edit</a></td>
            <td><a href='#' class='btn btn-outline-danger delete-user' data-
id='${i.id_user}'>Delete</a></td>
        </tr>`;
    }
```

```
    $('#usersTable').html(print);
}

function regexForRegistration(data) {
    let regexEmail = /^[A-z0-9._%+-]+@[A-z0-9.-]+\.[A-z]{2,}$/;
    let regexUsername = /^[\d\w\_\-\.]{6,30}$/;
    let regexPassword = /^[A-z]{3,}[0-9]{1,}$/;

    let errors = [];

    let printErrors = `<div class="alert alert-danger"<ul>`;

    if (!regexEmail.test(data.email)) {
        errors.push('Email is not in good format.');
    }
    if (!regexUsername.test(data.username)) {
        errors.push('Username must have minimum 6 characters.');
    }
    if (!regexPassword.test(data.password)) {
        errors.push('Password must have 3 letters and minimum 1 number.');
    }
    if (data.passwordConfirm != data.password) {
        errors.push('Passwords do not match.');
    }

    if (errors.length) {
        for (let i of errors) {
            printErrors += `<li>${i}</li>`;
        }
        printErrors += `</ul></div>`;

        return printErrors;
    }

    return null;
}

function resetRegistrationForm() {
    $('#emailReg').val('');
    $('#usernameReg').val('');
    $('#passwordReg').val('');
    $('#passwordRegConf').val('');
}
function resetUserForm() {
    $('#username').val('');
    $('#email').val('');
    $('#password').val('');
    $('#confirmPassword').val('');
    $('#role').val('0');
}
```

```
function handleParametersException(xhr) {
    let error = JSON.parse(xhr.responseText).errors;
    Object.keys(error);

    let printErrors = `<div class="alert alert-danger"<ul>`;
    for (let i of Object.values(error)) {
        printErrors += `<li>${i[0]}</li>`;
    }
    printErrors += `</ul></div>`;

    return printErrors;
}

$('#btnVote').click(function() {
    let idUser = $('#hdnIdUsr').val();
    let answer = $("input[name='teachingAns']:checked").val();
    // console.log(answer);
    if (!answer) {
        $('#votingRes').html('You must choose answer.');
    } else {
        $.ajax({
            url: '/api/vote',
            method: 'POST',
            dataType: 'json',
            data: {
                idUser: idUser,
                answer: answer
            },
            success: function(data, status, xhr) {
                if (xhr.status == 201) {
                    $('#votingRes').html('You have now become instructor.');

                    $("input[name='teachingAns']:checked").val('');

                    $('#btnCreateCourse').css('display', 'block');
                    $('#poll-instructor').css('display', 'none');
                    $('#poll-heading').css('display', 'none');
                    $('#btnVote').css('display', 'none');
                }
            },
            error: function(xhr, status, error) {
                console.log(xhr.status);
                console.log(error);
            }
        });
    }
});
```

4.8.2 cart.js

```
$(document).ready(function() {
    $('.add-course-to-cart').click(addToCart);
});

function addToCart() {
    let id = $(this).data('idcourse');

    let courses = coursesInCart();

    if (courses) {
        if (courseAlreadyInCart()) {
            alert('Course is already added in cart, You cannot add more.');
        } else {
            addInLocalStorage();
            alert('Successfully added course in cart! Go to checkout in cart.');
        }
    } else {
        addFirstCourse();
        alert('Successfully added course in cart! Go to checkout in cart.');
    }

    function courseAlreadyInCart() {
        return courses.filter((c) => c.id == id).length;
    }
    function addInLocalStorage() {
        let courses = coursesInCart();
        courses.push({
            id: id
        });
        localStorage.setItem('courses', JSON.stringify(courses));
        $('#numberInCart').html(courses.length);
    }
    function addFirstCourse() {
        let courses = [];
        courses[0] = {
            id: id
        };
        localStorage.setItem('courses', JSON.stringify(courses));
        $('#numberInCart').html(courses.length);
    }
}

function showCart() {
    let courses = coursesInCart();

    if (courses == null) {
        showEmptyCart();
    } else {
        $.ajax({
```

```
            url: '/cart/showCourses',
            method: 'GET',
            dataType: 'json',
            success: function(data) {
                data = data.filter((c) => {
                    for (let course of courses) {
                        if (c.id_course == course.id) {
                            return true;
                        }
                    }
                    return false;
                });
                makeTable(data);
                addItemsToCart(data);
            },
            error: function(xhr, status, error) {
                console.log(xhr.responseText);
                console.log(error);
            }
        });
    }
}

function makeTable(data) {
    if (data.length == 0) {
        $('.table-cart').html('<h2 class="text-danger">Your cart is empty.</h2>');
        $('.checkout-btn').css('display', 'none');
        $('.table-cart').css('border', 'none');
    } else {
        $('.checkout-btn').css('display', 'block');
        let html = '';

        data.forEach((d) => {
            html += `
            <tr>
                <td class='cart-image'><a href='#'><img
src='/img/courses/${d.image_small}' alt='${d.course_name}' /></a></td>
                <td class='cart-name'><a href='#'>${d.course_name}</a></td>
                <td class='cart-price'><span>${d.price} &euro;</span></td>
                <td>${d.total_hours}</td>
                <td><a href='javascript:void(0)'
onclick='removeFromCart(${d.id_course})' class='removeFromCart'><i class='fas fa-times
fa-2x'></i></a></td>
            </tr>
        `;
        });

        $('#cart').html(html);
    }

    $('#numberInCart').html(data.length);
```

```
}

function addItemsToCart(data) {
    let items = [];

    data.forEach((d) => {
        items.push(d.id_course);
    });

    $('#cartItems').val(items.join(','));
}

function removeFromCart(id) {
    let courses = coursesInCart();
    let filtered = courses.filter((c) => c.id != id);

    localStorage.setItem('courses', JSON.stringify(filtered));

    showCart();
}

function showEmptyCart() {
    $('#numberInCart').html('0');
}

function coursesInCart() {
    return JSON.parse(localStorage.getItem('courses'));
}

let courses = coursesInCart();
if (courses == null) {
    showEmptyCart();
} else {
    showCart();
}
```

4.8.3 contact.js

```
if (window.location.href.indexOf('contact') != -1) {
    var button = document.querySelector('#submitContact');
    button.addEventListener('click', contact);

    document.querySelector('#resetContact').addEventListener('click', function() {
        document.querySelector('#subject').value = '';
        document.querySelector('#email').value = '';
        document.querySelector('#message').value = '';
    });

    function contact() {
```

```
        var email = document.querySelector('#email').value;
        var subject = document.querySelector('#subject').value;
        var message = document.querySelector('#message').value;

        var emailError = document.querySelector('#emailHelp');
        var subjectError = document.querySelector('#subjectHelp');
        var messageError = document.querySelector('#messageHelp');

        var emailTrue = true;
        var subjectTrue = true;
        var messageTrue = true;

        var reSubject = /[A-z0-9]+/;
        var reEmail = /^([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-z]{2,4})$/;

        if (subject) {
            if (!reSubject.test(subject)) {
                subjectError.textContent = 'Subject is not in good format!';
                subjectTrue = false;
            }
        } else {
            subjectError.textContent = 'Subject field is required!';
            subjectTrue = false;
        }

        if (email) {
            if (!reEmail.test(email)) {
                emailError.textContent = 'Email is not in good format!';
                emailTrue = false;
            }
        } else {
            emailError.textContent = 'Email is required!';
            emailTrue = false;
        }

        if (!message) {
            messageError.textContent = 'Message is required!';
            messageTrue = false;
        }

        if (subjectTrue && emailTrue && messageTrue) {
            subjectError.textContent = '';
            emailError.textContent = '';
            messageError.textContent = '';

            $.ajax({
                headers: {
                    'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('message')
                },
                url: '/contact',
                method: 'POST',
```

```
                dataType: 'json',
                data: {
                    submitContact: 'send',
                    subject,
                    email,
                    message
                },
                success: function(data) {
                    document.querySelector('#subject').value = '';
                    document.querySelector('#email').value = '';
                    document.querySelector('#message').value = '';

                    var alertDiv = document.querySelector('#successMessage');
                    var message = document.querySelector('#msg');

                    alertDiv.classList.remove('invisible');
                    alertDiv.classList.add('alert-success');
                    message.textContent = data.success;
                    setTimeout(function() {
                        alertDiv.classList.add('invisible');
                    }, 2500);
                },
                error: function(xhr, status, err) {
                    var msgAlert = document.querySelector('#successMessage');
                    var message = document.querySelector('#msg');
                    switch (xhr.status) {
                        case 409:
                            msgAlert.classList.remove('invisible');
                            msgAlert.classList.add('alert-warning');
                            message.textContent = 'Conflict!';
                            break;
                        case 422:
                            msgAlert.classList.remove('invisible');
                            msgAlert.classList.add('alert-warning');
                            message.textContent = 'Error!';
                            break;
                    }
                }
            });
        } else {
            console.log('Error');
        }
    }
}
```

4.8.4 wish.js

```
$(document).ready(function() {
    const baseURL = 'http://localhost:8000/';
```

```
    const regExpWishlist = new RegExp(`^${baseURL}wishlist[\#]?$`);

    if (regExpWishlist.test(window.location.href)) {
        getAllWishesForOneUser();
    }

    numberOfWishes();
});

$('.wishhhh').click(addWish);

function addWish() {
    let idCourse = $(this).data('idcourse');

    $.ajax({
        url: '/api/addWish',
        method: 'POST',
        data: {
            idCourse: idCourse
        },
        success: function() {
            alert('Successfully added into wishes.');
            numberOfWishes();
        },
        error: function(xhr, status, error) {
            console.log(error);
            if (xhr.status == 400) {
                alert('You already have wish in wishlist.');
            }
            if (xhr.status == 404) {
                alert('Login if you want to add wish.');
            }
        }
    });
}

function numberOfWishes() {
    $.ajax({
        url: '/api/numberOfWishes',
        method: 'GET',
        dataType: 'json',
        success: function(data) {
            $('#numberOfWishes').html(data);
        },
        error: function(xhr, status, error) {
            console.log(error);
        }
    });
}

function getAllWishesForOneUser() {
```

```
    $.ajax({
        url: '/api/wishlist',
        method: 'GET',
        dataType: 'json',
        success: function(data) {
            printAllWishes(data);
            numberOfWishes();
        },
        error: function(xhr, status, error) {
            console.log(error);
        }
    });
}

function printAllWishes(data) {
    if (data.length == 0) {
        $('.table-wish').html('<h2 class="text-danger">Your wishlist is empty.</h2>');
        $('.table-wish').css('border', 'none');
    } else {
        let html = '';

        data.forEach((d) => {
            html += `
            <tr>
                <td class='wish-image'><a href='courses/${d.id_course}'><img
src='/img/courses/${d.image_small}' alt='${d.course_name}' /></a></td>
                <td class='wish-name'><a
href='courses/${d.id_course}'>${d.course_name}</a></td>
                <td class='wish-price'><span>${d.price} &euro;</span></td>
                <td class='wish-link'><a href='courses/${d.id_course}'>Visit</a></td>
                <td class='wish-deleteBtn'><a href='#' class='delete-wish' data-
idwish='${d.id_wish}'><i class='fas fa-trash-alt fa-2x'></i></a></td>
            </tr>
        `;
        });

        $('#wishlist').html(html);
    }
}

$(document).on('click', '.delete-wish', function(e) {
    e.preventDefault();

    let idWish = $(this).data('idwish');

    $.ajax({
        url: '/api/deleteWish',
        method: 'DELETE',
        data: {
            idWish: idWish
        },
```

```
        success: function() {
            getAllWishesForOneUser();
            $('.msg').text('Successfully deleted wish.');
        },
        error: function(xhr, status, error) {
            console.log(error);
        }
    });
});
```

4.9 CSS

Korišćen je deo preuzetog CSS-a za template.

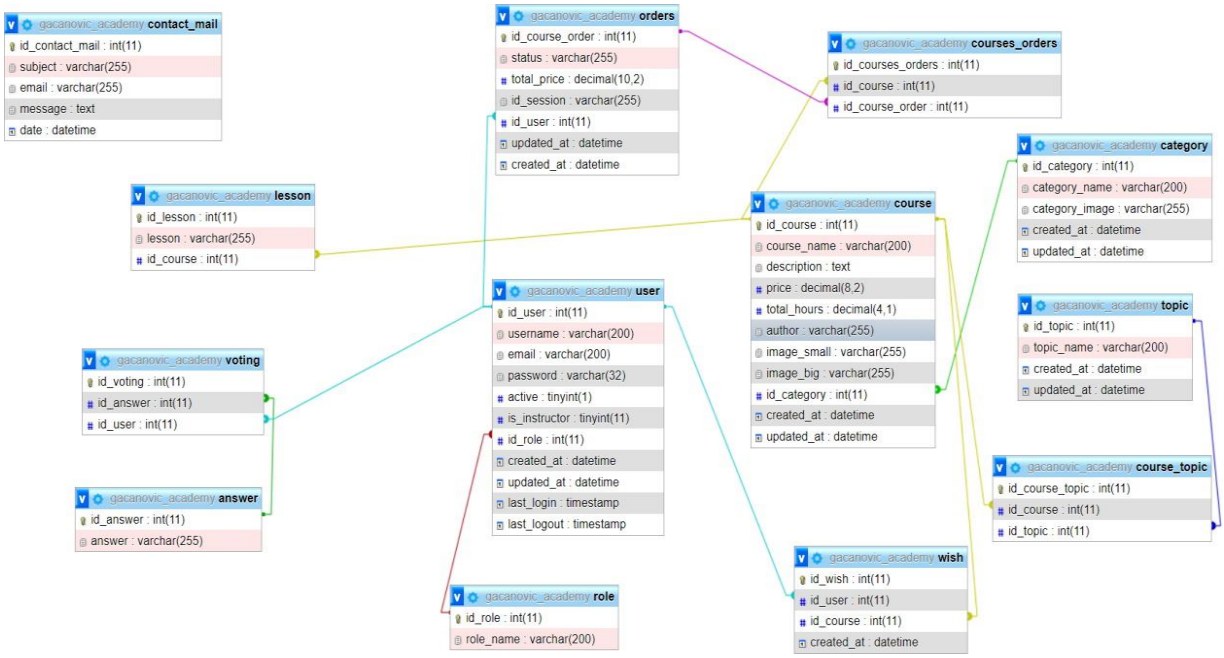
Custom CSS se nalazi u fajlu style.css, zbog obima ga ne postavljam u dokumentaciju.

https://www.os-templates.com/website-templates/template-demos/free-website-templates/trealop/

5 Baza

Struktura baze podataka (slika 31) se sastoji od 13 tabela.

Ovo je MySQL baza u kojoj su smešteni podaci o e-kursevima i njihovim propratnim stvarima. Svi podaci su numerisani jedinstvenim identifikatorom.



Slika 31. Baza podataka

6 Zaključak

Web aplikacija ‘‘Gačanović Academy” je online platform izrađena u Laravel frameworku na kojoj autorizovani korisnici izvršavaju kupovinu željenih e-kurseva iz različitih kategorija i tema. Autorizovani korisnik može biti i autor e-kurseva ukoliko je odgovorio na pitanje iz ankete. Prilikom dolaska na aplikaciju korisnik je imao pregled najjeftinijih e-kurseva po kategorijama, kao i pregled svih e-kurseva dostupnih za kupovinu. Nedostupni su oni koje je već kupio ili ukoliko je sam on autor nekog e-kursa. Pretragu je izvršio po nazivu e-kursa ili imenu autora, filtriranje po temama i kategorijama, a sortiranje po ceni. Autorizovani korisnik je nakon dodavanja e-kursa u listu želja odlučio da kupi e-kurs. Klikom na ikonicu korpe željeni e-kurs ubacio je u nju i onda je imao pregled svoje korpe gde je imao opciju da klikne na dugme ‘’Checkout” i da ostavi svoje podatke sa kartice i obavi proces plaćanja preko Stripe-a. Nakon uspešnog plaćanja korisnik je imao pristup sadržaju i detaljima samog e-kursa sa svim lekcijama. Na stranici kontakt mogao je poslati upit ili neke poruke poput pohvale ili kritike.

Administrator aplikacije je moga kroz logove da prati sve bitne aktivnosti na aplikaciji. Mogao je da oduzima korisnicima pravo da budu autori. Imao je uvid i mogućnost dodavanja, izemene i brisanja nad: korisnicima, e-kursevima, kategorijama i temama. Imao je pregled svih porudžbina i poruka u sistemu.

Korišćenjem Stripe API-a i Webhook-a uspešno i bezbedno se realizovalo plaćanje e-kurseva.

7 Literatura

https://stripe.com/docs/payments/checkout/how-checkout-works

https://laravel.com/

https://stackoverflow.com/

Dr Nenad Kojić, Milena Vesić **WEB Programiranje**, Beograd 2016.