Usporedna analiza PHP frameworka pri izradi web aplikacija

Fleis, Stella

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Humanities and Social Sciences / Sveučilište Josipa Jurja Strossmayera u Osijeku, Filozofski fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:142:340355

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-05**

FILOZOFSKI FAKULTET
SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

Repository / Repozitorij:

FFOS-repository - Repository of the Faculty of Humanities and Social Sciences Osijek

Sveučilište J.J. Strossmayera u Osijeku
Filozofski fakultet Osijek
Dvopredmetni diplomski studij nakladništva i informacijskih tehnologija

Stella Fleis

Usporedna analiza PHP frameworka pri izradi web aplikacija

Diplomski rad

Mentor: prof. dr. sc. Boris Badurina

Osijek, 2024.

Sveučilište J.J. Strossmayera u Osijeku
Filozofski fakultet Osijek
Odsjek za informacijske znanosti
Dvopredmetni diplomski studij nakladništva i informacijskih tehnologija

Stella Fleis

Usporedna analiza PHP frameworka pri izradi web aplikacija

Diplomski rad

Društvene znanosti, Informacijske i komunikacijske znanosti, Informacijsko i programsko inženjerstvo

Mentor: prof. dr. sc. Boris Badurina

Osijek, 2024.

Prilog: Izjava o akademskoj čestitosti i o suglasnosti za javno objavljivanje

Obveza je studenta da donju Izjavu vlastoručno potpiše i umetne kao treću stranicu završnoga, odnosno diplomskog rada.

IZJAVA

Izjavljujem s punom materijalnom i moralnom odgovornošću da sam ovaj rad samostalno napisao/napisala te da u njemu nema kopiranih ili prepisanih dijelova teksta tuđih radova, a da nisu označeni kao citati s navođenjem izvora odakle su preneseni.

Svojim vlastoručnim potpisom potvrđujem da sam suglasan/suglasna da Filozofski fakultet u Osijeku trajno pohrani i javno objavi ovaj moj rad u internetskoj bazi završnih i diplomskih radova knjižnice Filozofskog fakulteta u Osijeku, knjižnice Sveučilišta Josipa Jurja Strossmayera u Osijeku i Nacionalne i sveučilišne knjižnice u Zagrebu.

U Osijeku, 26.09.2024.

Stella Fleis

Stella Fleis, 0122228798

Sažetak:

Svrha ovog rada je istražit definicije PHP razvojnih okvira, razloge iza njihove popularnosti te analizirati kako izgleda proces izrade mrežnih aplikacije prilikom njihove implementacije. Cilj je prikazati kako pravilno odabrani razvojni okvir može, na jednostavan način, postići visoke standarde kvalitete i efikasnosti. Kroz razumijevanje značajki svakog od razvojnih okvira i njihovih specifičnosti, programeri mogu procijeniti odgovara li okvir njihovim potrebama te koji od njih je najbolji izbor za određeni projekt. Kako bi se postavila teorijska podloga, rad najprije započinje sa poviješću mrežnih aplikacija, prateći njihov razvoj od ranih statičkih stranica do kompleksnih mrežnih aplikacija koje znamo danas. Zatim slijedi povijesni pregled PHP jezika, objašnjenje njegovih prednosti zajedno s razlozima njegove široke rasprostranjenosti i popularnosti. U narednim poglavljima objašnjavaju se brojne prednosti razvojnih okvira, gdje se poseban fokus stavlja na neke od glavnih značajki koje nude, kao što su: MVC arhitektura, objektno relacijsko mapiranje, sustavi za predloške i sustavi za usmjeravanje. Navedene i mnoge druge značajke, usporedno se i detaljno analiziraju unutar pet najpopularnijih razvojnih okvira: Laravel, Symfony, CodeIgniter, Yii i CakePHP kako bi se prezentirale njihove mane i prednosti. Na temelju popularnosti, odabrani su Laravel i Symofny razvojni okviri kako bi se provela usporedna analiza procesa izrade mrežnih aplikacije u svakom od njih. U nastavku je prikazan način postavljanja projekta, kreiranje modela, kontrolera i pogleda, zajedno s implementiranjem ruta, migracijama i postavljanjem autentifikacijskih paketa, sve kako bi se dao detaljan uvid u njihove alate i različite pristupe rješenjima za određeni problem.

Ključne riječi: razvojni okviri, PHP, mrežne aplikacije, Laravel, Symfony

1. Uvod

Mrežne aplikacije utječu na način na koji komuniciramo, radimo i upravljamo svakodnevnim informacijama, zbog čega su postale gotovo neizostavan dio modernog života. Od samih početaka, još za vremena statičkih stranica, razvoj mrežnih aplikacija bio je potaknut potrebama društva i brzorastućim tehnološkim napredcima, zbog čega se i može reći da se Internet značajno promijenio od vremenu primitivnog prikaza sadržaja do današnjih složenih mrežnih aplikacija. Za izradu modernih i kvalitetnih mrežnih aplikacije, potrebno je poznavati temelje njihove arhitekture, kao i tehnologije koje omogućuju njihov razvoj.

Ovaj diplomski rad pruža analizu PHP razvojnih okvira, istovremeno objašnjavajuće ključne pojmove važne za razumijevanje mrežnih aplikacija. U poglavlju Mrežne aplikacije' objasniti će se definicija mrežne aplikacije, njena povijest i arhitekturu. Osim toga, razmotrit će se prednosti i nedostatci mrežnih aplikacija kako bi se bolje razumio njihov značaj, rasprostranjenost, ali i izazovi. Nakon toga fokus se prebacuje na PHP, jedan od najdugoročnijih i najrasprostranjenijih jezika za razvoj mrežnih aplikacija. Analizirati će se povijest PHP-a, njegovu popularnost i prednosti koje pruža programerima. Sljedeće je poglavlje posvećeno PHP razvojnim okvirima te njihovim glavnim značajkama kao što su MVC arhitektura, ORM, predlošci i sl. Nakon upoznavanja s osnovnim pojmovima definirajućih karakteristika razvojnih okvira, analizirati će se pet najpopularnijih PHP razvojnih okvira uključujući: Laravel, Symfony, CodeIgniter, Yii i CakePHP. Svaki okvir bit će analiziran prema svojim specifičnostima i direktno uspoređivan unutar tablice. Na kraju, u radu će se usporediti proces izrade aplikacija kroz dva najpopularnija PHP okvira: Laravel i Symfony. Analiza uspoređuje proces postavljanja projekta, kreiranje ruta, kontrolera, modela, migracija, sigurnosnih značajki te implementaciju funkcionalnosti kao što su registracija i prijava.

2. Mrežne aplikacije

2.1 Definicija i povijest mrežnih aplikacija

Mrežne aplikacije se mogu definirati kao aplikacijski softveri koje nije potrebno instalirati već im se može pristupiti putem udaljenog poslužitelja koristeći neki od preglednika. Cilj mrežnih aplikacija je da omogući korisnicima interaktivne radnje kao što su prijavljivanje na svoj račun, slanje mailova, instant poruke na društvenim mrežama ili kupovanje u online trgovinama tako što se podaci primaju i šalju tj. razmjenjuju, između preglednika i poslužitelja.[1]

Iako se mrežne aplikacije i mrežne stranice u govoru često naizmjenično koriste, važno je istaknuti razliku. Kao što i samo ime kaže, mrežne stranice su skup povezanih stranica koje prenose određene informacije, a pristupa im se putem preglednika. Razliku između mrežnih stranica i mrežnih aplikacije najlakše je objasniti po tome što mrežnu stranicu definira sadržaj koji prikazuju, a mrežne aplikacije, interaktivnost koju nudi korisnicima.[2] Bio je potreban dugačak razvoj kako bi napredovali od mrežnih stranica do mrežnih aplikacija. Mrežne aplikacije su tako primarno nastale zbog potrebe za promjenom statičkih mrežnih stranica koje su dominirale Internetom u ranim 1990-tima. Obični tekstni prikazi koji su se mogli samo i jedino čitati, ubrzo su dosadili korisnicima, zbog čega je, s vremenom, njihov razvoj išao u smjeru dodavanja grafičkih stilova, multimedijalnih elemenata poput slika, videa i audio zapisa.[3] Bez obzira na to, krajem 90tih, su tehnologije poput PHP-a i Java Script-a omogućile stvaranje istinski dinamičkih mrežnih sadržaja. Aplikacije su tako dobile mogućnost komuniciranja s bazom podataka pomoću čega bi prikazivale rezultate temeljene na korisničkoj interakciji.[4] Ovdje najveću ulogu igra AJAX (Asynchronous JavaScript and Extensible Markup Language) koji je omogućio komunikaciju sa poslužiteljima te uklonio potrebu da se stranice moraju podnijeti i zatim ponovno učitati kako bi došlo do očekivane promjene, što je dovelo do ogromne razlike u interaktivnosti korisničkih sučelja.[5] U razdoblju do 2010-ih postavljaju se temelji za razvoj moderne mrežne arhitekture organizirajući funkcionalnosti aplikacije u zasebne slojeve: prezentacijski sloj, sloj aplikacijske logike te sloj za pohranu

[1] What is a Web App? 2024. URL: https://www.codecademy.com/article/what-is-a-web-app (2024-06-22)
[2] Isto.
[3] Uryutin, Oleg. A brief history of web app. 2018. URL: https://oleg-uryutin.medium.com/a-brief-history-of-web-app-50d188f30d (2024-06-22)
[4] Gupta, Ayush. Modern Web Application Architecture History and Evolution. 2024. URL: https://www.c-sharpcorner.com/blogs/modern-web-application-architecture-history-and-evolution (2024-06-22)
[5] The Evolution of Modern Web Applications and Their Monitoring. 2020. URL: https://blog.uptrends.com/featured/the-evolution-of-modern-web-applications-and-their-monitoring (2024-06-22)

podataka, sve kako bi omogućio bolji razvoj složenih aplikacije s kompleksnim upravljanjem podacima. [6]

2.2. Arhitektura mrežne aplikacije

Kako bi mrežne aplikacije pružile obujamno, funkcionalno i interaktivno iskustvo koje korisnici očekuju kada ih koriste i pritom osigurale visoke performanse i sigurnost, moraju imati implementirane složene sustave višeslojne arhitekture. Arhitektura mrežne aplikacije je razvojni okvir koji nudi pojašnjenje kako funkcionira veza između klijenta i poslužitelja te određuje kako dijelovi aplikacije međusobno komuniciraju. Djeluje kao vodič za dizajn i razvoj, a sadrži niz principa i pravila dobre prakse kako bi aplikacija izvršila svoju namijenjenu funkciju. Služi kao plan koji organizira što će korisnik vidjeti (frontend), logiku koja stoji iza toga (backend) i kako će oni što bolje međusobno komunicirati. [7]

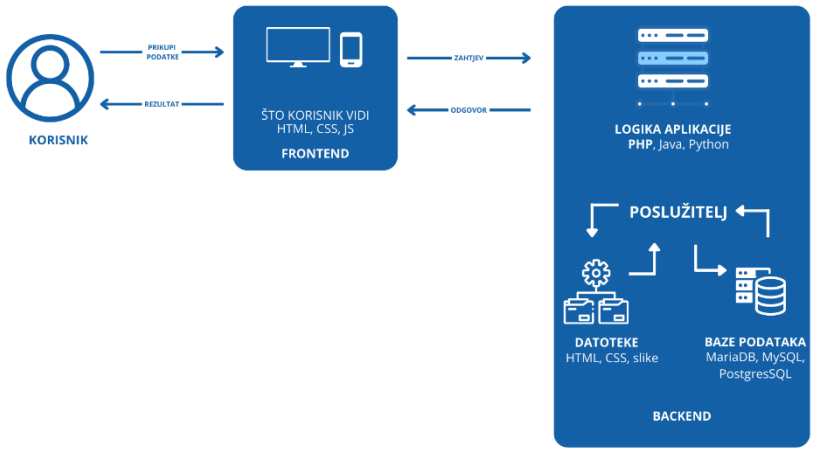
Glavni ciljevi koji ovakva arhitektura pokušava postići su sigurnost, učinkovitost, pouzdanost i skalabilnost sustava. Prije nego što ostvari sve navedeno, primarno mora omogućiti dobro korisničko iskustvo koje se ostvaruje komponentama na frontend-u. Kako bi fronted bio funkcionalan treba omogućiti i efikasan poslužitelj koji će se nositi sa logikom aplikacije, procesuiranjem podataka i komuniciranjem. Budući da mrežne aplikacije kontinuirano evoluiraju, sljedeći bitan cilj je omogućiti sistem koji će biti spreman na veću količinu korisnika i podataka bez kompromitiranja performansi tj. skalabilnost. Te naposlijetku, sigurnosne postavke moraju biti isprepletene u arhitekturu mrežne aplikacije kako bi napravili zaštitu od potencijalnih prijetnji i zajamčili korisnicima sigurnost njihovih informacija.[8]

Komponente arhitekture mrežnih aplikacija se dijele na: komponente korisničkog sučelja i strukturalne komponente. Komponente korisničkog sučelja stavljaju fokus na korisničko iskustvo i prikaz same mrežne aplikacije, što uključuje prikaz, panele, postavke i slično te apsolutno nema nikakve veze sa samom funkcionalnošću aplikacije. Strukturalne su pak komponente sve one koje

[6] Isto.
[7] Petrenko, Viacheslav. Understanding Contemporary Web Application Architecture: Key Components, Best Practices, and Beyond. 2021. URL: https://litslink.com/blog/web-application-architecture#title1 (2024-06-22)
[8] Isto.

predstavljaju glavne funkcionalnosti aplikacije i one s kojima korisnik može manipulirati, a dijele se na klijenta (preglednik), poslužitelja i bazu podataka. [9]

Ako se arhitektura mrežne aplikacije detaljnije raščlani, mogu se zamijetiti obrasci koji se dijele u zasebne slojeve zbog čega nosi naziv multi ili troslojna arhitektura. Tri sloja su: prezentacijski, aplikacijski te podatkovni sloj. Prezentacijskom sloju se pristupa putem preglednika, a sadrži elemente korisničkog sučelja izrađene koristeći HTML, CSS i JS. Aplikacijski sloj je zadužen za prihvaćanje korisničkih zahtjeva na pregledniku, njihovo procesuiranje i reguliranje kako se podacima može pristupiti, kako ih procesuirati i dostaviti . U suštini predstavlja kontroliranje tijeka podataka između sloja korisničkog sučelja i podatkovnog sloja što uključuje procesuiranje zahtjeva, odlučivanje koje dijelove sistema je potrebno aktivirati te priprema podataka koji će se vratiti i prikazati korisniku na korisničkom sučelju. Podatkovni sloj omogućuje pristup potrebnim podacima, a sastoji se od poslužitelja i sustava za upravljanje bazom podataka koji omogućuju komunikaciju s bazom podataka, aplikacijom i korisničkim sučeljem kako bi prikupili i organizirali podatke.[10] Osim tri navedena i osnovna sloja web arhitekture, postoje i drugi dijelovi koji su važni, ali odvojeni od glavne arhitekture. To je dodatni kod koji upravlja komunikacijom i sigurnošću te tzv. *third-party* integracija koja omogućuje korištenje API-ja za potrebe aplikacije kao što su implementacija prijave putem društvenih mreža i pristupnika za plaćanje. [11]



Slika 1. Prikaz arhitekture mrežne aplikacije

[9] How Web Works: Web Application Architecture for Beginners. 2021. URL: https://www.geeksforgeeks.org/how-web-works-web-application-architecture-for-beginners/ (2024-06-22)
[10] Isto.
[11] Isto.

2.3. Prednosti i nedostatci mrežne aplikacije

Mrežne aplikacije se svakodnevno koriste i postale su sastavni dio online iskustva. Kako bi se razumjela njihova popularnost, dobro je napraviti pregled njihovih prednosti, ali i upozoriti na mane. One su praktične, pristupačne i funkcionalne, ali kao i svaka druga tehnologija imaju i svoje nedostatke.

Glavna prednost mrežnih aplikacija je isplativost. Budući da izrada mrežnih aplikacija ne mora bit složen i dugotrajan proces, cjenovno je efikasno rješenje jer nije potrebno uložiti puno sredstava. Programeri pišu jedan kod što rezultira uštedom vremena i posljedično, troškova. U tu kalkulaciju ulazi i činjenica da su mrežne aplikacije kompatibilne s različitim platformama, nije ih potrebno skidati i nije ni bitno je li korisnik na stolnom računalu, tabletu ili mobilnom telefonu, već svi korisnici pristupaju istoj verziji aplikacije. Kod mrežnih aplikacija sve promjene se automatski postavljaju na poslužitelj, čineći aplikaciju uvijek ažurnom i lakšom za održavanje. Ovakav univerzalni pristup, omogućuje i širu bazu korisnika koja može uživati u benefitima koje mrežne aplikacije pružaju. Još jedan benefit kojeg mrežne aplikacije mogu nuditi je responzivnost. To znači da aplikacije imaju mogućnost prilagodbe različitim ekranima, bez obzira jesu li u pitanju stolna računala ili mobilni telefoni, čineći korisničko iskustvo za toliko boljim. Prednost je i dostupnost putem preglednika, što znači da korisnici lako mogu pronaći mrežnu aplikaciju putem tražilice, povećavajući vidljivost što može rezultirati poboljšanom promoviranju ili prodaji određenog proizvoda ili usluge. [12]

Prednost što se mrežne aplikacije uvijek nalaze nadohvat ruke samo putem interneta, može biti i veliki nedostatak ukoliko korisnik nema pristup Internetu. Iako u današnje vrijeme je to rijetkost, ipak je mana za ljude koji žive u mjestima s nestabilnim internetom. Također kada u pitanje dođu brzina aplikacije, napredne grafike ili specifične značajke operativnih sustava i uređaja, mrežne aplikacije ne mogu biti konkurencija nativnima – aplikacijama razvijenim samo za određeni operativni sustav.[13] Te naposlijetku, budući da se mrežnim aplikacijama pristupa putem preglednika, a oslanjaju se na poslužitelja, nedostatak predstavlja i rizik od sigurnosnih napada zbog čega je potrebno implementirati sigurnosne značajke. [14]

[12] Main Advantages & Disadvantages of Web Apps in 2024. 2024. URL: https://www.makeitsimple.co.uk/blog/web-app-advantages-disadvantages (2024-06-22)
[13] What Are the Advantages and Disadvantages of Web Applications? 2023. URL: https://www.itrobes.com/what-are-the-advantages-and-disadvantages-of-web-applications/ (2024-06-22)
[14] Main Advantages & Disadvantages of Web apps in 2024. Nav.dj.

3. PHP: Hypertext Preprocessor

3.1 Povijest PHP jezika

Rasmus Lerdorf osmislio je PHP-a programski jezik 1994. godine, a prvu verziju prezentirao javnosti 1995. godine postavljajući objavu na Usenet newsgroup naslova: „ Alati za osobnu mrežnu stranicu (PHP alati).“[15] U objavi Lerdorf objašnjava kako su ovi PHP alati zapravo skup CGI (Common Gateway Interface) binarnih datoteka napisanih u C programskom jeziku, u potpunosti besplatni, a daju odgovore na neke od najčešćih potreba koji su ljudi tada imali kao što su: **zaštita stranica lozinkom, onemogućavanje posjetitelja na temelju njihove domene, brojanje posjeta na stranici, jednostavno kreiranje obrazaca, pristup podacima iz tih obrazaca** i sl.[16]

Tijekom sljedeće godine, na isti način je najavljena i druga verzija pod nazivom „PHP/FI skriptni poslužiteljski jezik ugrađen u HTML “, ovaj put opisana kao najbrži i najjednostavniji alat za kreiranje mrežnih stranica s integriranim bazama podataka i drugim značajkama poput evidentiranja ili ograničavanja pristupa, podrška za prijenos datoteka, prilagođavanje HTTP zaglavlja što omogućuje naprednije značajke poput kolačića te podrška za uvjete i petlje.[17] Ova, druga verzija, lansirala je PHP iz seta alata u pravi programski jezik, podržavajući upravljanje bazama podatka, MySQL, Postgres, kolačiće i druge funkcionalnosti za svoje korisnike.[18]

PHP 2.0 je prikupio veću popularnost, ali pojavili su se određeni problemi sa stabilnošću korištenog parsera, zbog čega su se Zeev Suraski i Andi Gutmans odlučili napisati poboljšanu verziju što je dovelo do baze za treće verzije PHP-a. Ovo je verzija koja najviše podsjeća na PHP koji se danas koristi jer uključuje objektno-orijentirano programiranje i sigurnosna ažuriranja, te je verzija u kojoj PHP poprima akronim pod kojima ga danas znamo: Hypertext Preprocessor.[19] Navedeni doprinosi su potaknuli i druge programere na slične postupke, zbog čega je PHP prerastao u projekt upotpunosti u otvorenom pristupu omogućujući programerima diljem svijeta da daju svoj doprinos.

---

[15] Tatroe, Kevin; MacIntyre, Peter. Programming PHP: Creating Dynamic Web Pages. 4th ed. Sebastopol, CA: O’Reilly Media, 2020. Str. 40.
[16] Isto, str. 41.
[17] Isto, str. 43.
[18] PHP: Hypertext Preprocessor. PHP: History of PHP - Manual. URL: https://www.php.net/manual/en/history.php.php (2024-06-28)
[19] PHP - History. 2015. URL: https://www.tutorialspoint.com/php/php_history.htm (2024-06-28)

Andi i Zeev nastavili su aktivno raditi na unaprijeđivanju PHP-a kako bi se pomoću njega mogle raditi što kompliciranije aplikacije. Zbog čega je predstavljen 'Zend Engine', kompilator i izvršno okruženje za PHP verziju 4.0 koji je uvelike unaprijedio stabilnost i performanse.[20] PHP 5 je objavljen 2004. godine, uključujući mnoge nove značajke poput podrške za objektno orijentirano programiranje (OOP), PHP Data Objects (PDO) i brojna poboljšanja performansi zajedno s unaprijeđenim Zend Enginom 2.0.[21]

Sljedeće vrijedna spomena verzije je PHP 7.0 koji je objavljen zajedno i sa unaprijeđenom verzijom Zend Engina (3.0). Ovdje je nastavljen trend napredovanje u performansama, omogućujući ovoj verziji da radi i do dva puta brže od prethodnih uz smanjenu potrošnju memorije. Također su implementirane i važne nove značajke poput deklaracija skalarnog (integer, float, string i boolean) i povratnog tipa, omogućujući strožu kontrolu nad ulaznim parametrima i povratnim vrijednostima funkcija što povećava stabilnost i sigurnost koda te ujedno i smanjuje pogreške.[22]

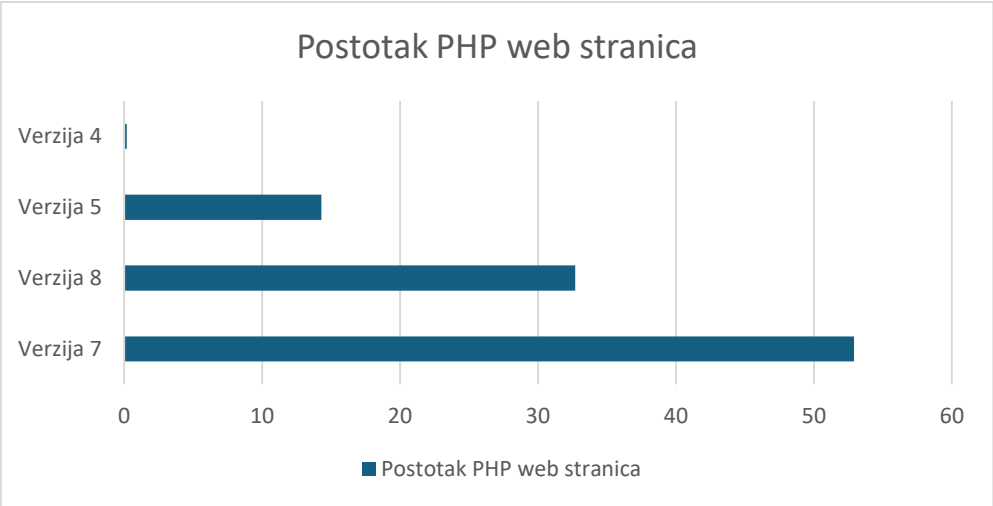
Zadnja velika predstavljena verzija je PHP 8 koji je izašao u 2020, godini a zadnje ažuriranje je verzija 8.3.8. Za PHP 8 najvažnije je spomenuti dolazak JIT (Just-In-Time) kompilatora koji prevodi PHP kod u strojni, omogućujući znatno brže izvršavanje aplikacija. Također je uvedeno definiranje više tipova za jednu varijablu, implementirani su i Atributi koji omogućuju dodavanje metapodataka izravno u PHP kod tj. dodavanje metapodataka u PHP klase što ultimativno olakšava rad s razvojnim okvirima te je poboljšano rukovanje s pogreškama što dodatno smanjuje mogućnost slučajnih grešaka i ujedno povećava stabilnost aplikacije. Ova verzija je programerima tako omogućila izradu sigurnih, brzih i održivih mrežnih aplikacija. [23]

[20] FATHER of PHP. URL: https://www.javatpoint.com/father-of-php (2024-06-28)
[21] PHP: Hypertext Preprocessor. PHP: History of PHP-Manual. Nav.dj
[22] PHP-History. Nav.dj.
[23] Vinugayathri. Exploration of PHP Version History from PHP/FI to PHP 8.3. 2019. URL: https://www.clariontech.com/blog/exploration-of-php-version-history-from-php/fi-to-php-7.3 (2024-06-30)

3.2 Popularnost PHP-a

Od samih početaka, još za vrijeme dok je nosio naziv PHP/FI, ovaj programski jezik je privukao veliku pažnju i popularnost u rastućoj zajednici web programera. Tako je 1998. godine imao već kultno pratiteljstvo, što potvrđuje i istraživanju tvrtke Netcraft, gdje je utvrđeno da je oko 60 000 domena bilo smješteno na poslužiteljima koji su imali instaliran PHP. U to vrijeme, ova brojka je bila ekvivalentna zauzimanju 1% svih domena na Internetu. [24]

Danas je ovaj postotak uvelike veći. Sa novijim verzijama, njegova je popularnost samo rasla (Slika 2.), a prema W3Techs-u, u 2024. godini, PHP je korišten na 76.1% svih stranica.[25] Iako je to ogroman broj mrežnih stranica, treba napomenuti da ove brojke nisu impresivne zbog same količine, već i zbog kvalitete koju predstavljaju. Tako se neke od najpoznatijih tvrtki poput Facebook-a, Etsy-a i Slack-a mogu pohvaliti da su izgrađeni uz pomoć PHP-a.[26] Najveći razlog zašto je postotak toliko visok, skriva se u činjenici da je 40% svih mrežnih stranica napravljeno pomoću CMS-a (Content Managmenet System), a čak tri, od liste najpopularnijih CMS-ova: Wordpress, Joomla i Drupal, su napisani u PHP-u. [27]



Slika 2. Postotak mrežnih stranica koje koriste neke od PHP verzija

[24] PHP: Hypertext Preprocessor. PHP: History of PHP-Manual. Nav.dj
[25] Usage Statistics and Market Share of PHP for Websites. 2024. URL: https://w3techs.com/technologies/details/pl-php (2024-06-30)
[26] Advantages and Disadvantages of PHP. 2024. URL: https://ellow.io/advantages-and-disadvantages-of-php/ 2024-06-30)
[27] PHP Market Share. 2021. URL: https://tinyurl.com/m34te6e2 (2024-06-30)

3.3 Prednosti korištenja PHP-a

Kao što je u prethodnim poglavljima utvrđeno, PHP je vrlo popularan i besplatan poslužiteljski skriptni jezik, čije su glavne značajke dinamičan sadržaj, interakcija s bazom podataka, objektno-orijentirano programiranje i ekstenzivan ekosistem biblioteka i razvojnih okvira. [28]

Popularnost i široka rasprostranjenost PHP-a može se objasniti brojnim prednostima koje nudi. Najprije, jedan od vrlo bitnih faktora za početnike, ali i sve one koji žele napredovati s razvojem tehnologije, predstavlja jednostavnost učenja i korištenja. S obzirom na veličinu zajednice, za PHP postoji opširan niz izvora, tutorijala i općenito podrške drugih korisnika na raznim forumima i drugim platforama gdje se razmjenjuju znanja, daju savjeti i predlažu poboljšanja.[29] Zatim, u **otvorenom je pristupu**, što znači da ga svi mogu slobodno koristiti što uključuje i sve radne okvire, biblioteke i baze podataka. Ima bogatu zbirku funkcija i klasa koji su zaduženi za najpopularnije potrebe poput rukovanje datotekama, datumskim funkcijama i sl., što ultimativno štedi vrijeme razvoja i smanjuje redundantnost koda što dovodi do još jedne velike prednosti; veće isplativosti. Troškovi održavanja su minimalni, svi resursi lako dostupni, a sigurnosna ažuriranja česta. Može se jednostavno preuzeti, kompatibilan je s različitim operativnim sustavima i neovisan je o platformi.[30] Uz to, PHP je kompatibilan s poznatim web poslužiteljima, uključujući Apache i Nginx, što programerima omogućuje da koriste PHP za gotovo sve vrste mrežnih aplikacija, od malih osobnih projekata do velikih sustava na razini poduzeća.[31] Treba istaknuti da su PHP skripte brže od bilo kojih drugih usporednih jezika. Čak i povezivanje s bazom podataka i učitavanje potrebnih podataka iz tablica, brži su od ostalih programskih jezika što stvara veliku prednost kod korisnika. Razlog tome je što se može nositi s više istovremenih zahtjeva, uvelike povećavajući performanse. Ima i mnoge predefinirane funkcije za enkripciju podataka kako bi se omogućila veća razina sigurnosti što često ne ide u korist fleksibilnosti jezika, ali PHP omogućuje oboje. On je skalabilan što znači da može podnijeti velike količine prometa i podataka. **Dizajniran je za učinkovit rad s bazama podataka i može kombinirati s drugim programskim jezicima i tehnologijama poput HMTL-a i JavaScripta** što omogućuje pronalaženje najboljeg rješenja za specifične potrebe. [32]

[28] History of PHP: Evolution of a Web Development. 2024. URL: https://wpwebinfotech.com/blog/history-of-php/#what-is-the-future-of-php (2024-07-03)
[29] Advantages and Disadvantages of PHP. Nav.dj.
[30] Isto.
[31] Agarwal, Yash. Advantages of PHP over Other Programming Languages. 2023. URL: https://www.scaler.com/topics/php-tutorial/advantages-of-php/ (2024-07-03)
[32] PHP Unique Features. 2019. URL: https://www.geeksforgeeks.org/php-unique-features/ (2024-07-03)

4. PHP razvojni okviri i njihove glavne značajke

4.1 Definicija i svrha

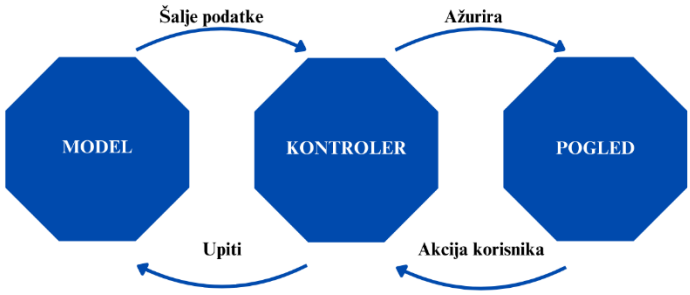
PHP postoji već godinama i na napretku njegova koda se konstantno radi, što je rezultiralo tome da je postao robustan, ali i kompliciran. Bez obzira na tu činjenicu, to nije utjecalo na njegovu popularnost. Razlog tome leži u razvojnim okvirima, koji su omogućili da korištenje PHP-a postane puno ugodnije iskustvo, a u brojnim slučajevima oni čak popravljaju i nedostatke u temeljima samog jezika.[33] Razvojni okviri predstavljaju zbirku unaprijed pripremljenih komponenti i biblioteka koji primarno služe za uštedu vremena i resursa, jer omogućuju implementaciju gotovih rješenja, za razliku od ručne izgradnje projekta iz nule.[34] Programeri ih koriste kako bi imali bolji način za web razvoj i jednostavnije održavanje svojih projekata. Prednost uštede vremena i olakšanog razvoja proizlazi iz činjenice da razvojni okviri nude već gotova rješenje za najčešće zahtjeve pojedine aplikacije, kao što su: autentikacija, pristup bazi podataka, sigurnosni zahtjevi i upravljanje URL-ovima.[35] Oni potiču i RAD (Rapid Application Development) koji se fokusira na veću fleksibilnost, više iteracija i ažuriranja kako bi se što brže došlo do što boljih rezultata. S druge strane, razvojni okviri mogu uvelike pomoći ljudima koji tek ulaze u svijet web razvoja. Takvim korisnicima oni osiguravaju izradu stabilnih aplikacija koje imaju ispravnu arhitekturu i pravilnu komunikacijom s bazom podataka. [36] **Benefit pak za sve korisnike je izbjegavanje repetitivnog pisanja koda**, kako bi se mogli fokusirati na glavnu funkcionalnost aplikacije s rezultatom konzistentnog, kvalitetnog i održivog koda koji slijedi ustaljene prakse i obrasce PHP zajednice.[37] Svrha pojedinog PHP razvojnog okvira ovisi o potrebama projekta i željama programera, a pri odabiru pravog potrebno je obratiti pažnju na ključne faktore. Najprije, treba razmotriti zahtjeve i opseg projekta; za manje projekte prikladniji su lakši razvojni okviri, dok za veće i složenije projekte potrebni su oni robusniji s naprednijim mogućnostima. Optimizacija performansi i skalabilnost su ključni faktor kako bi rezultat bio kvalitetno korisničko iskustvo, a podrška zajednice, učestalost ažuriranja i redovito održavanje ključni su za osiguravanje kvalitete i stabilnosti aplikacije. [38]

[33] PHP Market Share. Nav.dj.
[34] White, Emily. Why Should You Use PHP Frameworks for Web Development? 2022. URL: https://stackify.com/why-should-you-use-php-frameworks-for-web-development/ (2024-07-04)
[35] What are the benefits and drawbacks of using PHP frameworks for performance optimization? 2023. URL: https://tinyurl.com/48v64jxa (2024-07-04)
[36] Discussing PHP Frameworks: What, When, Why and Which? 2021. URL: https://www.jotform.com/blog/discussing-php-frameworks/ (2024-07-04)
[37] What are the benefits and drawbacks of using PHP frameworks for performance optimization? Nav.dj.
[38] Top 10 Best PHP Frameworks in 2024. 2023. URL: https://www.cloudways.com/blog/best-php-frameworks/ (2024-07-04)

4.2 MVC arhitektura

Jedan od glavnih načina kako razvojni okviri olakšavaju proces razvoja, čineći ga bržim i jednostavnijim proizlazi iz implementacije Model-View-Controller (Model-Pogled-Kontroler) arhitekture tj. MVC. U MVC arhitekturi, Model se odnosi na podatke, View na prezentaciju tih podataka, a Controller na logiku same aplikacije. Ovakva arhitektura predstavlja organizaciju koda te postavlja jasnu granicu između logike aplikacije i njenog sučelja kako bi se omogućilo razdvojeno modificiranje bilo kojih elemenata bez straha da će promijene negativno utjecati na njih.[39] Važno je napomenuti i razliku između već spomenute web arhitekture i MVC-a. Iako su povezani pojmovi, koji konceptualno zvuče slično, predstavljaju različite ideje. Definirana arhitektura mrežne aplikacije predstavlja širi pojam uključujući cjelokupnu strukturu i tijek aplikaciju, dok je MVC organizacijska tehnika koja se koristi unutar aplikacijskog sloja kako bi olakšala i ubrzala proces razvoja aplikacije.

Razvojni okviri temeljeni na MVC arhitekturi, prema svome imenu, kod dijele na tri glavne komponente: Model – upravlja s podacima u bazi podataka, Pogled – predstavlja korisničko sučelje i Kontroler – sadrži svu logiku i pomaže povezati Model sa Pogledom (Slika 3.)[40]



Slika 3. MVC arhitektura

[39] Discussing PHP Frameworks: What, When, Why and Which? Nav.dj.
[40] PHP MVC Architecture. 2021. URL: (https://www.javatpoint.com/php-mvc-architecture) (2024-07-05)

Model je zadužen da sve radnje koje korisnik pokrene imaju operaciju za obradu podataka koji se koriste u aplikaciji, tako što dohvaća podatke iz baze podataka s kojima izvodi određenu operaciju, a zatim te podatke ponovno pohranjuje. Odgovoran je za upravljanje podacima tijekom njihove cirkulacije između baze podataka i konačnog pogleda u korisničkom sučelju. Tijekom tog procesa, Model ne zna što se događa s podacima kada se proslijede, tako je njegova jedina svrha obraditi dobivene podatke ili pretražiti određene podatke i proslijediti ih Kontroleru. Sveukupno predstavlja najsloženiji dio MVC arhitekture, jer bez njega ne postoji poveznica između Pogleda i Kontrolera.[41]

Pogled definira kako će izgledati predložak koji se šalje pregledniku za prikaz korisniku. Komunicira i s Modelom i Kontrolerom, a glavna mu je svrha dati modelu korisničko sučelje za prikaz podatak. Pogledu pripadaju komponente poput gumba, tekstnih okvira, izbornika i sl., a služe kako bi pomogli na jasan način korisniku prikazati podatke. Najjednostavniji primjer bio bi klik gumba na Pogledu kojim se pokreće radnja kojom upravlja Kontroler.[42]

Zadaća Kontrolera je da upravlja s podacima koje dobije od korisnika na Pogledu, te sukladno tome i ažurira Model. Bez korisničke interakcije, Kontroler nema svrhu. Tako se svaka funkcija Kontrolera može shvatiti kao okidač pokrenut korisničkom interakcijom. On sakuplja informacije koje onda prosljeđuje Modelu za potrebe pohrane i organizacije tih podatka. Važno je istaknuti da je ovo sustav za jednosmjeran protok podataka jer je Kontroler povezan samo s jednim Modelom i jednim Pogledom.

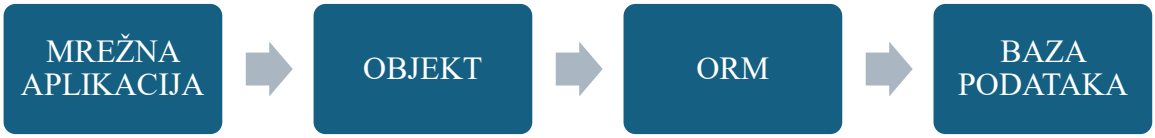
Implementacija MVC arhitekture u projekt bila bi organizirana na sljedeći način: Model će sadržavati funkcije zadužene za rukovanje podacima, Pogled će sadržavati HMTL kod za prikaz podataka, a Kontroler će sadržavati funkcije zadužene za rukovanje korisničkom interakcijom i za ažuriranje Modela i Pogleda.[43] Glavna prednost implementacije ovakve MVC arhitekture je optimiziranje performansi aplikacije, mogućnost ponovne upotrebe koda i sigurnost. A glavne prednosti korištenja razvojnih okvira koji podržavaju MVC arhitekturu su: organizacija velikih projekata, brži proces razvoja, lakša provedba izmjena, pomoć u pozivanju asinkronih metoda što

[41] Hopkins, Callum. PHP Master | the MVC Pattern and PHP, Part 1. 2013. URL: https://www.javatpoint.com/php-mvc-architecture (2024-07-05)
[42] Introduction to Laravel and MVC Framework. 2022. URL: https://www.geeksforgeeks.org/introduction-to-laravel-and-mvc-framework/ (2024-07-05)
[43] Hopkins, Callum. Nav. dj.

omogućava istovremeno procesuiranje više zadataka, a time se poboljšavaju performanse i ostvaruje bolje korisničko iskustvo.[44]

4.3 Objektno-relacijsko mapiranje

Objektno-relacijsko mapiranje (ORM), u kontekstu PHP-a, je tehnika koju programeri koriste kako bi povezali relacijske baze podataka s principima objektno-orijentiranog programiranja (OOP) tj. pretvorili podatke iz baze podataka u PHP klase i objekte.[45] Proces povezivanja objekata iz OOP-a s tablicama relacijskih baza podataka se zove Mapiranje (Slika 4). Prema objektno orijentiranom programiranju klasa se može definirati kao opisnik objekta, a objekt kao instanca klase. Tako bi na primjer klasa „Stvar“ imala određena definirajuća svojstva poput „naziv“, „materijal“ i „boja“, objekt bi bio instanca toga, znači pridavanje specifične vrijednosti navedenim svojstvima. Ovo je bitno definirati jer u ORM-u, klase su ekvivalentne onome što bi bile tablice u relacijskog bazi podataka, a svojstva tih klasa predstavljaju stupce u tablici što omogućuje korištenje objekata kako bi se pristupilo traženim podacima, umjesto pisanje SQL upita. ORM se često koristi za rad s CRUD operacijama (Create – Read- Update – Delete) kako bi olakšano kreirali novi zapisi u relacijskoj bazi podataka, dohvatili ih, po potrebi ažurirali te naposlijetku obrisali i to sve dok ORM automatski stvara potrebne SQL upite kako bi se ove operacije uspješno izvršile.[46]



Slika 4. prikaz ORM-a iz objekta aplikacije u entitet baze podataka

Prednosti korištenja ORM-a uključuje bolju i lakšu interakciju s relacijskim bazama podataka, veću čitljivost koda, poboljšane performanse i sigurnosne značajke. Sam kod postaje optimiziraniji i lakši za održavanje. ORM okviri pak rješavaju rukovanje pogreškama, provjeravaju valjanost

[44] Introduction to Laravel and MVC Framework. Nav. dj.
[45] How do ORM frameworks simplify database interactions in PHP? 2023. URL: https://tinyurl.com/5xtujmxt (2024-07-05)
[46] Horvat, Marko. Objektno relacijsko mapiranje. 2023. URL: edu.asoo.hr/wp-content/uploads/2024/03/G16-64-Objektno-relacijsko-mapiranje-Finalno.pdf (2024-07-05)

podataka, ubrzavaju rad, podržavaju rukovanje s raznim bazama podataka i to sve dok se pridržavaju PHP konvencija i standarda. [47]

Najpoznatiji predstavnici implementacije ORM-a su Eloquent kojeg koristi Laravel okvir i Doctrine kojeg koristi Symfony. Eloquent je intuitivan i snažan alat za rad s podacima je omogućuje jednostavnu i čitljivu sintaksu. Funkcionira na način da svaka tablica u bazi ima odgovarajući model koji olakšava umetanje, ažuriranje i brisanje podataka. Doctrine ORM je fleksibilan i omogućuje mapiranje podataka u objekte za izvođenje složenih upita. **Glavna razlika između njih je način na koji rade apstrakciju sloja baze podataka.** Eloquent za to koristi 'Active Records' u kojem svaki model odgovara jednom redu u tablici bazi podataka sa pripadajućim metodama. Eloquent iz Laravela i Model iz CodeIgnitera koriste ovaj obrazac. S druge strane postoji i 'Data Mapper' kojeg koriste Doctrine i Propel gdje svaki objekt odgovara jednom redu u tablici, ali je drukčiji po tome što se se mapiranje obavlja kroz odvojeni sloj koji upravlja interakcijama s bazom podataka. Eloquent i Doctrine podržavaju migracije koje pomažu održati sinkronizacije između baze podataka i koda. Eloquent to radi pomoću artisan naredbi, dok symfony u svom CLI koristi doctrine naredbe. Pri odabiru ORM-a s kojim se želi raditi potrebno je provjeriti značajke koje nudi te kompatibilnost s verzijom PHP-a, bazom podataka i razvojnim okvirom. Poželjna je velika zajednica i dobro raspisana dokumentacija, ali na kraju odluka ovisi o preferencijama i zahtjevima individualne osobe.[48]

4.4 Predlošci (templating)

Kako bi se izbjeglo pisanje PHP koda direktno u HTML datoteku, mogu se koristiti sustavi za upravljanje predlošcima koji omogućuju učinkovito razdvajanje prezentacijskog i logičkog sloja aplikacije. Ovaj pristup omogućuje promjenu izgleda mrežnih stranice bez mijenjanja temeljnog koda. HTML dokument tada ima predodređena mjesta za izraze i varijable koja se pomoću sustava za predloške zamjenjuju aktualnim PHP podacima.[49]

[47] How do ORM frameworks simplify database interactions in PHP? Nav. dj.
[48] What Are Some Best Practices for Using Laravel's Eloquent ORM vs Symfony's Doctrine ORM? 2023. URL: www.linkedin.com/advice/0/what-some-best-practices-using-laravels-eloquent (2024-08-20)
[49] J, Olivia. How to Use PHP Templating Engines for Web Development. 2023. URL: https://medium.com/@olivia.j.01101001/how-to-use-php-templating-engines-for-web-development-6800c9048b9c (2024-07-10)

Za korištenja predložaka, potrebno je najprije ih instalirati na poslužitelj i tek onda primijeniti u PHP-u. Neki od popularnih su: Smarty, Twig i Blade, a najlakše ih je postaviti uz pomoć Composera. Zatim, treba izraditi predloške u zasebnim datotekama i proslijediti ih sustavu zajedno s podacima koje želimo prezentirati. [50]

Predlošci mogu biti ključni u raznim domenama projekata za koji se korist, jer omogućuju stvaranje skalabilnih, dinamičnih i personaliziranih mrežnih aplikacija koje je lakše održavati, a apsolutno odgovaraju specifičnim potrebama projekta kojem služe. Tako na primjer, pri izradi e-commerce projekta, predlošci omogućuju dinamični prikaz proizvoda, preporuka ili popusta temeljeno na preferencijama korisnika ili stanju proizvoda. Ili društvene mreže, gdje predlošci olakšavaju stvaranje dinamičkih korisničkih profila i prikaz obavijesti u stvarnom vremenu, gdje se sadržaj ažurira ovisno o korisničkoj interakciji. Još jedan dobar primjer za korištenje predložaka su edukacijske platforme. PHP sustavi za izradu predložaka pomažu u strukturi i prikazu obrazovnog sadržaja, prikazuju korisnikov napredak i stvaraju alate za učenje kako bi korisnicima ponudili dinamično i edukacijsko iskustvo. [51]

Prednosti korištenja sustava za predloške uključuju: lakše održavanje jer se pri ažuriranjima mogu ažurirati samo predlošci bez uplitanje u PHP kod. Zatim, veću fleksibilnost, bolje performanse jer smanjuje opterećenje na poslužitelja i povećanu sigurnost jer ovi sustavi sprječavaju sigurnosne napade. Predlošci nude puno mogućnosti, ali odabir ovisi o specifičnim potrebama i preferencijama.[52]

4.5 Sustavi usmjeravanje (routing)

Tijekom navigiranja kroz mrežne aplikacije, često se znaju koristiti i vidjeti puni nazivi datoteka u URL-u kao što je primjer server/login.php. Kako bi se ispravilo tj. poboljšalo takvo strukturiranje aplikacija i kako bi se uredili „neuredni“ URL-ovi, uveli su se sustavi usmjeravanja odnosno routinga. Routing je sustav usmjeravanja koji navodi HTTP zahtjev prema određenoj funkciji ili metodi zaduženoj za njegovo rukovanje. Na taj način definira kako se pristupa različitim dijelovima aplikacije bez potrebe za upisivanjem cijelog naziva datoteke. Postavljanjem takvih

[50] J, Olivia. Nav. dj.
[51] Sharma, Vinit. How to Use Templating in PHP Frameworks for Business Projects? 2024. URL: https://www.clariontech.com/blog/how-to-use-templating-in-php-frameworks-for-business-projects (2024-07-15)
[52] J, Olivia. Nav. dj.

ruta bi rezultiralo, od prvotnog primjera sa: server/login omogućujući pristup istoj datoteci login.php.[53]

Za primjer kako funkcionira odnos između zahtjeva i prikaza rezultata može se uzeti jednostavan primjer: https://example.com/login. Najprije aplikacija primi zahtjev, zatim ga podijeli u komponente kao što su metode, rute i sl., nakon čega traži specifično definiranu rutu koja odgovara zahtjevu, a nakon što je pronađe, odradi definiranu funkciju i vraća odgovor. Tako bi definirali GET rutu za URL logina, a odgovor bi bio definirana login stranica. U jednom od najpoznatijih PHP razvojnih okvira Laravel, routing se može implementirati u samo par linija koda.[54] (Slika 5.)

```
4   Route::get('/login', function() {
5       return view('login');
6   });
```

Slika 5. Primjer routinga prema Laravel-u

Benefiti korištenja routinga uključuju: fleksibilnost, bolju organizaciju koda omogućujući da se sve rute nalaze na istoj lokaciji, zatim poboljšano upravljanje u slučaju da rutu ne želimo nazvati isto kao datoteke te implementiranje provjera pomoću ruta umjesto ponavljanja istog koda. [55]

[53] Mbula, Abel Lifaefi. How to Build a Routing System for a PHP App from Scratch. 2023. URL: https://tinyurl.com/yscf7uyb (2024-07-16)
[54] Sparks, Matt. Building a PHP Framework: Part 8 - Routing. 2018. URL: https://dev.to/mattsparks/building-a-php-framework-part-8---routing-4jgf (2024-07-16)
[55] Edgar. Why Use a Router in PHP. 2020, (https://uhded.com/php-router) (2024-07-16)

5. Pregled popularnih PHP razvojnih okvira

Prednosti korištenja PHP razvojnih okvira su tolike da su već postali dio ustaljene prakse pri izradi mrežnih aplikacija. Olakšavaju i ubrzavaju izradu svih ponavljajućih i repetitivnih dijelova u web razvoju, ujedno smanjujući vjerojatnost pojavljivanja fatalnih grešaka. Uz to pružaju standardizirane funkcije i klase iz korištenih biblioteka, prateći zadane standarde, čineći kod organiziranijim i čitljivijim.[56]

Postoje određeni kriteriji koje treba uzeti u obzir pri odabiru PHP razvojnih okvira za razvoj aplikacije. Neki od glavnih značajki koje dobar razvojnih okvir mora imati za optimalni razvoj su: MVC arhitektura, velik raspon raspoloživih biblioteka kako bi se uštedilo što više vremena, mogućnost testiranja, performanse i skalabilnost te jednostavnost i popularnost. Zavisno o potrebama projekta na kojem se radi mogu se odabrati razvojnih okviri koji su komplicirani jer pružaju kompleksnije značajke, ali pod cijenu dugotrajnijeg i napornijeg procesa upoznavanja s razvojnim okvirom, što se onda treba razmotriti samo u krajnjoj potrebi. Popularnost je pak bitna zbog veće količine dostupnih biblioteka, detaljno raspisane dokumentacije, ali naravno i zbog potpore zajednice koja je bitna tijekom procesa učenja, napredovanja ili za bilo koji drugi oblik suradnje.[57] Naposlijetku, kriterij koji je od velike važnosti pri odabiru razvojnog okvira je održavanje i podrška razvojnog tima tog okvira. Bitno je odabrati razvojni okvir s redovitim održavanjima i ažuriranjima kako bi se ispravile potencijalne greške, popravile sigurnosne značajke i predstavila daljnja napredovanja, omogućujući stabilnost i dugotrajnost projekta.

U velikom rasponu dostupnih PHP razvojnih okvira teško je odrediti koji odabrati za analizu ili primjenu. Svaki od njih ima svoje prednosti i mane. Neki su napravljeni za velike i kompleksne projekte što ih čini težim za učenje, dok neki drugi nude manje značajki, ali su zato brži. Postoje mnogobrojni PHP razvojni okvira koje programeri mogu odabrati za svoj projekt. No ipak se u literaturi često izdvajaju sljedeći: Laravel, Symfony, CodeIgniter, Yii, CakePHP Laminas, FuelPHP. Phalcon, PHPPixie, Slim, Lumen, Fat-Free Framework. Pet najpopularnijih će se detaljnije objasniti u sljedećim poglavljima, dok za navedene, manje popularne će se navesti njihove glavne funkcionalnosti. Laminas, poznatiji kao bivši Zend razvojni okvir nudi opsežne biblioteke i stavlja fokus na značajke sigurnosti. Podržava MVC arhitekturu i opsežan niz bazi

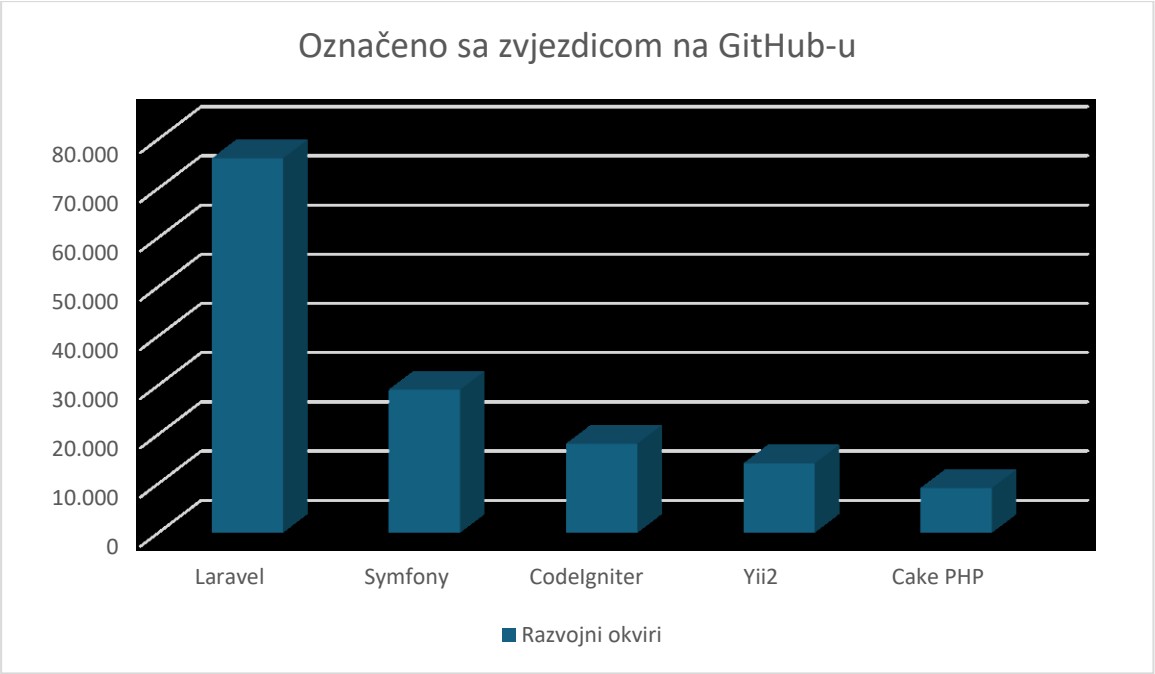
[56] Kuflinski, Yaroslav. PHP Frameworks Explained in 5 Simple Questions. 2019. URL: https://hackernoon.com/php-frameworks-explained-in-5-simple-questions-uvz31i7 (2024-07-18)
[57] Isto.

podataka, te nudi modularnu arhitekturu. FuelPHP također podržava MVC i modularnu arhitekturu, a karakterizira ga čista sintaksa. Njegove prednosti uključuju moćan sustav objektno-relacijskog mapiranja, sigurnosne značajke i sustav za validaciju obrazaca i podataka, dok je nedostatak manja podrška zajednice i rijetka ažuriranja. Phalcon je razvojni okvir koji je napisan kao C-ekstenzija integrirana u PHP kako bi se poboljšale performanse, a predstavlja odličan izbor za programere kojima je fokus sigurnost i fleksibilnost. Uključuje podršku za PostgreSQL, MySQL i SQLITE, te se lako integrira s drugima bibliotekama čineći ga jednostavnim za korištenje. PHPPixie je razvojni okvir dizajnira za brzu izradu jednostavnih mrežnih aplikacija. Karakterizira ga MVC arhitektura, alati za sigurnost te vlastiti query builder i ugrađeni ORM. Slim je mikro PHP razvojni okvir što znači da pruža minimalnu podršku za HTTP zahtjeve i njihovo preusmjeravanje na odgovarajuće kontrolere. Prednosti uključuju: integracija s PHP komponentama, upravljanje sesijama i enkripciju kolačića što omogućuje bolju sigurnost. Nedostatci su mal ekosustav i manjak naprednih značajki što odmah znači da nije prikladan za kompleksne aplikacije. Lumen je također mikro razvojni okvir, temelji se na Laravelu, a posebno je dizajniran za izradu manjih aplikacija i mikroservisa. Prednosti uključuju kompatibilnost s Laravel značajkama i komponentama, brz razvoj mikroservisa i jednostavna nadogradnja. Nedostatci su ograničene značajke ako se uspoređuje s potpunim Laravel razvojnim okvirom, a budući da mu je fokus na mikroservisima i API-jima znači da nije prikladan za velike aplikacije. Te naposlijetku, Fat-Free je mikro razvojni okvira, nema složenu strukturu direktorija niti potrebu za postavljanje Composera. Budući da ima minimalističku strukturu, omogućuje brzo razvijanje aplikacije, podržava validaciju podataka i nudi alate za sigurnosne značajke, ima ORM i podržava SQL i NoSQL baze podataka. Najveći nedostatak navedenih, manje korištenih razvojnih okvira, predstavlja ograničena zajednica. Manja zajednica, ekosustav i prihvaćenost u industriji uvelike otežavaju pronalaženje potrebnih resursa i podrške prilikom izrade mrežnih aplikacija.[58]

U narednim poglavljima prezentirati će se značajke pet najpopularnijih PHP razvojnih okvira, temeljno na rezultatima prema GitHubu i Packagistu, repozitoriju gdje PHP programeri mogu objaviti svoje biblioteke, razvojne okvire i sl.[59] (Slika 6).

[58] Upendraprasadmahto. Top 12 PHP Frameworks for Web Development in 2024. 2024. URL: https://medium.com/@upendraprasadmahto652/top-12-php-frameworks-for-web-development-in-2024-81a43874072a (2024-09-18)

[59] Ribeiro, Claudio. What Are the Best PHP Frameworks for 2023? 2024. URL: https://www.sitepoint.com/best-php-frameworks/#whentouseaphpframework (2024-07-24)



Slika 6. Prikaz broja projekata razvojnih okvira označenih zvjezdicom na GitHub-u

Direktniji prikaz popularnosti određenog razvojnog okvira može se vidjeti na stranicama Packagist-a koji pruža i statistike poput ukupnog broja preuzimanja određenog razvojnog okvira, statistiku u zadnjih 30 dana, čak i 24 sata te pregled popularnosti te graf prikaza popularnosti kroz godine. (Tablica 1.) No i prema podacima iz drugog izvora, Packagista, može se doći do istog zaključka, Laravel je uvelike najpopularniji razvojni okvir kojeg programeri biraju na dnevnoj bazi. [60]

Razvojni okvir:	Sveukupni broj preuzimanja:
Laravel	352 035 678
Symfony	80 926 879
Yii	22 959 867
CakePHP	13 749 418
CodeIgniter	2 181 202

Tablica 1. Sveukupni broj preuzimanja razvojnih okvira prema Packagist-u

[60] Crozat, Benjamin. Is PHP Dead? Usage Statistics and Market Share for 2024. 2024. URL: https://benjamincrozat.com/php-is-dead-2024 (2024-07-25)

5.1 Laravel

Laravel je razvojni okvir u otvorenom pristupu koji je nastao 2011. godine rezultatom rada Taylor Otwell-a, a zadnja verzija, Laravel 11, je izašla 2024. godine. Od samih početaka, kao što se moglo vidjeti iz prethodnog poglavlja, Laravel je preuzeo ulogu jednog od najpopularnijih PHP razvojnih okvira. Njegov cilj je vrlo jednostavan, poboljšati iskustvo tijekom razvoja mrežnih aplikacija, zadržavajući visoki standard kvalitete. Popularnost razvojnog okvira rezultirala je u impresivnom broju foruma, edukacijskih videa i drugih diskursa na internetu, a baš zbog velike popularnosti, Laravel ima bogatu biblioteku i unaprijed predefinirane funkcionalnosti koje uklanjaju potrebu da se često korištene značajke izgrađuju iz temelja. [61]

Neke od glavnih značajki zbog kojeg je toliko popularan uključuju arhitekturu koja je bazirana na MVC modelu što omogućuje pisanje čitljivog i jasnog koda čime se poboljšavaju performanse i skalabilnost aplikacije.[62] Zatim ORM koji se u Laravelu zove Eloquent ORM, a omogućuje interakciju s bazom podataka koristeći PHP sintaksu, što rezultira elegantnim načinom za upravljanje bazom podataka tako što programerima olakšava interkaciju s tablicama omogućujući im manipulacije poput umetanja, ažuriranja i brisanje podataka bez potrebe za poznavanje SQL-a. Zatim sustav za upravljanje predlošcima (templating): Blade, koji dolazi automatski postavljen u sklopu razvojnog okvira, a omogućuje lakšu sintaksu za kreiranje dinamičnih i kompliciranijih izgleda mrežne stranice. Još jedan benefit iznimno popularnog razvojnog okvira je **veliki broj paketa u otvorenom pristupu**, a neki od njih koriste Blade za prikaz, takvi unaprijed pripremljeni paketi uvelike olakšavaju implementaciju raznih proces poput na primjer Breeze – paketa za autentikaciju ili Cashier – paket za upravljanje naplatama. Te naravno, velika prednost je i automatski integrirani alat: **Artisan Command Line Interface**. Njegova glavna zadaća je pomoći programerima da na brz i lak način izvedu zadatke i operacija kao što su generiranje koda, pokretanje testova, pokretanje poslužitelja ili samo olakšan rad s podacima.[63]

[61] Nauman, Areeba. Top 5 PHP Frameworks 2024: Transform Web Development. 2024. URL: https://cyberpanel.net/blog/top-5-php-frameworks (2024-07-28)
[62] Isto.
[63] Top Backend Frameworks for Web Development in 2022: PHP Frameworks.. 2022. URL: https://pwr-code.com/blog/top-backend-frameworks-for-web-development-in-2022-php-frameworks) (2024-28-07)

Neki od nedostataka uključuju nešto opsežniji proces upoznavanja s okvirom, pruža osnovne značajke dok se sve ostale funkcionalnosti moraju dodavati putem paketa i ekstenzija te ponekad zna imati problema s ažuriranjima na novije verzije, uzrokujući bugove. [64]

5.2 Symfony

Symfony je najpopularniji odabir razvojnog okvira za potrebe projekata velikih razmjera. Osmislio ga je Fabien Potencier 2005. godine, a zadnja verzija v7.1 je izašla 2024. godine. Ima poprilično veliki ekosustav, koji se najčešće koristi za izradu kompleksnijih mrežnih aplikacija i mikroservisa, što ga čini optimalnim za iskusnije PHP korisnike. Iako je Symfony nešto teži za naučiti, on također okuplja veliku zajednicu, od otprilike 300 000 programera koji podržavaju proces upoznavanja s okvirom.[65]

Glavne značajke predstavljaju: modularnu arhitekturu koja omogućuje programerima fleksibilnost i povećanu skalabilnost, jer programeri mogu koristiti samo komponente razvojnog okvira koje su im potrebne, a zanemariti ostatak. Zatim Doctrine ORM, koji predstavlja alat za bolje upravljanje bazom podataka. Također ima sustav za predloške Twig kako bi programeri mogli koristiti dinamične predloške za potrebe svojih aplikacija. Prednost Symfony razvojnog okruženja je i mogućnost korištenja mnogobrojnih paketa i ekstenzija koje se implementiraju za razne potrebe, u rasponu od osnovnih funkcionalnosti do pružanja podrške za sve baze podataka. Budući da je Symfony namijenjen za potrebe većih projekata, jedna od najbitnijih značajki je dugoročna podrška tj. **LTS** (Long-Term Support) koji služi kao garancija da će razvojni okvir dobivati redovna ažuriranja i potrebnu podršku tijekom određenog razdoblja kako bi se omogućila očekivana stabilnost na projektu.[66] Zajedno s tim ulazi i podrška za testiranje, kako bi programerima omogućili bolju kvalitetu i pouzdanost krajnjeg proizvoda. Na kraju krajeva, Symfony se predstavlja kao siguran, pouzdan i multifunkcionalan izbor. Sensiolabs, tvrtka pod kojom je Symfony u vlasništvu, jamče podršku do tri godine za verzije ovog razvojnog okvira uključujući razne vodiče, certifikate i općenito podršku za sva pitanja.[67]

[64] Isto.
[65] Nauman, Areeba. Nav. dj.
[66] Isto.
[67] Top Backend Frameworks for Web Development in 2022: PHP Frameworks. Nav. dj.

Negativne strane uključuju: kompleksnost okvira i dugotrajno učenje, česta potreba za drugim tehnologijama kako bi se ostvarili određeni ciljevi što lako može dovesti do sukoba verzija, problema pri ažuriranja ili sigurnosnih briga te veća potrošnja memorije i sporije performanse upravo jer je toliko kompleksan. [68]

5.3 CodeIgniter

Rick Elis je 2006. predstavio razvojni okvir CodeIgniter, a posljednja verzija 4.5.4 je također izašla 2024. godine. Ovo je MVC razvojni okvir koji je izrađen za potrebe programera kojima je cilj koristiti jednostavan alat, ali koji i nudi široki raspon funkcionalnosti, dok pruža i impresivne rezultate brzine prilikom izvođenja aplikacije. Kao i kod drugih popularnih razvojnih okvira, korisnici mogu očekivati zadovoljavajuću razinu podrške u obliku dokumentacije, foruma, edukacijskih videa i foruma. [69]

Neki od glavnih razloga zašto programeri odabiru CodeIgniter razvojni okvir predstavlja njegova iznimno mala veličina preuzimanja, razvojnih okvir zauzima samo 1.1 MB, dok je korisnički priručnik 1.6MB. Baš zato što je tako male veličine ukazuje i na činjenicu da koristi manje resursa usporedno s drugim razvojnim okvirima, što je jedna od stvari koje ga čine savršenim za brz razvoj aplikacija. Osim toga, nije potrebna skoro nikakva konfiguracija, treba samo postaviti bazu podataka, a sve ostalo dolazi već unaprijed postavljeno. Iz veličine priručnika, može se vidjeti i da je dokumentacija vrlo jasno raspisana, sadrži vodič za navigiranje kroz razvojni okvir i objašnjava njegove komponente. Još jedna zanimljivost kod CodeIgniter je ta što podržava MVC arhitekturu, čak i potiče njeno korištenje, zbog svih razloga koji su objašnjeni i u prijašnjim poglavljima, ali ukoliko programer ne želi koristit ovu arhitekturu, sam razvojni okvir ga ni ne tjera na to. Osim toga, na službenoj stranici CodeIgniter-a, kao veliku prednost uzimaju i sigurnosne značajke koje razvojni okvir nudi, kao što su zaštita od CSRF (Cross-Site Request Forgery) i XSS (Cross-Site Scripting) napada. Zaštita je implementirana koristeći skrivene tokene u POST zahtjeve i s kojom se filtriraju ulazni podaci prije prikaza korisnicima.[70] Te na kraju, prednost CodeIgniter-a uključuje

[68] Isto.
[69] Shah, Jigar.The Most Popular PHP Framework to Use in 2023. 2024. URL: https://wpwebinfotech.com/blog/php-framework-comparison/ (2024-08-01)
[70] CodeIgniter. URL: https://codeigniter.com/ (2024-08-01)

i fleksibilan sustav usmjeravanja (routing). Omogućuje stvaranje prilagođenih ruta za potrebe svake aplikacije kako bi se ostvarila bolja preglednost. [71]

Nedostatci uključuju: nedostatak sustava za predloške, već korisnici sami moraju pisati kod ili integrirati neki od alata s treće strane, zbog svoje manje veličine ima i limitirane integrirane funkcionalnosti, isto tako u usporedbi s ekstremno popularnim Laravel i Symfonijem ima manji ekosustav ekstenzija, te veće razmake između svake nove verzije. [72]

5.4 Yii

Yii je jedan od starijih razvojnih okvira, čije je alpha verzija izašla 2006, a službena prva verzija, 2008. godine.[73] Nakon toga, u pothvatu suočavanja s određenim arhitekturnim problemima originalnog okvira, nastaje Yii 2.0 koji je izašao 2014, a njegova zadnja verzija 2.0.51 je izašla 2024. godine. Yii2 okvir se često preporuča kao najbolja opcija za početnike upravo zbog svog jasnog i direktnog priručnika s kojim se lako koristiti i pomoću kojega se brzo može početi raditi na vlastitom projektu.[74]

Osim toga što ima iznimno korisnu dokumentaciju koja pomaže korisnicima da što prije krenu upotrebljavati ovaj razvojni okvir, još jednu veliku prednost u tome pogledu predstavlja i prilagodljivost Yii2 okvira. Lako se može prilagoditi potrebama malih, srednjih ili velikih projekata jer ima veliki izbor funkcionalnosti za ponuditi, što skraćuje vrijeme razvoja.[75] Tako prilikom same instalacije nudi dvije opcije: osnovnu i naprednu. Razlika u njima je to što napredna opcija ima sustav za autentifikaciju korisnika. Ovakve podijele omogućuju programerima da započnu s jednostavnijim kodom, te na temelju njega samo nadograđuju svoje znanje.[76] Sljedeći benefit po kojem se Yii ističe od drugih je učinkovit način cashiranja podataka koji omogućuje veću brzinu odgovora i učinkovitost aplikacije. Ovaj okvir podržava četiri takva mehanizma, cashing: podataka, fragmenata, HTTP-a i stranice, s komponentama cDummyCache, cMemCache i cFileCache. Kada je u pitanju komunikacija s bazom podataka, Yii2 se oslanja na API za pristup

[71] Nauman, Areeba. Nav. dj.
[72] Ali, Inshal. 10 Best PHP Frameworks for Web Development in 2024. 2024. URL: www.cloudways.com/blog/best-php-frameworks/ (2024-08-02)
[73] Brotherton, Claire. The Most Popular PHP Frameworks to Use. 2023. URL: kinsta.com/blog/php-frameworks/#symfony (2024-08-02)
[74] Shah, Jigar. Nav. dj.
[75] Top Backend Frameworks for Web Development in 2022: PHP Frameworks. Nav. dj.
[76] Ribeiro, Claudio. Nav. dj.

relacijskim bazama podataka, a podržava ih ukupno sedam, uključujući najpopularnije MySQL i MariaDB, a ukoliko programeri odluče raditi i s Active Rocordom, obrascom za rad s bazama podataka, mogu uvelike smanjiti broj SQL upita[77] Naposlijetku, kada govorimo o prednostima ovog razvojnog okvira, potrebno je spomenuti i Gii, alat za kreiranje često korištenih odlomaka koda, što značajno ubrzava razvoj same aplikacije. [78]

Glavni nedostatak Yii2 razvojnog okvira su rijetka ažuriranja i nešto teža prilagodba već postojećih funkcionalnosti. [79]

5.5 CakePHP

CakePHP je nastao 2005. godine, a napravio ga je Michal Tatarynowicz sa namjerom da revolucionalizira svijet web razvoja. Otkako je izašao predstavlja razvojni okvir koji čini proces izrade aplikacija bržim i jednostavnijim, a njegova zadnja verzija, CakePHP 5 je izašla 2023. godine. Budući da je ovaj okvir star preko 15 godina, ima veliku zajednicu za podršku.[80]

Posebnost CakePHP-a je u odabiru tzv. pristupa 'konvencije iznad konfiguracije.' To znači da se ovaj MVC razvojni okvir pokušava što manje oslanjati na potrebu za ručnim konfiguracijama, implementirajući utvrđene konvencije, što programerima omogućuje brži rad jer sam okvir već unaprijed dolazi s postavljenom strukturom koja se mora pratiti. To ne znači da CakePHP nije fleksibilan, čak štoviše, omogućuje stvaranje raznih funkcionalnosti koje nisu dostupne odmah u preuzetom paketu. Integrirajući biblioteke i omogućujući stvaranje novih komponenti daje programerima sve potrebne alate kako bi došli do rješenja specifičnima za potrebe pojedinog projekta. Uz to ima i ugrađeni ORM koji, naravno, pojednostavljuje interakciju s bazama podataka, povećava dosljednost koda, ali dolazi i s ugrađenom autentikacijom i validacijom podataka. Te naposlijetku, velika prednost su i ugrađene značajke sigurnosti, koje uklanjaju potrebu za samostalnim kodiranjem, ujedno štedeći vrijeme, dok osiguravaju zaštitu aplikacije. U to se uključuje: zaštita od CSRF napada, razne validacije unosa i upravljanje sesijama. Sve ove značajke čine CakePHP dobrim odabirom za razvoj sigurnih aplikacija, koje su fleksibilne, jednostavne i skalabilne. Kao i prethodni razvojni okvir, i CakePHP ima generator koda koji se zove Bake.

[77] Top Backend Frameworks for Web Development in 2022: PHP Frameworks. Nav. dj.
[78] Shah, Jigar. Nav. dj.
[79] Top Backend Frameworks for Web Development in 2022: PHP Frameworks. Nav. dj.
[80] Nauman, Areeba. Nav. dj.

Također ubrzava izradu koda, a funkcionalnosti mu uključuju: automatizirano kreiranje modela, kontrolera i pogleda te drugih komponenti aplikacije. [81]

Nedostatci ovog razvojnog okvira uključuju pad u popularnosti tijekom zadnjih godina, teži proces učenja okvira jer se strogo drži svojih konvencija, te se označava kao razvojni okvir koji nije toliko prilagođen novom korisniku . Zatim, u nekim slučajevima, mijenjanje predefiniranih predložaka može rezultirati problemima u samoj MVC arhitekturi, a najveći nedostatak predstavlja jednosmjeran sustav usmjeravanja. [82]

[81] Isto.
[82] Shah, Jigar. Nav. dj.

5.6 Usporedna tablica značajki razvojnih okvira

U ovoj analizi oslanjamo se na različite kriterije kao što su arhitektura, popularnost, težina učenja, podrška zajednice, dokumentacija, ORM, sustav usmjeravanja, predložak, kompatibilnost s bazom podataka, testiranje, sigurnost, proširenja i performanse, koristeći dostupne online resurse za usporedbu.[83]

	Laravel	Symfony	CodeIgniter	Yii2	CakePHP
Godina izd.:	2011	2005	2006	2014	2005
Popularnost:	Vrlo popularan	Popularan	Popularan	Popularan	Pad popularnosti
Podrška zajednice:	Aktivna zajednica	Aktivna zajednica	Aktivna zajednica	Aktivna zajednica	Aktivna zajednica
Dokumentacija:	Izvrsna	Izvrsna	U redu	Izvrsna	U redu
Arhitektura:	MVC	Full Stack	MVC	MVC	MVC
ORM:	Eloquent	Doctrine	Ugrađeni ORM (Active Record)	Yii ORM	CakePHP ORM
Sustav usmjeravanja:	Moćan	Fleksibilan	Jednostavan	Fleksibilan	Fleksibilan i zaostao
Sustav predložaka:	Blade	Twig	Prilagođeni	Prilagođeni	Prilagođeni

Tablica 2. Usporedba značajki pet najpopularnijih PHP razvojnih okvira

[83] Top Backend Frameworks for Web Development in 2022: PHP Frameworks. Nav. dj.

	Laravel	Symfony	CodeIgniter	Yii2	CakePHP
Baze podataka:	MySQL, PostgreSQL, SQLite, SQL Server	MySQL, PostgreSQL, SQLite, SQL Server, Oracle	MySQL, PostgreSQL, SQLite, Oracle, Microsoft BI	MySQL, PostgreSQL, SQLite, SQL Server, Oracle	MySQL, PostgreSQL, SQLite, SQL Server, Oracle, MariaDB
Generiranje koda:	Artisan CLI	Symfony CLI	Spark CLI	Gii	Bake CLI
Sigurnosne značajke:	Zaštita od CSRF napada, enkripcije, hashing Middleware	Autentifikacija, autorizacija, enkripcija, zaštita od CSRF-a	Autentifikacija, autorizacija, pristup temeljen na ulozi, zaštita od CSRF-a	Kontrola pristup na osnovi uloga, zaštita od CSRF-a	Kontrola pristup na osnovi uloga, zaštita od CSRF-a
Proširenja:	Veliki ekosustav biblioteka i paketa	Širok raspon biblioteka i paketa	Adekvatan broj paketa i biblioteka	Adekvatan broj paketa i biblioteka	Adekvatan broj paketa i biblioteka

Tablica 3. Usporedba značajki pet najpopularnijih PHP razvojnih okvira

Razvojni okvir	Zadnja verzija:	Brzina izvršenja (ms)	Brzina odgovora (ms)
Laravel	11	50-60	100
Symfony	7	40-50	90
CodeIgniter	4.5	30-40	70

Tablica 4. Usporedba performansi tri najpopularnija PHP razvojna okvira[84]

[84] Hossain, Arafat Ar. Top 3 PHP Frameworks: Speed, Response Time, and Efficiency Compared. 2024. URL: https://dev.to/arafatweb/top-3-php-frameworks-speed-response-time-and-efficiency-compared-25bi#performance-comparison-table (2024-08-05)

6. Usporedna analiza procesa izrade aplikacije

U ovome poglavlju fokus će biti na usporednoj analizi dva najpopularnija PHP razvojna okvira – Laravel-a i Symfony-a, tako što će se izraditi CRUD aplikacije u svakom. Usporedna analiza omogućuje dublje razumijevanje teme na temelju jasno istaknutih sličnosti, ali i razlika prilikom izrade aplikacije. Navedeni razvojni okviri su odabrani upravo zbog svoje popularnosti u zajednici i svih povezujućih benefita koji su već detaljno prikazani u prethodnim poglavljima. Razvojni okviri su česti odabir prilikom razvoja aplikacija zbog čega je važno utvrditi njihove glavne značajke i izvan isključivo teorijskog pogleda. Budući da svaki razvojni okvir ima jedinstvenu arhitekturu i način funkcioniranja potrebno ih je usporediti kako bi znali prepoznati njihove prednosti i nedostatke te na temelju njih donijeti odluku o najboljem izboru.

Cilj analize je usporediti i dati izravan uvid u proces izrade ključnih dijelova gotovo svake aplikacije koristeći Laravel i Symfony razvojne okvire te dobiti odgovor koji od njih predstavlja bolji odabir pri kriterijima jednostavnosti, fleksibilnosti i samog iskustva razvoja aplikacije. U Laravel-u će se izraditi CRUD aplikacija za bilješke 'Post-it', a u Symofny-u 'Library' za knjižnicu gdje korisnici mogu vidjeti, stvoriti, ažurirati ili obrisati bilješke/knjige. Tijekom procesa izrade aplikacije, bilježiti će se detalji vezani u proces i način implementacije koda uz njihovo objašnjenje za čije će se potrebe primarno analizirati službena dokumentacija odabranih razvojnih okvira, ali i drugi relevantni izvori. Ključni dijelovi aplikacije na temelju kojih se radi usporedna analiza najprije uključuju postavljanje projekta gdje se definiraju potrebni zahtjevi za stvaranje samog projekta te prikazuju procesi njegovog postavljanja. Zatim se u svakom razvojnom okviru opisuje kreiranje ruta i kontrolera ključnih za povezivanje korisničkih zahtjeva s odgovarajućim funkcijama unutar aplikacije. Nakon čega se uspoređuju način izrade modela i rješenja razvojnih okvira za rad s bazom podataka, uključujući opis procesa migracije, ali i način implementacije objektno-relacijskog mapiranja. Nakon postavljanja logike aplikacije, analiza će se usmjeriti na usporedbu načina prikaza podataka u odabranim razvojnim okvirima. Usporedba se fokusira na sustave za predloške koje okviri nude te koliko ih je lako koristiti. Osim toga usporediti će se i osnovne dostupne sigurnosne značajke koje razvojni okviri mogu implementirati kao što je zaštita od zlonamjernih napada. Naposlijetku, usporediti će se lakoća implementacije paketa koji uvelike olakšavaju kreiranje funkcionalnosti kao što je na primjer autentifikacija korisnika.

6.1 Laravel

6.1.1 Postavljanje projekta

Prije početka izrade 'Post-it' CRUD Laravel aplikacije, potrebno je ispuniti sljedeće zahtjeve:

- Najnoviju verziju PHP-a (verzija 8.3.10)
- Razvojni poslužitelj (XAMPP)
- Najnoviju verziju Composer-a (verzija 2.7.8)
- Node.js (verzija 20.17.0) i npm (verzija 10.8.2)
- Git (verzija 2.46.0)
- IDE alat (VS Code) i ekstenzije

Za postavljanje radnog okruženja svi potrebni alati, su bili preuzeti sa službenih stranica. Kako bi počeli rad s Laravelom 11, potrebno je imati najnoviju verziju PHP-a da bi se dosegla optimalna razina performansi i sigurnosti. XAMPP predstavlja lokalno razvojno okruženje za kojeg je bitno znati da se sastoji od HTTP poslužitelja, baze podataka i interpretera za PHP jezik. Composer omogućuje instaliranje i ažuriranje paketa i biblioteka potrebnih za projekt. Potrebno je instalirati i Git koji pomaže Composeru u preuzimanju, ažuriranju paketa i općenito za upravljanje ovisnostima, te naravno za verzioniranje koda. Node.js i NPM omogućuju rad s front-end alatima kao što je Vite koji služi za ubrzaniji proces izgradnje i optimizaciju front-end resursa. Korištenje IDE-a poput VS Code-a s odgovarajućim ekstenzijama omogućuje učinkovitost razvoja i podršku za različite tehnologije, a neki od korištenih za ovaj primjer uključuju: Laravel Blade Formatter, PHP devsense, Prettier code formater itd.[85]

Nakon postavljanja radnog okruženja može se nastaviti na kreiranje novog Laravel projekta, a to je najlakše postići sa Composer-ovom naredbom create-project (Slika 7.)

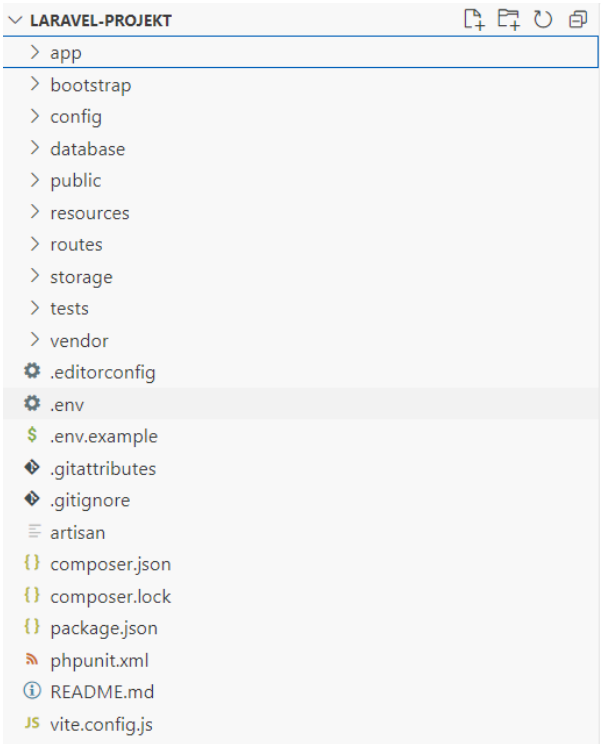
```
C:\Users\Stella>composer create-project laravel/laravel laravel-projekt
```

Slika 7. Izrada Laravel projekta pomoću composer-a

[85] Setting up Laravel on Windows 10: A Comprehensive Guide. 2024. URL: https://hackmd.io/@editorial/install-laravel-windows-10 (2024-08-05)

Nakon izvođenja ove naredbe preuzimaju se i instaliraju Laravel paketi u direktorij laravel-projekt, zatim se konfiguriraju i postavljaju potrebni direktoriji i datoteke, postavljaju se aplikacijski ključevi i migracije potrebne za definiranje strukture baze podataka.

Zatim se može pogledati struktura direktorija projekta u VS Code IDE alatu. (Slika 8). Direktorij **'app'** sadržava sve klase i osnovni kod cijele aplikcije, dok direktorij **'bootstrap'** sadrži datoteku app.php kojom se pokreće framework zajedno s cache direktorijem koji pohranjuje generirane podatke razvojnog okvira. Direktorij **'config'** sadrži sve konfiguracijske datoteke. Vrlo bitan direktorij, **'database'** sadrži sve migracije baze podataka, modele i nešto što se zove sjeme tj. seed. 'Seeding' označava proces popunjava baze podataka s inicijalnim podacima ili testnim podacima kako bi na brzi način dodali podatke bez ručnog unošenja. Ubrzavaju razvojni proces. Direktorij **'public'** sadrži datoteku index.php koji sadrži JS kod, CSS i assete. Direktorij **'routes'** sadrži sve definicije ruta. Direktorij **'storage'** sarži pohranjene log-ove, Blade predloške i cach na temelju datoteka. Te na kraju **'vendor'** direktorij koji je zadužen za pohranu svih Composerovih ovisnosti. [86]



Slika 8. Struktura direktorija Laravel projekta

[86] Directory Structure. Laravel. URL: https://laravel.com/docs/11.x/structure (2024-08-05)

Zatim je potrebno pokrenuti poslužitelj sa naredbom **'php artisan serve'** koji u Laravelu omogućuje razvoj aplikacije na lokalnom računalu te nam omogućuje da vidimo aplikaciju u pregledniku upisivanjem localhost:8000. Artisan je CLI koji je automatski uključen u Laravel razvojni okvir, a kao što se može vidjeti iz strukture direktorija, nalazi se u root direktoriju spremljen kao skripta pod nazivom **'artisan'** i sadrži niz naredbi koje uvelike olakšavaju razvoj aplikacije. [87] Nakon čega je potrebno instalirati npm i dati naredbu npm run dev koji omogućuje razvojno okruženje s Vite-om kako bi se efikasno upravljalo s resursima, a sve promjene automatski ažurirale i prikazale.

6.1.2 Kreiranje ruta i kontrolera

Prije nego što krenemo s definiranjem željenih ruta, potrebno je stvoriti kontrolere jer kontroleri, između ostalog i grupiraju povezane akcije i upravljaju rutama, što čini proces lociranja uvelike olakšanim, dok poboljšava samu logiku aplikacije. Za kreiranje kontrolera će se također koristiti artisan sa naredbom: **php artisan make:controller HomeController** koji će stvoriti novu datoteku kontroleru na sljedeću putanju 'app/http/controllers' (Slika 9.) Nakon čega se mogu definirati rute u datoteci web.php koje koriste definirane metode kontrolera. U ruti se nalazi naziv kontrolera: HomeController, klasa, i naziv metode: home. Na kraju se nalazi i ime rute koje se također zove 'home' (Slika 10.)

```
PS C:\Users\Stella\laravel-projekt> php artisan make:controller HomeController

  INFO  Controller [C:\Users\Stella\laravel-projekt\app\Http\Controllers\HomeController.php] created successfully.
```

Slika 9. Izrada kontrolera

```
use App\Http\Controllers\HomeController;
use Illuminate\Support\Facades\Route;

Route::get('/', [HomeController::class, 'home'])->name('home');
```

Slika 10. Prikaz ruta u Laravelu

[87] Artisan Console. Laravel. URL: https://laravel.com/docs/11.x/artisan (2024-08-06)

Kako bi izradili potrebne rute i kontroler za aplikaciju 'Post-it', naprije je potrebno stvoriti **PostController** također artisan naredbom: **php artisan make:controller PostController --resource --model=Post** gdje –resource Artisanu govori da generira resource controller koji sadrži osnovne metode potrebne za CRUD. Što znači kada se stvori ovaj kontroler da će već imati spremno 7 unaprijed definiranih metoda: index, create, store, show, edit, update i destroy. Dok –model znači da se kontroler automatski povezuje s modelom Post, što omogućuje da se instanca modela Post u metodama edit, show, update i destroy automatski veže na ID. Što omogućuje direktan rad s modelom bez potrebe za dohvaćanjem podataka iz baze podataka. (Slika 11.)

```
63          /**
64           * Update the specified resource in storage.
65           */
66          public function update(Request $request, Post $post)
67          {
68              $data = $request->validate([
69                  'post' => ['required', 'string']
70              ]);
71
72              $post->update($data);
73              return to_route('post.show', $post)->with('message', 'Post updated successfully!');
74          }
75
76          /**
77           * Remove the specified resource from storage.
78           */
79          public function destroy(Post $post)
80          {
81              $post->delete();
82              return to_route('post.index')->with('message', 'Post deleted successfully!');
83          }
84      }
```

Slika 11. Prikaz metoda u PostController-u

Nakon kreiranja kontrolera, potrebno je definirati rute u 'web.php' datoteci koje odgovaraju akcijama definiranim u njemu. Svaka ruta se može pojedinačno definirati za odgovarajući proces ili se može koristiti **Route::resource** koji automatski generira rute za standardne CRUD operacije. (Slika 12.)

```php
<?php

use App\Http\Controllers\PostController;
use App\Http\Controllers\ProfileController;
use Illuminate\Support\Facades\Route;

Route::redirect('/', '/post')->name('dashboard');

Route::middleware(['auth', 'verified'])->group(function () {
    // Uncomment these routes if you prefer individual route definitions:
    // Route::get('/post', [PostController::class, 'index'])->name('post.index');
    // Route::get('/post/create', [PostController::class, 'create'])->name('post.create');
    // Route::post('/post', [PostController::class, 'store'])->name('post.store');
    // Route::get('/post/{id}', [PostController::class, 'show'])->name('post.show');
    // Route::get('/post/{id}/edit', [PostController::class, 'edit'])->name('post.edit');
    // Route::put('/post/{id}', [PostController::class, 'update'])->name('post.update');
    // Route::delete('/post/{id}', [PostController::class, 'destroy'])->name('post.destroy');

    Route::resource('post', PostController::class);
});
```

Slika 12. Prikaz ruta za CRUD

6.1.3 Kreiranje modela i migracija

Nakon kreiranja kontrolera slijedi kreiranja Eloquent modela i vezane migracije. Za ovo koristimo naredbu **php artisan make:model Post -m**. Što će napraviti model naziva Post, a oznaka '-m' predstavlja da želimo odmah stvoriti i migracijsku datoteku koja omogućuje definiranje strukture tablica u bazi podataka. Ovako se generiraju dvije datoteke: model datoteka i datoteka za migraciju. (Slika 13.) A nakon što se postave promjene u datoteku tablice kao što je postavljanje primarnog ključa tablice 'id', zatim 'post' i user_id' stupaca, potrebno je u terminalu pokrenuti naredbu **'php artisan migrate'** kako bi se promijene primijenile u tablici baze podataka. (Slika 14.)

```
PS C:\Users\Stella\laravel-projekt>
php artisan make:model Post -m

  INFO  Model [C:\Users\Stella\laravel-projekt\app\Models\Post.php] created successfully.

  INFO  Migration [C:\Users\Stella\laravel-projekt\database\migrations/2024_08_24_101402_create_posts_table.php] created successfully.

PS C:\Users\Stella\laravel-projekt>
```

Slika 13. Stvaranje Post modela

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('posts', function (Blueprint $table) {
            $table->id();
            $table->longText('post');
            $table->foreignId('user_id')->constrained('users');
            $table->timestamps();
        });
    }
```

Slika 14. Prikaz kreiranja i strukture tablice u bazi

6.1.4 Kreiranje pogleda s Blade-om

Kako bi se prikazao sadržaj, prema MVC-u sada je potrebno definirati Pogled. Pogled se nalazi u u Views direktoriju, a potrebna datoteka će se također stvoriti pomoću artisana. Naredba za stvaranje pogleda je: **'php artisan make:view post.create**, gdje post predstavlja direktorij, a create datoteku za Pogled. Na isti način će se napraviti i svi drugi pregledi: index, edit i show. Nakon kreiranja datoteka, potrebno je definirati odgovor koji se šalje pregledniku u kontroleru 'PostController' (Slika 15.) Kod 'view ('post.create')' upućuje Laravel koji odgovor treba prikazati i koji pogled pritom koristiti.

```php
1 reference | 0 overrides
public function create()
{
    return view('post.create');
}
```

Slika 15. Način prikaza pogleda pregledniku

Kako bi izbjegli nepotrebno ponavljanje koda, potrebno je je napraviti predložak izgleda stranice tj. layout. Unutar **'view'** direktorija, izrađujemo **'components '** direktorij koji sadržava **layout.blade.php** datoteku. U layoutu se definira osnovna struktura stranice, a za prikaz potrebnih

varijabli u Blade-u koriste se duple vitičaste zagrade. Specifična varijabla za prikaz sadržaja u layout-u se zove **$slot**. U generiranim pogledima: index, create, edit i show sadržaj stavljamo u komponentu <x-layout></x-layout> koja omogućava da što god je unutar ove komponente se prosljeđuje u $slot varijablu u layout datoteci.

Sljedeći korak predstavlja prikaz podataka iz baze podataka. Kako bi to napravili vraćamo se u PostController datoteku i definiramo varijablu $posts u kojoj se nalazi upit s kojim dohvaćamo podatke iz postavljene tablice i definiramo kako ih želimo prikazat. Da bi proslijedili varijable Pogledu, potrebno je staviti array kao drugi argument unutar view-a. (Slika 16.)

```
public function index()
{
    $posts = Post::query()->orderBy('created_at', 'desc')->paginate();
    return view('post.index', ['posts' => $posts]);
}
```

Slika 16. Prikaz dohvaćanja podataka

Nakon što je u kontroleru postavljen upit za prikaz post-it podataka, potrebno ih je iterirati i pokazi u pogledu. Za iteraciju kroz postove koristi se foreach direktiva. Za to se u Laravelu mogu koristiti direktive unutar Blade predložaka. Tako da, osim što Blade nudi opciju nasljeđivanja predložaka ima i vrlo pozitivan aspekt tzv. prečaca za uobičajene PHP strukture. Direktive predstavljaju posebnu sintaksu zaduženu za obavljanje zadataka kao što su uvjeti, petlje ili prikazivanje podataka. Uvijek započinju s simbolom **'@'**, a neki od primjera su: @if, @foreach, @include i @csrf. Direktivne na jednostavan i čitljiv način omogućuju integraciju PHP logike u Pogled. Ovakvi prečaci, osim što čine Pogled predloške preglednijim, omogućuju programerima da pišu manje koda i tako štede vrijeme. Za najlakšu implementaciju sigurnosnih funkcionalnosti koristi se direktiva @csrf. Laravel preporučuje korištenje ove direktive u svakoj definiranoj HTML formi aplikacije jer se njome automatski generira token kojim se provjerava legitimnost izvora. Ovo je sigurnosna mjera koja štiti od zlonamjernih napada u obliku neovlaštenih radnji, a automatski se uključuje u sve POST, PUT, PATCH i DELETE zahtjeve koje i koristimo u ovoj CRUD aplikaciji.[88] (Slika 17.)

[88] Blade Templates. Laravel. URL: https://laravel.com/docs/11.x/blade#blade-directives (2024-08-07)

```
<div class="posts">
    @foreach ($posts as $post)
        <div class="post">
            <div class="post-body">
                {{ Str::words($post->post, 5) }}
            </div>
            <div class="post-buttons">
                <a href="{{ route('post.show', $post) }}" class="post-edit-button">View</a>
                <a href="{{ route('post.edit', $post) }}" class="post-edit-button">Edit</a>
                <form action="{{ route('post.destroy', $post) }}" method="POST">
                    @csrf
                    @method('DELETE')
                    <button class="post-delete-button">Delete</button>
                </form>
            </div>
        </div>
    @endforeach
</div>
```

Slika 17. Prikaz direktiva u Blade-u

Kako bi se uredio izgled aplikacije koristi se css. Direktorij za css u kojem se nalazi datoteka app.css, nalazi se u 'resources' direktoriju, te se u njoj primjenjuju svi stilovi. Kako bi css datoteke bile primijenjene, one moraju biti smještene unutar 'public' direktorija jer moraju biti dostupne putem web-a, za što će se koristiti Vite poslužitelj. Za implementaciju se također koristi direktiva '@vite' unutar layout.blade.php, gdje se postavljaju datoteke koje želimo implementirati, što omogućuje da Laravel generira i automatski poveže resurse s HTML dokumentom. (Slika 18.)

```
@vite(['resources/css/app.css', 'resources/js/app.js'])
```

Slika 18. Učitavanje stilova i skripti putem Vite-a

Nakon implementiranja HTML i CSS koda za izgled aplikacije, potrebno je postaviti i rute za funkcionalnost unutar Pogled datoteka. Kada na primjer kreiramo novi post ili odustanemo od kreacije, rute usmjeravaju na točnu lokaciju. Unutar Blade predložaka rute se postavljaju u dvostrukim vitičastim zagradama kako bi se PHP izrazi prikazali direktno unutar HTML-a. Funkcijama se generira URL za traženu rutu koja je definirana unutar web.php datoteke koja se prikazuje u HTML-u. (Slika 19.)

```
<div class="post-buttons">
    <a href="{{ route('post.index') }}" class="post-cancel-button">Cancel</a>
    <button class="post-submit-button">Submit</button>
```

Slika 19. Prikaz ruta unutar Blade predloška

6.1.5 Registracija i login

Svrha paketa je mogućnost proširivanja funkcionalnosti aplikacije. Definiraju se kao zbirke koda koje omogućuju povećanje produktivnosti tako što koriste već postojeće resurse. Ovi paketi sadrže rute, kontrolere, poglede i konfiguracijski kod, kako bi poboljšali aplikacije izrađene u Laravel razvojnom okviru.[89] Kako bi se iskoristile prednosti koje ovaj razvojni okvir nudi za postavljanje registracije i logina koristiti će se Breeze, jednostavan paketa za implementaciju funkcionalnosti autentifikacije. Paket nudi razne mogućnosti kao što su prijava, registracija, potvrda i resetiranje lozinke te i 'profil' stranicu za ažuriranje korisničkog imena, e-maila i lozinke.[90]

Za instalaciju Breeza paketa koristi se sljedeća naredba: **'composer require laravel/breeze --dev'**, gdje dev ovisnost označava da se ne koristi u produkcijskom okruženju i da će se prikazati pod 'devDependencies' unutar **'package.json'** datoteke. Nakon što je dovršeno preuzimanje, potrebno je instalirati Breeze sa sljedećom naredbom: **'php artisan breeze:install'**. Ova naredba će automatski generirati osnovnu strukturu koda, poglede i rute za autentifikaciju te implementirati Tailwind CSS – okvir za izgradnju sučelja.

Nakon instalacije, aplikacija prestaje raditi jer je sadržaj unutar web.php i app.css datoteka obrisan, koje je potrebno ručno povratiti. Nakon instaliranja Breeza, u datoteci web.php gdje se nalaze sve rute, može se vidjeti da je ovaj okvir automatski konfigurirao middleware metode koje služe kao sloj između zahtjeva koji je postavio korisnik i odgovora kontrolera. Middleware **'auth'** osigurava da korisnik mora biti prijavljen za pristup, a **'verified'** označava da korisnik mora imati potvrđen e-mail, što znači da korisnik definiranim rutama može pristupiti samo ako je prijavljen. [91] (Slika 20.)

```
Route::middleware(['auth', 'verified'])->group(function () {

Route::get('/post',[PostController::class, 'index'])->name('post.index');
Route::get('/post/create', [PostController::class, 'create'])->name('post.create');
Route::post('/post',[PostController::class, 'store'])->name('post.store');
Route::get('/post/{id}',[PostController::class, 'show'])->name('post.show');
Route::get('/post/{id}/edit',[PostController::class, 'edit'])->name('post.edit');
Route::put('/post/{id}',[PostController::class, 'update'])->name('post.update');
Route::delete('/post/{id}',[PostController::class, 'destroy'])->name('post.destroy')
});
```

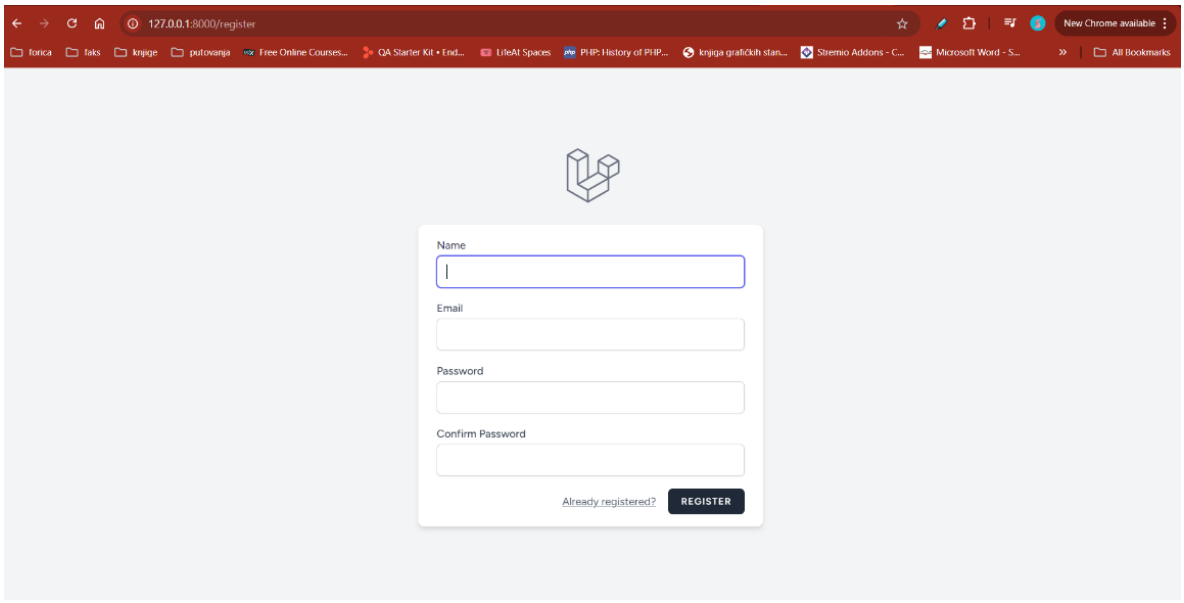
Slika 20. Middleware zaštita

[89] Package Development. Laravel. URL: https://laravel.com/docs/11.x/packages (2024-08-07)
[90] Starter Kits. Laravel. URL. https://laravel.com/docs/11.x/starter-kits#laravel-breeze (2024-08-08)
[91] Middleware. Laravel. URL: https://laravel.com/docs/11.x/middleware (2024-08-08)

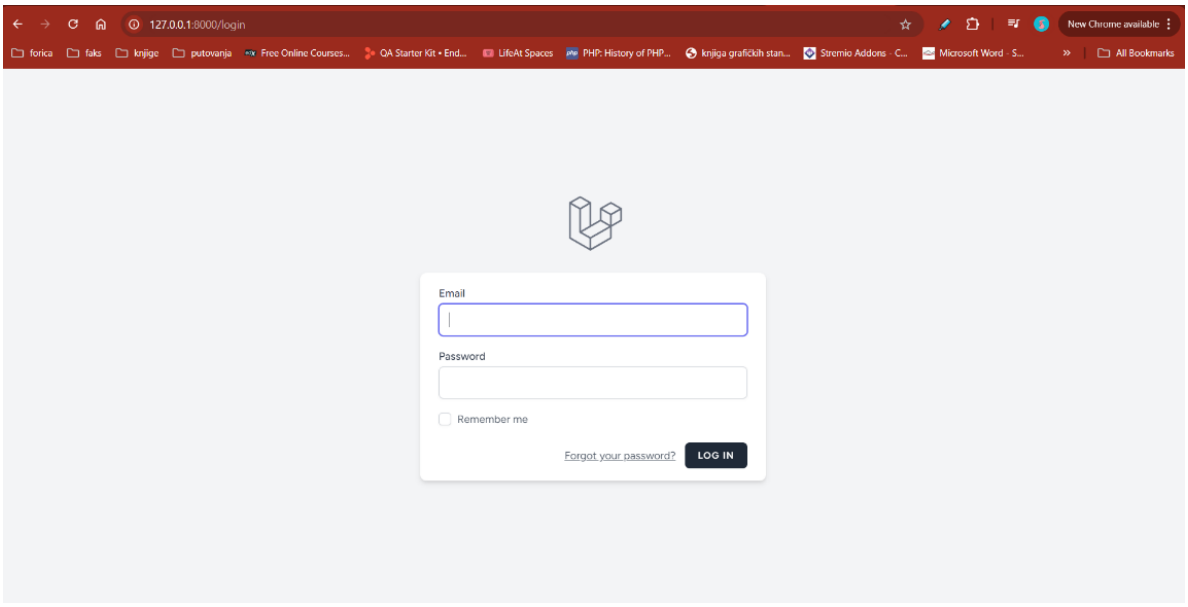
Za verifikaciju email adrese, u modelu User.php potrebno je dodati MustVerifyEmail sučelje koje omogućuje da se nakon registracije korisnika odmah i pošalje verifikacijski e-mail. Link za verifikaciju može se pronaći u datoteci **storage/logs/laravel.log**.

Te naposlijetku, potrebno je napraviti filtraciju postova prema korisniku što se odrađuje u PostControlleru postavljanjem uvjeta: **' ->where('user_id', request()->user()->id)**'.

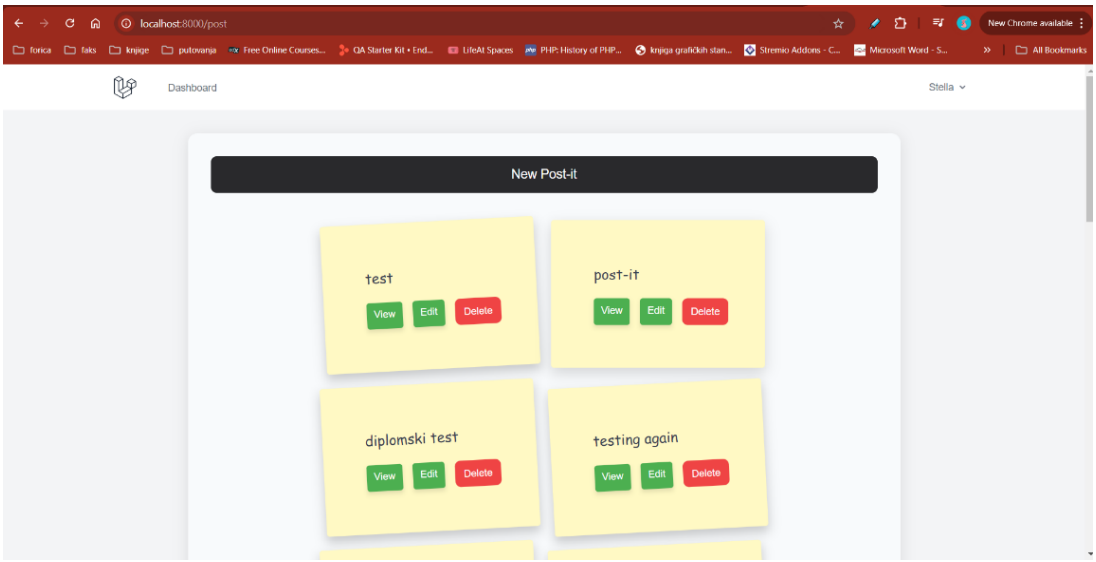
Prikaz cjelovite aplikacije sa funkcionalnostima registracije (Slika 21.), logina (Slika 22.) i CRUD-a (Slika 23.) može se pronaći na sljedećem linku: https://github.com/sfleis/diplomski-laravel



Slika 21. Prikaz registracije



Slika 22. Prikaz logina



Slika 23. Prikaz aplikacije u pregledniku

6.2 Symfony

6.2.1 Postavljanje projekta

Prije početka izrade Symfony CRUD 'Library' aplikacije, potrebno je ispuniti sljedeće zahtjeve:

- Najnoviju verziju PHP-a (verzija 8.3.10)
- Symfony CLI (verzija 5.10.2)
- XAMPP
- Najnoviju verziju Composer-a (verzija 2.7.8)
- IDE alat (VS Code) i ekstenzije

Budući da je detaljno objašnjeno postavljanje Laravel projekta, isto vrijedi i za Symfony projekt. Najnovija verzija Symfony CLI je skinuta sa službene stranice. Važno je napomenuti da Symfony CLI olakšava upravljanjem projektom, ali razvojnim okvirom upravlja Composer. Prilikom kreiranja novog Symfony projekta naredbom **'symfony symfony_projekt'** generiraju se svi potrebni paketi i ovisnosti. (Slika 20.)

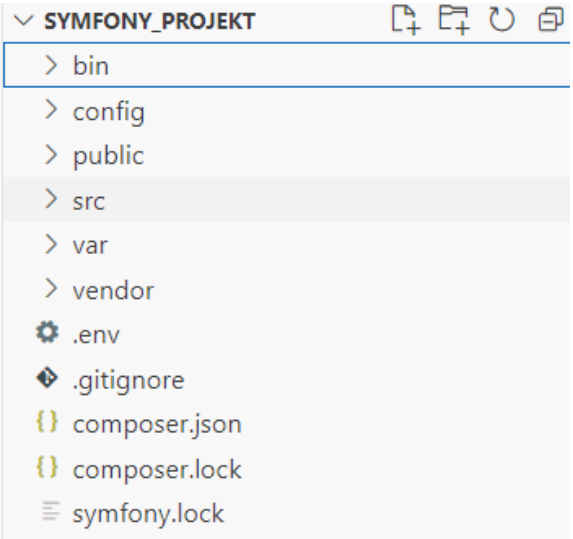
```
C:\Users\Stella>symfony new symfony_projekt
* Creating a new Symfony project with Composer
* Setting up the project under Git version control
  (running git init C:\Users\Stella\symfony_projekt)

[OK] Your project is now ready in C:\Users\Stella\symfony_projekt
```

Slika 20. Kreiranje Symfony projekta

Kako bi pokrenuli poslužitelj, potrebno je upisati putanju do napravljenog projekta: cd C:\Users\Stella\symfony_projekt te upisati sljedeću naredbu: **' symfony server:start '**. Nakon čega dobivamo poruku da je poslužitelj aktivan i da mu se može pristupiti putem localhosta. Otvaranjem linka prikazuje se 'Welcome to Symfony 7' stranica.

Otvaranjem projekta u VS Codu može se vidjeti osnovna struktura projekta samo sa komponentama potrebnima za pokretanja projekta u pregledniku. (Slika 24.)



Slika 24. Struktura direktorija Symfony projekta

Osnovni direktorij 'bin' uključuje datoteku 'console' koja predstavlja glavnu ulaznu točku za naredbe iz konzole i skripte za pokretanje aplikacije. Direktorij 'config' sadrži zadane konfiguracijske datoteke. Direktorij 'public' sadrži datotetku 'index.php', a predstavlja ulaz za dinamičke HTTP resurse. Najvažniji direktorij je 'src', unutar njega se pohranjuje aplikacijski kod. Datoteka .env omogućuje pohranu varijabli okruženja za vašu aplikaciju, dok vendor sadrži sve ovisnosti preuzete putem Composer-a. Direktorij sadržava sve log-ove i cache. [92]

6.2.2 Kreiranje ruta i kontrolera

Prije početka stvaranja kontrolera u Symfoniju prvo je potrebno definirati konfiguracijske formate. U Symfoniju postoje sljedeće opcije formata: YAML, XML, anotacije i PHP. Svaki od njih ima svoje prednosti i nedostatke, YAML se bira zbog svoje jednostavnosti i čitljivosti, ali ga ne podržavaju svi IDE alati, XML je široko korišten, ali je opširan, dok je PHP snažan i fleksibilan, ali najkompleksniji od izbora. Za ovaj projekt koristiti će se anotacije.

Za kreiranje kontrolera također će se koristit CLI. No prije nego što stvorimo kontroler potrebno je dati sljedeću naredbu: **'composer require doctrine maker'** što omogućuje stvaranje 'doctrine' i 'maker' ovisnosti. Na ovaj način se postavlja Doctrine ORM za upravljanje bazom podataka i maker paketa za generiranje različitih komponenti aplikacije.

[92] Ghorecha, Yogendra. Symfony Framework: (Directory structure). 2023.URL: https://www.linkedin.com/pulse/symfony-framework-directory-structure-yogi-ghorecha (2024-08-08)

Naredbom: **'symfony console make:controller BooksController'** stvara se BookController datoteka unutar src/Controller direktorija, ali i istoimena klasa. (Slika 25.)

```
PS C:\Users\Stella\symfony_projekt> symfony console make:controller BooksController
 created: src/Controller/BooksController.php
```

Slika 25. Izrada kontroler

Klasa BooksController nasljeđuje metode i funkcionalnosti iz 'AbstractControllera' koje olakšavaju našem postavljenom kontroleru da pomoću postavljenih metoda prikazuje pogled, vraća odgovor klijentu i preusmjerava na točnu putanju. Potrebne rute i metode kreirati će se unutar 'BookController' klase. **Pa tako ruta: #[Route('/books', name: 'books')] definira putanju koja govori Symfonyju koja metoda u kontroleru treba biti pozvana kada korisnik posjeti određeni URL** (Slika 26.). Kada korisnik pošalje GET zahtjev na /books, aktivira se metoda index(). Ova metoda ima zadatak dohvatiti sve knjige iz baze podataka i prikazati ih putem odgovarajućeg Twig predloška. Ruta nosi naziv 'books', što olakšava njezino korištenje i referenciranje u drugim dijelovima aplikacije. Kako bi olakšali dinamično rukovanje s rutama tj. da bi se omogućilo dohvaćanje i obrada podataka na temelju vrijednosti unutar URL-a koriste se parametri. Tzv. 'parametar rute' imaju definirane parametre tj. varijable unutar URL strukture. Za njihovo definiranje, varijabla se stavlja unutar vitičastih zagrada odmah nakon putanje.

```
25
26      #[Route('/books', methods: ['GET'], name: 'books')]
27      public function index(): Response
28      {
29          $books = $this->bookRepository->findAll();
30
31          return $this->render('books/index.html.twig', [
32              'books' => $books,
33          ]);
34      }
35
```

Slika 26. Prikaz definicije ruta i metode Kontrolera

6.2.3 Kreiranje entiteta i migracije

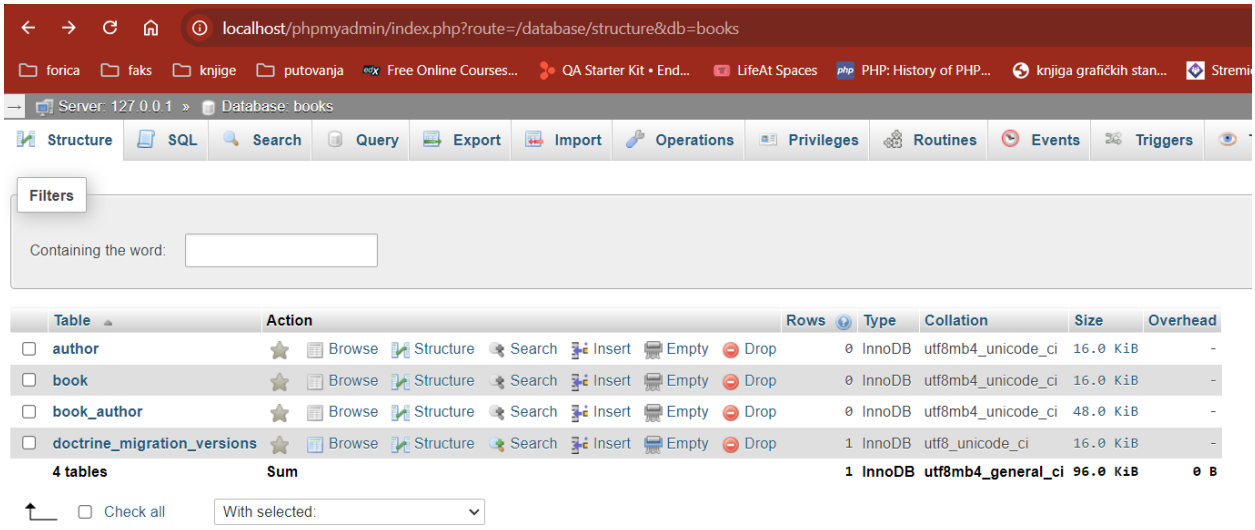
Za upravljanje bazama podataka u Symfonyju se koristi Doctrine ORM. On omogućuje rad s bazom podataka u skladu s PHP objektima, gdje jedna klasa predstavlja tablicu, a atribut klase jedan stupac u toj tablici. Uz sve ostale funkcionalnost, omogućuje izbjegavanje pisanja SQL upita za CRUD operacije. Prije instalacije Doctrine-a potrebno je skinuti potrebne pakete za sam ORM i za generiranje entiteta i migracija. U . env datoteci je potrebno postaviti informacije za DATABASE_URL kao što su MySQL korisničko ime, adresa i port na kojem se nalazi te naziv baze podataka koju želimo stvoriti kako bi se ORM mogao povezati. Naredbom **'symfony console doctrine:database:create'** kreira se baza koju smo definirali u .env datoteci.

Specifičnost Symfonyja je što nema definiran sloj modela kao neki drugi okviri. Za razliku od tradicionalnih MVC uzoraka gdje modeli upravljaju i podacima i bazom podataka, Symfony ih odvaja u Entitete koji definiraju strukturu podataka, dok repozitoriji upravljaju operacijama s bazom podataka. Repozitoriji se automatski generiraju kada se kreira entitet i nude ugrađene metode za česte operacije s bazom podataka. Ova struktura odvaja predstavljanje podataka (entitete) od pristupa podacima (repozitoriji).[93]

Izrada entiteta poziva se sljedećom naredbom: **'symfony console make:entity'** koja zatim generira klasu s anotacijama koju Doctrine koristi za mapiranje PHP objekata na tablice baze podataka te istoimeni odgovarajući repozitorij, u ovom slučaju **'BookRepository'** koji je povezana s entitetom i koristi se za interakciju s bazom podataka.. Pri definiranju svojstava potrebno je odrediti naziv svojstva, tip podataka, duljinu i može li biti null vrijednost. Doctrine se zatim brine o mapiranju na temelju tih postavki. Isto je napravljeno i za autora te njihovu ManyToMany vezu.

Migracija se vrši naredbom **'symfony console make:migration'** kreira se nova migracijska datoteka koja bilježi promjene u bazi podataka. Ova datoteka pohranjuje se u mapu migrations i ime joj uključuje datum i vrijeme. Primjena pak, ove migracije vrši se pomoću naredbe: **'symfony console doctrine:migration:migrate'**. Provjerom u phpmyadmin-u može se vidjeti da je stvaranje tablica i njihova migracija bila uspješna (Slika 27).

[93] Databases and the Doctrine ORM. Symfony. URL: https://symfony.com/doc/current/doctrine.html (2024-08-09)



Slika 27. Prikaz stvorenih eniteta

6.2.4 Kreiranje pogleda s Twig-om

Symfony se koristi sa Twig sustavom za predloške. Omogućuje pisanje sažetih i čitljivih predložaka. Neke od njegovih funkcionalnosti i prednosti uključuju: 'if', 'for' i 'else' naredbe za rukovanje s logikom, zatim filtri za izlazne varijable poput 'upper' koji pretvara tekst u velika slova, niz funkcija za rukovanje s različitim zadacima. Pruža i sigurnosne značajke kao što je automatsko escape-anje kako bi se spriječili Cross-Site-Scripting napadi. Bitno je napomenuti da Twig koristi 3 sintaktičke konstrukcije na koje treba obratiti pažnju:

- {{ ... }}: za prikazivanje sadržaja varijable ili rezultata izraza
- {% ... %}: za izvođenje logike, poput uvjeta i petlji
- {# ... #}: za dodavanje komentara u predložak [94]

Twig se nije automatski skinuo kod postavljanja projekta zbog čega je potrebno napraviti sljedeću naredbu u CLI-ju: **'composer require twig'**. Izvršenjem naredbe stvara se direktorij 'templates' unutar kojeg se nalazi 'base.html.twig' datoteka unutar koje će se spremati Pogledi. Pogledi u Symfonyju se prikazuju na temelju metode unutar kontrolera, a kako bi se prikazali, koristi se

[94] Creating and Using Templates. Symfony. URL: https://symfony.com/doc/current/templates.html (2024-08-10)

metode **'render()'** koja automatski zna da treba pretražiti 'templates' direktorij pa nije potrebno upisivati cijelu putanju. (Slika 28.)

```php
class BooksController extends AbstractController
{
    #[Route('/books', name: 'app_books')]
    4 references | 0 overrides
    public function index(): Response
    {
        return $this->render('index.html.twig');
    }
}
```

Slika 28. Render metode za prikaz predloška iz kontrolera

Kako bi uštedjeli vrijeme i spriječili nepotrebno ponavljanje koda kao što je definiranje stilova ili označivati zaglavlje i podnožje za svaku datoteku, treba se koristiti datoteka za raspored sadržaja tj. layout. U **'base.html.twig'** datoteci treba se definirati raspored, a u **'index.html.twig'** datoteci sav promjenjiv sadržaj. Element :{% block {NAME} %} {% endblock &} je mjesto gdje 'child' predložak može imati proširenje sadržaja. Kako bi naslijedio sadržaj iz base datoteke, potrebno je definirati extends sintaksu s uputom što želimo naslijediti, u ovom slučaju to je datoteka 'base.html.twig'. Time govorimo da želimo kopirati cijelu stranicu base.html.twig u pozadini i urediti određene blokove. Ovako se definiraju samo specifični 'blokovi' koje želimo promijeniti. Element: {% block body %} {% endblock %} označuje mjesto gdje se prikazuje sav sadržaj koji želimo prikazati na stranici.

Za potrebe ove aplikacije, treba stvoriti direktorij 'books' unutar direktorija templates unutar kojeg stavljamo sve potrebne stranice: indeks.html.twig za prikaz svih knjiga, show.html.twig za prikaz detalja jedne knjige, create.html.twig za dodavanje nove knjige i edit.html.twig za uređivanje postojeće (Slika 29.). Kako bismo mogli raditi s tablicom books, potrebno je povezati se s BookRepository-jem što se postiže kreiranjem privatne varijable unutar Kontrolera.

Kako bi se olakšao proces unosa i uređivanja knjiga, koristiti će se Symfony FormType. Form type je klasa koja određuje strukturu forme i definira polja koja sadrži. Kada korisnik pošalje podatke putem forme, form type automatski mapira unesene vrijednosti na odgovarajuća svojstva entiteta..

Za stvaranje formytype koriste se sljedeće naredbe: **composer require symfony/form** i **symfony console make:form BookFormType Book**.

```
{% extends "./base.html.twig" %}

{% block body %}
<div class="w-4/5 m-auto pt-40">
    <h1 class="text-6xl text-center pb-12">
        Add Book
    </h1>

    {{ form_start(form) }}

        {{ form_widget(form) }}
        <button
            type="submit"
            class="uppercase mt-15 bg-blue-500 text-gray-100 text-lg font-extrabold py-4 px-8 rounded-3xl">
            Submit Post
        </button>

    {{ form_end(form) }}
</div>
{% endblock %}
```

Slika 29. Prikaz Create pogleda

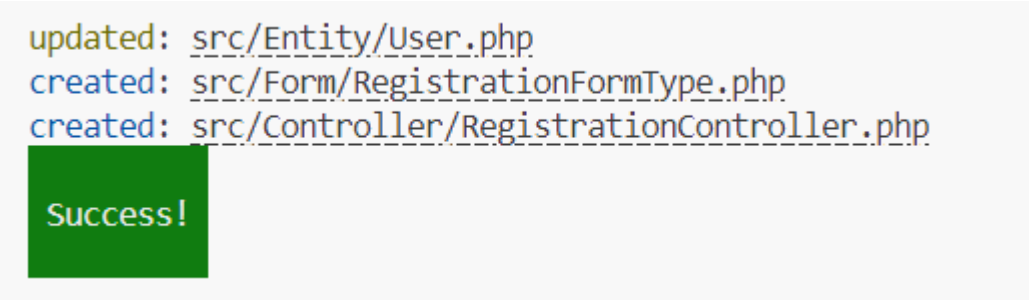
6.2.5 Registracija i login

Symfony automatski ima konfigurirane sigurnosne alate kao sesije i zaštitu od CSRF napada, ali za značajke autentifikacije i autorizacije potrebno je skinuti 'SecurityBundle'. Pokretanjem naredbe **'composer require symfony/security-bundle'** pokreće se instalacija paketa. Nakon instalacije, potrebno je pregledati postavke unutar 'config/packages/security.yaml' datoteke i po potrebi ih promijeniti.

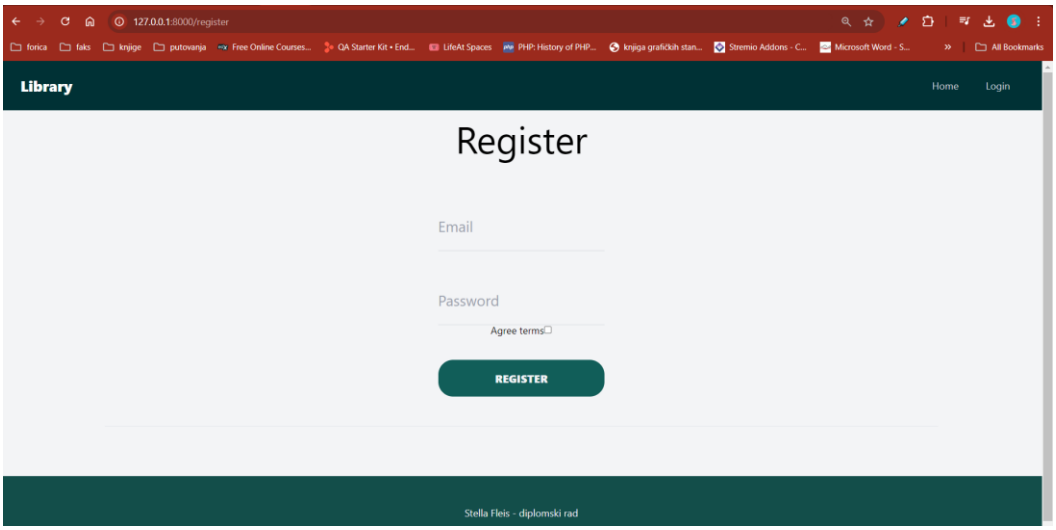
Zatim je potrebno koristiti naredbu **'symfony console make:user User'** kako bi stvorili Entitet korisnik koji se kasnije može koristiti za autentifikaciju u aplikaciji. Nakon pokretanja, potrebno je odgovoriti na sljedeća pitanja: želimo li spremiti korisnika u bazu podataka, prema čemu će se jedinstveno identificirati korisnik (prema email-u) i želimo li hashirati lozinku. Nakon potvrdnog odgovora na sva pitanja, implementacija je gotova, a Symfony je generirao Entitet User i User Repozitorij.

Kako bi se napravila migracija postavljaju se naredbe **'symfony console make:migration'** i **'symfony console doctrine:migration:migrate'** kojima se ažurira baza podataka prema novonastalim Entitetima, a ako provjerimo u phpmyadminu možemo vidjeti da je napravljena tablica User i da ima id, email, rolu i password stupce.

Symfony omogućuje da stvorimo predložak za registraciju sa naredbom: **'symfony console make:registration-form'** kojm se automatski generiraju: obrazac za registraciju, registracijski kontroler i predložak za prikaz obrasca. (Slika 30.) Obrasci omogućuju korisnicima da se registriraju na pogledu, dok kontroler obrađuje prijave i autentifikaciju. (Slika 31.)



Slika 30. Generirane datoteke za registraciju korisnika



Slika 31. Prikaz registracije u pregledniku

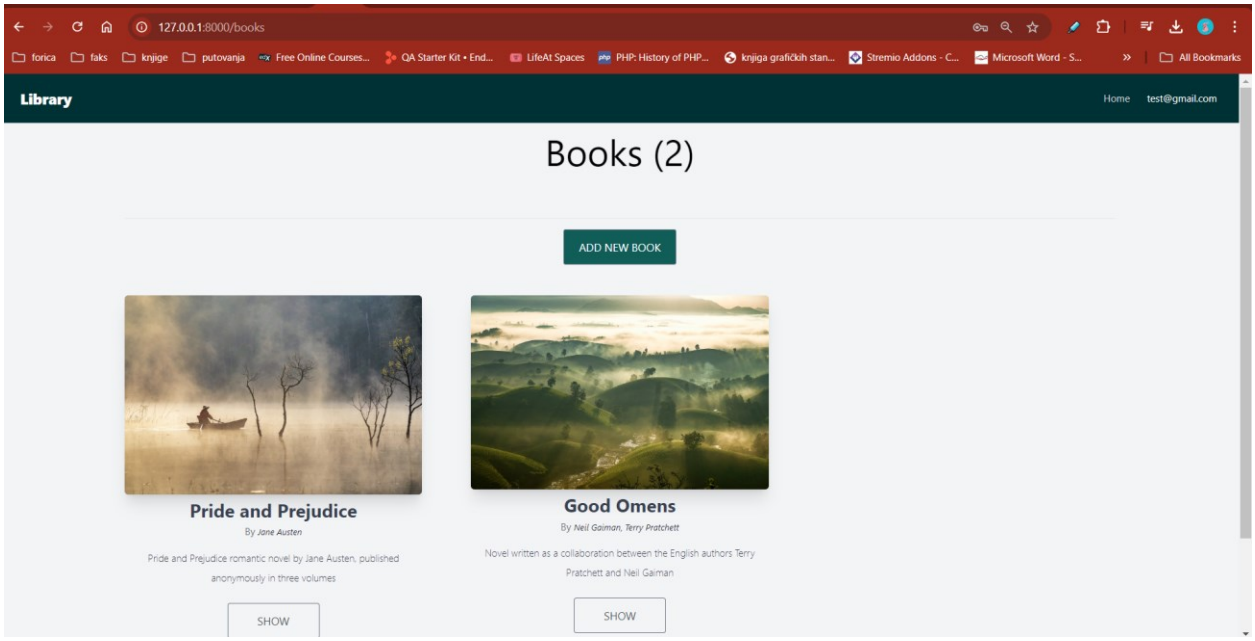
Baš kao i za registraciju, postoji i CLI naredba s kojom će se generirati potrebne datoteke za autentifikaciju korisnika. Naredba je: **'symfony console make:auth'** tijekom čijeg postavljanja je potrebno definirati naziv klase autentifikatora, naziv kontrolera, odabira opcije za odjavu i 'remember me' funkcionalnosti. Nakon odabira opcija generiraju se potrebne datoteke za autentifikaciju, a potrebna konfiguracija se ažurira. (Slika 32.) Za kraj treba urediti metodu **onAuthenticationSuccess** unutar src/Security direktorija kako bi se definirala ruta nakon uspješne prijave. (Slika 33.)

```
 [1] Always activate remember me
> 0

created: src/Security/AppCustomAuthenticator.php
updated: config/packages/security.yaml
created: src/Controller/SecurityController.php
created: templates/security/login.html.twig

Success!
```

Slika 32. Generirane datoteke za autentifikaciju korisnika



Slika 33. Prikaz aplikacije u pregledniku nakon logina

Prikaz cjelovite aplikacije sa funkcionalnostima registracije, logina i CRUD-a može se pronaći na sljedećem linku: https://github.com/sfleis/diplomski-symfony

6.3 Rasprava

Laravel i Symfony predstavljaju neke od najpopularnijih i najtraženijih razvojnih okvira, jer nude bogat izbor značajki i funkcionalnosti. Postavljanjem projekta i izradom nekih od ključnih dijelova aplikacije omogućen je detaljan prikaz usporedbe na praktičnim primjerima. Tako je pri samom postavljanju projekta za oba razvojna okvira korišten CLI koji uvelike olakšava proces instalacije. Nakon instalacije, prednost u samoj strukturi projekta preuzima Laravel koji ima puno jasnije definirane direktorije predodređene za kontrolere, modele, rute i sl, Osim toga, za razliku od Symfony-ja, dolazi sa već unaprijed instaliranim paketima koji se najčešće koriste što ga čini uvelike bržim, jednostavnijim i pristupačnijim izborom, pogotovo za početnike. Pri izradi ruta i kontrolera, Laravel koristi čitljiviju i jednostavniju sintaksu prilagođeniju za bržu izradu projekta, dok Symfony nudi nešto kompleksnija rješenja, ali namijenjena za složenije aplikacije s opsežnije definiranim rutama i kontrolerima. Pri postavljanju i kreiranju modela i migracija, Laravel se još jednom ispostavio kao jednostavnije rješenje, jer je Symfony zahtijevao postavljanje dodatnih konfiguracija, koristeći puno više koraka nego što je Laravel zahtijevao za isti rezultat. Za izradu Pogleda i Laravel i Symfony imaju odlične sustave za predloške, koje je jednostavno za koristiti, lako umeću PHP kod te omogućuju nasljeđivanje drugih datoteka. Iako je i ovdje Laravel-ov Blade nešto jednostavniji za upotrebu, treba istaknuti da Twig – Symfony-jev sustav za predloške stavlja puno veću naglasak na sigurnosti, što ga čini boljom opcijom. Naposlijetku, prema kriteriju: usporedba implementacije paketa treće strane, Laravel i Symfony nude uvelike slična rješenja. U oba razvojna okvira instalacija se vrši pomoću Composer-a, ali Symfony zahtijeva puno više naredbi, čak i ručnu konfiguraciju datoteka, za uspješno postavljanje funkcionalnosti. Ovakva kompleksnost mu pak omogućuje veću fleksibilnost, ali i kontrolu programera nad konfiguracijom.

Sličnosti ovih razvojnih okvira su brojne: oba okvira su iznimno popularna s velikom podrškom zajednice, podržavaju MVC arhitekturu, objektno-relacijsko mapiranje i dolaze s CLI alatom za olakšano izvođenje zadataka. Iako se na temelju usporednog prikaza procesa izrade CRUD aplikacija u Laravel-u i Symfony-ju može reći da su razlike na ovoj skali gotovo neprimjetne, ipak ih je potrebno istaknuti. Naime Symfony je pružio nešto teže iskustvo izrade aplikacije, ali ujedno i omogućio puno veću fleksibilnost pri konfiguraciji aplikacije, što ukazuje da Symfoy ni nije namijenjen za izradu jednostavnih aplikacija, gdje Laravel dolazi do izražaja, već je savršen za složene, velike projekte kojima navigiraju napredniji programeri.

7. Zaključak

Uloga razvojnih okvira primarno je uštediti vrijeme i resurse pri razvoju aplikacija, dostavljajući rješenja koja je lako implementirati i jednostavno za održavati. PHP razvojni okviri omogućuju programerima da izrađuju projekte koji su temeljeni na kvalitetnom kodu koji prati ustaljenje prakse i obrasce programskog jezika, izbjegavajući repetitivnost. Veliki benefit je to što razvojni okviri mogu pomoći ljudima koji su tek početnici u razvoju mrežnih aplikacija. Iako je uvijek potrebno poznavanje barem osnovnih principa programskog jezika, razvojni okviri uvelike olakšavaju izradu stabilnih aplikacija s pravilnim postavljanjem komunikacije s bazama podataka i same arhitekture aplikacije.

U moru dostupnih PHP razvojnih okvira, teško se odlučiti za jedan ispravni/najbolji/točni, upravo jer on ni ne postoji. Svaki programer, bio on profesionalac ili entuzijast, odabrati će razvojni okvir koji najbolje odgovara njegovim potrebama. Neki razvojni okviri su primjereniji za kompleksne projekte, oni su teži za naučiti, ali vrijedni jer nude ogroman raspon značajki, dok su drugi jednostavni i brzi. Bez obzira koja od ova dva smjera pojedinac odabere, potrebno je razmotriti neke od sljedećih kriterija koji su raspravljeni u teorijskom dijelu rada kako bi se donijela odluka: koliko je popularan sam okvir, je li jednostavan ili kompleksan, koliki je raspon raspoloživih biblioteka, je li temeljen na MVC arhitekturi, koje značajke podržava te kolika je podrška zajednice. S tim na umu, prikazani su pet najpopularnijih PHP razvojnih okvira gdje su jasno istaknute njihove prednosti i mane.

U praktičnom dijelu rada, na temelju navedenih ali i dodatnih kriterija, napravljena je usporedna analiza izrade CRUD aplikacije u dva najpopularnija PHP razvojna okvira – Laravel-u i Symfony-ju. Analiza je rezultirala specifičnim primjerima okvira u praksi, gdje se fokus stavio na osnovne korake pri razvoju aplikacije kao što su: postavljanje projekta, kreiranje ruta, kontrolera, modela i pogleda, korištenje migracija te implementacija sustava autentifikacije. Prilikom izrade aplikacije proizašli su mnogi izazovi iz kojih se, u stvarnom vremenu, moglo vidjeti zašto je dokumentacija i podrška zajednice toliko važan faktor prilikom odabira razvojnog okvira. Pri suočavanju s problemom bilo je potrebno samo proučiti detaljno raspisanu dokumentaciju ovih okvira ili pomoć potražiti na nekim od foruma i u vrlom kratkom roku, problem bi bio uspješno riješen.
Prilikom analize Laravel se kroz sve kriterije prikazao kao kvalitetniji odabir prema kriterijima jednostavnosti, fleksibilnosti i iskustva tijekom razvoja aplikacije. Ima jednostavniju krivulju

učenja zbog čega je odabran kao bolji izbor. Prilikom prezentacije ovog odabira važno je napomenuti ograničenja pri analizi: kao činjenica da je prednost Laravel-u dana temeljem subjektivne procjene, što upućuje na ograničenja u iskustvu autorice. Da su aplikacije jednostavne prirode što može značiti da rezultati ne mogu odražavati punu snagu svakog okvira. Da je praktična analiza ograničena na dva najpopularnija okvira, uskraćujući pregled ostalih razvojnih okvira koji potencijalno nude bolja rješenja. Daljnja istraživanja bi trebala obuhvatiti više razvojnih okvira, uključujući i nove pridošlice na tržište te istražiti njihove jednostavne i napredne funkcionalnosti.

Odabir PHP razvojnog okvira za specifičan projekt mora biti osobna odluka. U radu su prikazani najpopularniji razvojni okviri, sa definiranim setom prednosti i nedostataka. U konačnici je najbitnije procijeniti potrebe projekta, ograničenja s kojima se možemo susresti i vlastite preferencije kako bi razvojni okvir dosegao svoj puni potencijal te napravio ono što najbolje zna, olakšao razvojni proces dok održava visoki standard kvalitete.

8. Literatura

Advantages and Disadvantages of PHP. 2024. URL: https://ellow.io/advantages-and-disadvantages-of-php/

Agarwal, Yash. Advantages of PHP over Other Programming Languages. 2023. URL: https://www.scaler.com/topics/php-tutorial/advantages-of-php/

Ali, Inshal. 10 Best PHP Frameworks for Web Development in 2024. 2024. URL: www.cloudways.com/blog/best-php-frameworks/

Artisian Console. Laravel. URL: https://laravel.com/docs/11.x/artisan

Blade Templates. Laravel. URL: https://laravel.com/docs/11.x/blade#blade-directives

Brotherton, Claire. The Most Popular PHP Frameworks to Use. 2023. URL: https://kinsta.com/blog/php-frameworks/#symfony

CodeIgniter. URL: https://codeigniter.com/

Crozat, Benjamin. Is PHP Dead? Usage Statistics and Market Share for 2024. 2024. URL: https://benjamincrozat.com/php-is-dead-2024

Databases and the Doctrine ORM. Symfony. URL: https://symfony.com/doc/current/doctrine.html

Directory Structure. Laravel. URL: https://laravel.com/docs/11.x/structure

Edgar. Why Use a Router in PHP. 2020. URL: https://uhded.com/php-router

Father of PHP. URL: https://www.javatpoint.com/father-of-php

Ghorecha, Yogendra. Symfony Framework: (Directory structure). 2023. URL: https://www.linkedin.com/pulse/symfony-framework-directory-structure-yogi-ghorecha

Gupta, Ayush. Modern Web Application Architecture History and Evolution. 2024. URL: https://www.c-sharpcorner.com/blogs/modern-web-application-architecture-history-and-evolution

History of PHP: Evolution of a Web Development. 2024. URL: https://wpwebinfotech.com/blog/history-of-php/#what-is-the-future-of-php

Hopkins, Callum. PHP Master | the MVC Pattern and PHP, Part 1. 2013. URL: https://www.javatpoint.com/php-mvc-architecture

Horvat, Marko. Objektno relacijsko mapiranje. 2023. URL: edu.asoo.hr/wp-content/uploads/2024/03/G16-64-Objektno-relacijsko-mapiranje-Finalno.pdf

Hossain, Arafat Ar. Top 3 PHP Frameworks: Speed, Response Time, and Efficiency Compared. 2024. URL: https://dev.to/arafatweb/top-3-php-frameworks-speed-response-time-and-efficiency-compared-25bi#performance-comparison-table

How do ORM frameworks simplify database interactions in PHP? 2023. URL: https://tinyurl.com/5xtujmxt

How Web Works: Web Application Architecture for Beginners. 2021. URL: https://www.geeksforgeeks.org/how-web-works-web-application-architecture-for-beginners/

Introduction to Laravel and MVC Framework. 2022. URL: https://www.geeksforgeeks.org/introduction-to-laravel-and-mvc-framework/

J, Olivia. How to Use PHP Templating Engines for Web Development. 2023. URL: https://medium.com/@olivia.j.01101001/how-to-use-php-templating-engines-for-web-development-6800c9048b9c

Kuflinski, Yaroslav. PHP Frameworks Explained in 5 Simple Questions. 2019. URL: https://hackernoon.com/php-frameworks-explained-in-5-simple-questions-uvz31i7

Main Advantages & Disadvantages of Web Apps in 2024. 2024. URL: https://www.makeitsimple.co.uk/blog/web-app-advantages-disadvantages

Mbula, Abel Lifaefi. How to Build a Routing System for a PHP App from Scratch. 2023. URL: https://tinyurl.com/yscf7uyb

Middleware. Laravel. URL: https://laravel.com/docs/11.x/middleware

Nauman, Areeba. Top 5 PHP Frameworks 2024: Transform Web Development. 2024. URL: https://cyberpanel.net/blog/top-5-php-frameworks

Package Development. Laravel. URL: https://laravel.com/docs/11.x/packages

Petrenko, Viacheslav. Understanding Contemporary Web Application Architecture: Key Components, Best Practices, and Beyond. 2021. URL: https://litslink.com/blog/web-application-architecture#title1

PHP - History. 2015. URL: https://www.tutorialspoint.com/php/php_history.htm

PHP Market Share. 2021. URL: https://kinsta.com/php-market-share/#:~=PHP%20is%20the%20most%20used,due%20to%20its%20widespread%20use

PHP MVC Architecture. 2021. URL: https://www.javatpoint.com/php-mvc-architecture

PHP Unique Features. 2019. URL: https://www.geeksforgeeks.org/php-unique-features/

PHP: Hypertext Preprocessor. PHP: History of PHP - Manual. // Php.net, 2024. URL: https://www.php.net/manual/en/history.php.php

Ribeiro, Claudio. What Are the Best PHP Frameworks for 2023? 2024. URL: https://www.sitepoint.com/best-php-frameworks/#whentouseaphpframework

Setting up Laravel on Windows 10: A Comprehensive Guide. 2024. URL: https://hackmd.io/@editorial/install-laravel-windows-10

Shah, Jigar. The Most Popular PHP Framework to Use in 2023. 2024. URL: https://wpwebinfotech.com/blog/php-framework-comparison/

Sharma, Vinit. How to Use Templating in PHP Frameworks for Business Projects? 2024. URL: https://www.clariontech.com/blog/how-to-use-templating-in-php-frameworks-for-business-projects

Sparks, Matt. Building a PHP Framework: Part 8 - Routing. 2018. URL: https://dev.to/mattsparks/building-a-php-framework-part-8---routing-4jgf

Starter Kits. Laravel. URL: https://laravel.com/docs/11.x/starter-kits#laravel-breeze

Tatroe, Kevin; MacIntyre, Peter. Programming PHP: Creating Dynamic Web Pages. 4th ed. Sebastopol, CA: O'Reilly Media, 2020.

The Evolution of Modern Web Applications and Their Monitoring. 2020. URL: https://blog.uptrends.com/featured/the-evolution-of-modern-web-applications-and-their-monitoring

Top 10 Best PHP Frameworks in 2024. 2023. URL: https://www.cloudways.com/blog/best-php-frameworks/

Top Backend Frameworks for Web Development in 2022: PHP Frameworks.. 2022. URL: https://pwr-code.com/blog/top-backend-frameworks-for-web-development-in-2022-php-frameworks)

Upendraprasadmahto. Top 12 PHP Frameworks for Web Development in 2024. 2024. URL: https://medium.com/@upendraprasadmahto652/top-12-php-frameworks-for-web-development-in-2024-81a43874072a (2024-09-18)

Uryutin, Oleg. A brief history of web app. 2018. URL: https://oleg-uryutin.medium.com/a-brief-history-of-web-app-50d188f30d

Usage Statistics and Market Share of PHP for Websites. 2024. URL: https://w3techs.com/technologies/details/pl-php

Vinugayathri. Exploration of PHP Version History from PHP/FI to PHP 8.3. 2019. URL: https://www.clariontech.com/blog/exploration-of-php-version-history-from-php/fi-to-php-7.3

What Are Some Best Practices for Using Laravel's Eloquent ORM vs Symfony's Doctrine ORM? 2023. URL: www.linkedin.com/advice/0/what-some-best-practices-using-laravels-eloquent (2024-08-20)

What Are the Advantages and Disadvantages of Web Applications? 2023. URL: https://www.itrobes.com/what-are-the-advantages-and-disadvantages-of-web-applications/

What is a Web App? 2024. URL: https://www.codecademy.com/article/what-is-a-web-app

White, Emily. Why Should You Use PHP Frameworks for Web Development? 2022. URL: https://stackify.com/why-should-you-use-php-frameworks-for-web-development/