**SVEUČILIŠTE U SPLITU**

**FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE**

**SEMINARSKI RAD**

**REACT**

**Mijo Jurić-Pešić, Roko Jaman, Marko Jokić**

**Split, Ožujak 2024.**

*Tablica 1‑1 Doprinosi koautora rada*

|  |  |
| --- | --- |
| **Koautori** | **Kratki opis doprinosa u izradi rada** |
| Mijo Jurić-Pešić |  |
| Roko Jaman |  |
| Marko Jokić |  |

**IZJAVA**

Ovom izjavom potvrđujemo da samo seminarski rad s naslovom „**React**“ pod mentorstvom prof. dr. sc. Ivana Nepostojećeg pisali samostalno, primijenivši znanja i vještine stečene tijekom studiranja na Fakultetu elektrotehnike, strojarstva i brodogradnje, kao i metodologiju znanstveno-istraživačkog rada te uz korištenje literature koja je navedena u radu. Spoznaje, stavove, zaključke, teorije i zakonitosti drugih autora koje smo izravno ili parafrazirajući naveli u seminarsko radu citirali smo i povezali s korištenim bibliografskim jedinicama.

Studenti

Mijo Jurić-Pešić

Roko Jaman

Marko Jokić

Sadržaj

[1. Uvod 5](#_Toc162450037)

[2. Što je React? 6](#_Toc162450038)

[2.1 Početak Reacta kao JavaScript biblioteke 6](#_Toc162450039)

[2.2 Jordan Walke- um iza Reacta 6](#_Toc162450040)

[3 Uvod u React 7](#_Toc162450041)

[3.1 Dinamičko ažuriranje korisničkog sučelja 7](#_Toc162450042)

[3.2 Upravljanje stanjem (State Management) 7](#_Toc162450043)

[3.3 Komponentni pristup (Component-based Approach) 7](#_Toc162450044)

[4 Što React čini boljim? 9](#_Toc162450045)

[4.1 Usporedba sa Next JS-om 9](#_Toc162450046)

[4.2 Usporedba sa Angularom 9](#_Toc162450047)

[5 Glavne značajke Reacta 10](#_Toc162450048)

[5.1 Alati 10](#_Toc162450049)

[5.2 Varijable 11](#_Toc162450050)

[5.3 Flux i Redux 11](#_Toc162450051)

[6 Literatura 12](#_Toc162450052)

# Uvod

Glavni cilj ovog seminarskog rada jest prikazati što je te kako funkcionira React. Prvo poglavlje govori o tome što je React, njegov početak i razvoj u JavaScript biblioteku koju danas poznajemo. U drugom poglavlju ćemo dati uvod u React te koje probleme rješaje. Treće poglavlje ćemo iskoristiti da pokažemo što React čini boljim od ostalih frameworka usporevdivši ga sa Next JS-om te Angularom, a u četvrtom poglavlju ćemo obraditi glavne značajke Reacta.

# Što je React?

## Početak Reacta kao JavaScript biblioteke

React se prvi puta pojavio 2013. godine te od tada postaje alat za razvijanje modernih web apliakcija. Kao glavni stvaratelj Reacta ističe se Jordan Walke koji je vodio Facebook tim koji je radio na razvoju istog. Sama potreba za razvijanjem Reacta je bila ta što su ljudi iz Facebooka uvidjeli potrebu za razvijanjem alata koji bi im olakšao razvoj čitavog korisničkog sučelja. Od tada, React se koristi za jednu jednostavnu stvar, a to je da se koristi prilikom prikaza podataka u korisničkom sučelju.

## Jordan Walke- um iza Reacta

Od 2011, Jordan Walke je bio softverski inženjer u Facebooku. Facebook je napustio 2021. godine kad je odlučio osnovati svoju tvrtku. Osim Reacta, Walke se može pohvaliti izumom ReasonML-a, programskog jezika sa temeljem na Ocamil-u sa naglaskom na jednstavnost i efikasnost, čitljivost i praktičnost koji se koristi kao alternativa JavaScriptu. Kao i React, ReasonML se koristi i danas u Facebooku za razvoj web aplikacija jer se kompajlira u JavaScriptu što znači da se može koristiti sa već postojećim JavaScrpit kôdom.

# Uvod u React

React je osmislen da rješi probleme koje su donosile nove i složene web aplikacije. Utjecaj razvoja samog web-a se može najviše osjetiti na samim programerima; kako je rasla složenost istih, tako je opadala i produktivnost programera. S druge strane, korisnici su također bili ti koji su na drugoj strani osjetili značajan pad preformansi na svjoj strani zbog povećane složenosti web aplikacija. Zato se React razvio kao rješenje na specifičan set problema 1. Ramatrat ćemo tri glavna rješenja koja React nudi, a to su: dinamičko ažuriranje korisničkog sučelja (Dynamic UI Updates), upravljanje stanjem (State Managment) i komponentni pristup (Component-based Approach).

## Dinamičko ažuriranje korisničkog sučelja

React koristi Virtual DOM (Document Object Model) te ponovno učitaje bilo što kada se stanje promjeni 2. React ne radi potpuno ažuriranje, već ažurira podatke pojedine komponente. Kada se podatci promjene, kreira se novi virtualni prikaz, na temelju kojeg uz minimalne promjene se kreira novi DOM. Samim time se smanjuje vrijeme ažuriranja DOM-a koje je jako dugo te se time postižu bolje preformanse.

## Upravljanje stanjem (State Management)

Kako web aplikacija raste, tako raste i složenost njezine strukture te na većim aplikacijama takve strukture često uzrokuju probleme ili neželjene promjene u radu same aplikacije. Koristeći React, mi ne utječemo na model aplikacije, već samo vizualni sloj same aplikacije. Također, standardne aplikacije koriste kompleksno upravljanje što dovodi do nečitkog te teško održivog koda. Kako bi rili taj problem, React uvodi Flux, jednosmjeran tok podatka koji nadopunjuje React. Pošto Flux nije framework, prolazeći mnoge implementacije istog, Redux postaje glavna state management biblioteka za developere u Reactu.

## Komponentni pristup (Component-based Approach)

Koristeći komponente iz prethodnih projekata ili već gotova softverska rješenja, uštedjet ćemo vrijeme, resurse te same developere. Iz tog razloga React koristi komponentni pristup, kojim koristimo već gotove i testirane komponente kojima povećajemo modularnost i skalabilnost same aplikacije. Također, lakše je razvijati aplikaciju „dio po dio“, nego čitavu odjednom. React to rješava tako da čitavo korisničko sučelje dijeli na manje komponente što olakšaje rad na komponentama te održavanje same aplikacije.

# Što React čini boljim?

U web-dev industriji, broj tehnologija, frameworka i alata neprestano raste. Kod biranja frameworka ili librarya za projekt, JavaScript ima mnoštvo izbora za front-end development kao što su: Vue, Ember, Backbone, jQuery, Svelte i mnogi drugi. U ovom seminaru ćemo se usredotočiti na usporedbu Reacta s dva najpopularnija frameworka: Next JS i Angular. Odredit ćemo performanse, dokumentaciju, održavanje i značajke frameworka i najbolju opciju za određene situacije.

## Usporedba sa Next JS-om

Next JS je framework za razvoj web aplikacija uz pomoć React librarya. Koristi se kao nadogradnja na React i time donosi dodatne značajke. Iako su Next JS i React više u partnerstvu nego natjecanju postoje razlike u korištenju tih tehnologija kao što su server-side rendering, static site generation, napredniji routing sustavi lakše korištenje API-ova. Odabir između Reacta i Next JS-a najviše ovisi o složenosti projekta te jesu li potrebne dodatne mogućnosti koje donosi Next JS.

## Usporedba sa Angularom

Angular je također framework za web aplikacije s glavnom razlikom da se implementira s TypeScriptom, što je superset JavaScripta. Razvio ga je Google 2016. godine na temelju njegovog prethodnika AngualJS. Angular i React se razlikuju u arhitekturi za prikaz stranice: React koristi virtualni DOM, a angular stvarni DOM i MVC pristup (Model-View-Controller). MVC znači da se aplikacija dijeli u tri dijela: modeli koji se koriste za spremanje podataka, view prikazuje podatke iz modela na ekran korisnika i controller se bavi interakcijom između modela i view-a, za razliku od Reacta koji koristi komponentni pristup.

# Glavne značajke Reacta

React donosi niz ključnih značajki koje olakšavaju razvoj modernih web aplikacija. Sa svojom arhitekturom koja se temelji na komponentama, programeri mogu stvoriti elemente korisničkog sučelja poput gumba ili traka za pretraživanje, koji se zatim mogu ponovno koristiti u cijeloj aplikaciji. Ovo ne samo da usmjerava razvojni proces, već i poboljšava mogućnost održavanja. Jedna od prepoznatljivih značajki Reacta je JSX (JavaScript XML). JSX izgleda slično HTML-u i omogućuje razvojnim programerima pisanje oznaka izravno unutar svog JavaScript koda. Ova jedinstvena mogućnost pojednostavljuje kodiranje pružajući intuitivniji način strukturiranja komponenti. Reactov utjecaj ne završava na pregledniku; također igra vitalnu ulogu u razvoju mobilnih aplikacija s varijantom koja se zove React Native. React također koristi virtualni DOM (Document Object Model), koncept programiranja gdje se idealna ili "virtualna" reprezentacija korisničkog sučelja čuva u memoriji i sinkronizira sa "pravim" DOM-om Reactovim postupkom usklađivanja. Ova značajka donosi visoku učinkovitost u prikazivanju web stranica, čineći aplikacije bržima i osjetljivijima.

Upravljanje stanjem u Reactu je fleksibilno; dok postoji ugrađena podrška kroz 'state' u komponentama klase ili 'useState' kuku u funkcionalnim komponentama, React također prihvaća biblioteke trećih strana kao što je Redux za veće aplikacije.

Konačno, osim što je izvrstan za izgradnju korisničkih sučelja na web platformi sa svojom glavnom bibliotekom react-dom, također podržava razvoj mobilne platforme koristeći react-native. To omogućuje stvaranje i web-aplikacija i izvornih mobilnih aplikacija koristeći uglavnom slične obrasce kodiranja.

## Alati

React dolazi s bogatim skupom alata koji pomažu razvojnim timovima u izgradnji performantnih aplikacija. Kao što je istaknuto, "React Developer Tools pruža duboke uvide u stanje vaše aplikacije i omogućuje vam učinkovito dijagnosticiranje performansi."3. Ovaj alat omogućuje programerima da duboko pregledaju i razumiju stanje komponenti u stvarnom vremenu, poboljšavajući tako proces razvoja.

## Varijable

Varijable u Reactu omogućuju dinamičko upravljanje podacima i stanjem aplikacije. Kroz korištenje varijabli, "React pruža jednostavan način za upravljanje stanjem komponenti."3. Ova fleksibilnost omogućuje programerima da lako manipuliraju podacima unutar komponenata, što dovodi do bržeg i efikasnijeg razvoja aplikacija.

## Flux i Redux

Flux i Redux su arhitekturni obrasci koji se često koriste s Reactom za upravljanje stanjem aplikacije. Kroz korištenje ovih biblioteka, "React aplikacije postaju skalabilnije i lakše za održavanje."4. Ovi alati omogućuju efikasno upravljanje složenim stanjima aplikacije, što rezultira boljom skalabilnošću i održivošću projekta.

Redux je biblioteka otvorenog koda objavljena nekoliko godina nakon Reacta koja koristi koncept "pohrane" podataka kako bi pomogla u praćenju stanja aplikacije. Iako u Reactu možete dijeliti podatke o stanju među komponentama, ponekad je to teško učiniti učinkovito, posebno za složene aplikacije. U Reduxu sve komponente imaju izravan pristup istim trgovinama, što olakšava prijenos stanja između različitih dijelova aplikacije.

Flux je aplikacijska arhitektura koju Facebook interno koristi za izgradnju klijentske web aplikacije s Reactom. Korisno je kada projekt ima dinamičke podatke, a moramo ažurirati podatke na učinkovit način. Smanjuje pogreške tijekom izvođenja.

# Literatura

(1) Gackenheimer, C. *Introduction to React*; Apress, 2015.

(2) Chen, S.; Thaduri, U. R.; Ballamudi, V. K. R. Front-End Development in React: An Overview. *Eng. Int.* **2019**, *7* (2), 117–126. https://doi.org/10.18034/ei.v7i2.662.

(3) *React*. https://react.dev/ (accessed 2024-03-27).

(4) *GitHub - facebookarchive/flux: Application Architecture for Building User Interfaces*. https://github.com/facebookarchive/flux (accessed 2024-03-27).