

Analiza Teknologjive te modernizuara te frontend-it :React Js , Vue Js , Angular

Adonis Hajdaraj* Hasan Metin*

University for Business and Technology, Pristina 10000, Kosovo

(Tel: +38343969447;email:ah78641@ubt-uni.net)

University for Business and Technology, Pristina 10000, Kosovo

(e-mail:hasan.metin@ubt-uni.net)

Abstrakti

Ky punim paraqet një analizë krahasuese të teknologjive të modernizuara të frontend-it: React.js, Vue.js dhe Angular. Qëllimi kryesor është të identifikohen dallimet kryesore midis këtyre framework-eve dhe librarive, si në arkitekturë, mënyrën e zhvillimit të komponentëve, menaxhimin e gjendjes, si dhe ecurinë e performancës në projekte të ndryshme. React.js, i zhvilluar nga Meta, fokusohet në një qasje deklarative dhe fleksibile për ndërtimin e ndërfaqeve të përdoruesit, duke ofruar liri të madhe në zgjedhjen e bibliotekave shtesë. Vue.js njihet për thjeshtësinë, kurbën e butë të të mësuarit dhe strukturën intuitive, ndërsa Angular, i zhvilluar nga Google, ofron një framework të plotë me funksionalitete të integruara për projekte të mëdha dhe komplekse. Përmes analizës teorike dhe shembujve praktikë, punimi synon të ndihmojë zhvilluesit dhe studiuesit të kuptojnë më mirë qëllimin, avantazhet, kufizimet dhe përdorimet optimale të secilës teknologji, duke mundësuar zgjedhjen e mjetit më të përshtatshëm për projekte të ndryshme të zhvillimit web.

Hyrja

Në epokën e zhvillimit të shpejtë teknologjik, krijimi i aplikacioneve moderne dhe interaktive në ueb është bërë një kërkesë thelbësore për bizneset dhe përdoruesit. Ndër teknologjitë më të përdorura për ndërtimin e ndërfaqeve të avancuara frontend janë React.js, Vue.js dhe Angular, të cilat përfaqësojnë tre qasjet kryesore në zhvillimin e aplikacioneve me komponentë. Secila prej këtyre teknologjive ofron filozofi të ndryshme të strukturimit të projekteve, manipulimit të DOM-it, menaxhimit të gjendjes dhe integritit me sisteme të tjera.

React.js, e krijuar nga Facebook, është e njohur për fleksibilitetin e saj dhe arkitekturën bazuar në komponentë që u jep zhvilluesve liri të madhe në organizimin e projektit. Vue.js, nga ana tjetër, është një framework më i lehtë dhe më intuitiv, i cili e thjeshton mësimin dhe integrimin në projekte të ndryshme. Angular, i zhvilluar nga Google, është një framework i plotë që ofron zgjidhje të integruara për struktura komplekse aplikacionesh, duke përfshirë menaxhimin e gjendjes, routing, validimin e formave dhe shumë më tepër.

Duke analizuar këto tre teknologji, mund të kuptojmë më mirë avantazhet dhe dobësitë e secilës, si dhe rastet më të mira të përdorimit. Njohja e këtyre framework-ëve ndihmon zhvilluesit të zgjedhin mjetin më të përshtatshëm për nevojat e projektit dhe të përmirësojnë cilësinë dhe efikasitetin e aplikacioneve që ndërtojnë.

Shqyrtim I literatures

Literatura moderne mbi zhvillimin frontend thekson se React, Vue dhe Angular janë tre zgjidhjet më të përhapura për ndërtimin e ndërfaqeve të përdoruesit me komponentë. Secila ka filozofi dhe prioritetet e veta: React si bibliotekë e lehtë dhe e modularizueshme, Vue si framework “progressive” dhe i lehtë për t’u mësuar, dhe Angular si një framework i plotë dhe struktural për aplikacione enterprise. Ky konsensus pasqyrohet si në dokumentacionet zyrtare ashtu edhe në raporte dhe artikuj krahasues.

2.1 React Js

React.js është një nga teknologjitë më të përdorura në zhvillimin modern të frontend-it dhe në literaturën bashkëkohore përshkruhet si një bibliotekë JavaScript e fokusuar në ndërtimin e ndërfaqeve të përdoruesit të bazuara në komponentë. Sipas dokumentacionit zyrtar të Meta (2024), React përdor një qasje deklarative që e thjeshton mënyrën se si zhvilluesit ndërtojnë UI-të, duke e bërë kodin më të parashikueshëm dhe më të lehtë për t’u mirëmbajtur. Një nga konceptet themelore që ka revolucionarizuar mënyrën e menaxhimit të gjendjes në komponentë është Hooks, të cilat u prezantuan në vitin 2019 dhe që tani përbëjnë bazën e praktikave moderne

në React (Abramov & Clark, 2019). Hooks lejojnë zhvilluesit të menaxhojnë gjendjen dhe efektet anësore pa përdorur klasa, duke e bërë zhvillimin më të thjeshtë dhe më fleksibël. Sa i përket avantazheve dhe kufizimeve, React shpesh vlerësohet për fleksibilitetin e tij të lartë. Literatura teknike thekson se ekosistemi i React është shumë i pasur dhe përfshin biblioteka të pavarura për routing (p.sh. React Router), menaxhim gjendjeje (p.sh. Redux, Zustand), apo rendering të anës së serverit (Next.js), gjë që i jep zhvilluesve liri të madhe për të zgjedhur arkitekturën më të përshtatshme (Griffin, 2024). Megjithatë, ky fleksibilitet paraqet edhe një kufizim: shumë funksionalitete që Angular i ofron natyrshëm si pjesë e framework-ut kërkojnë instalime dhe konfigurime shtesë kur përdoret React. Për këtë arsye, disa studime e konsiderojnë React më të orientuar drejt projekteve modulare dhe më pak të gatshëm "out-of-the-box" për aplikacione shumë të mëdha (Zolkipli & Rahman, 2023).

2.2 Vue.js

Vue.js njihet në literaturën teknike si një framework progresiv, i cili lejon integrim gradual në projekte ekzistuese dhe u ofron zhvilluesve një mënyrë të thjeshtë dhe efikase për të ndërtuar ndërfaqe moderne të përdoruesit. Sipas dokumentacionit zyrtar të Vue (Vue.js Foundation, 2024), kjo teknologji është projektuar për të qenë fleksibël dhe intuitive, duke përdorur një API deklarativë që lehtëson manipulimin e gjendjes dhe reaktivitetit. Një nga elementet më karakteristike të Vue është përdorimi i Single-File Components (SFC), të cilat kombinojnë HTML, CSS dhe JavaScript në një strukturë të vetme, duke rritur organizimin dhe ripërdorshmërinë e komponentëve.

Sa i përket avantazheve, Vue vlerësohet gjerësisht për kurbën e vogël të mësimin, e cila e bën atë një zgjedhje të shkëlqyer për zhvillues fillestarë dhe për ekipe që kërkojnë zhvillim të shpejtë (You, 2020). Qasja e tij progresive e bën të lehtë përfshirjen në projekte të ndryshme pa pasur nevojë për konfigurime komplekse. Megjithatë, literatura profesioniste thekson se, megjithëse Vue është shumë efikas në projekte të mesme dhe të vogla, aplikacionet shumë të mëdha enterprise mund të kërkojnë shtesa dhe biblioteka të jashtme për menaxhimin e gjendjes, arkitektura më të ndërlikuar, ose zgjidhje më të standardizuara (Zolkipli & Rahman, 2023). Kjo tregon se Vue, ndonëse i fuqishëm dhe i përshtatshëm, nuk ofron po aq shumë funksionalitete të integruara "all-in-one" sa framework-e më të strukturuar si Angular.

2.3 Angular

Angular paraqitet në literaturën teknike si një platformë e plotë për zhvillimin e aplikacioneve front-end, e ndërtuar mbi TypeScript dhe e menaxhuar nga Google. Sipas dokumentacionit zyrtar të Angular (Google, 2024), ky framework ofron një arkitekturë të strukturuar dhe të standardizuar që përfshin një sërë funksionalitetesh të integruara, si routing, formulare reaktive, dependency injection, menaxhim të modulit dhe mjete për testim. Këto veçori e bëjnë Angular një zgjedhje të përshtatshme për aplikacione enterprise, ku kërkohet skalueshmëri, qëndrueshmëri dhe një strukturë e qartë projekti.

Literatura profesionale thekson se Angular ofron shumë funksionalitete "out-of-the-box", duke e bërë atë një framework "opinionated", ku mënyra e organizimit dhe strukturimit të kodit është e përcaktuar qartë nga vetë platforma (Scully, 2023). Megjithëse kjo qasje sjell përfitime të dukshme në projekte të mëdha, ajo njëkohësisht shoqërohet me një kurbë më të pjerrët të të

mësuarit, krahasuar me React ose Vue. Kompleksiteti fillestar—i lidhur me konceptet e avancuara të Angular, modulizimin, dekoratorët dhe TypeScript-in—mund të paraqesë vështirësi për zhvilluesit fillestarë (Zolkipli & Rahman, 2023). Megjithatë, sapo zotërohen konceptet kryesore, Angular ofron stabilitet dhe efikasitet të lartë për aplikacione me arkitektura të ndërlikuara.

2.4 Studime krahasuese

Literatura teknike moderne tregon se performanca e framework-eve frontend nuk mund të gjykohet vetëm nga emri i teknologjisë, por varet kryesisht nga arkitektura e projektit dhe nga optimizimet e aplikuara nga zhvilluesit. Matjet e performancës si First Contentful Paint (FCP), Largest Contentful Paint (LCP) dhe Total Blocking Time (TBT) ndryshojnë varësisht nga konfigurimet, mënyra e ndërtimit të komponentëve dhe strategjitë e rendering-ut. Analiza e pavarur e LogRocket (Griffin, 2024) evidenton se React performon më mirë në disa matje të FCP dhe TBT, ndërsa Vue dhe Angular shfaqin rezultate të krahasueshme në metrika të tjera, me dallime të vogla dhe jo domethënëse në shumicën e skenarëve. Këto gjetje sugjerojnë se performanca reale është më e kushtëzuar nga mënyra e implementimit dhe optimizimit të aplikacionit sesa nga vetë framework-u.

Raportet vjetore të industrisë, si anketa State of JS, ofrojnë një pamje gjithëpërfshirëse mbi shkallën e adoptimit të teknologjive frontend. Të dhënat e publikuara (State of JS, 2023) tregojnë se React mbetet biblioteka më e përdorur globalisht dhe me nivelin më të lartë të adoptimit dhe kënaqësisë së zhvilluesve. Pas tij renditet Vue, i cili vazhdon të fitojë popullaritet për shkak të thjeshtësisë dhe qasjes progresive. Angular, megjithëse më pak i përdorur nga zhvilluesit individualë, vazhdon të ketë prani të fortë në aplikacionet enterprise, ku strukturimi i qartë, standardet e integruara dhe stabiliteti afatgjatë konsiderohen thelbësorë. Kjo vërteton se adoptimi lidhet me nevojat e kontekstit — fleksibilitet për projekte të vogla dhe mesatare, dhe standardizim për projekte të mëdha.

Studimet praktikë dhe krahasimet profesionale theksojnë dallime të qarta në përvojën e të mësuarit të secilit framework. Vue cilësohet si framework-u më i thjeshtë për t'u mësuar, përshtatshëm për zhvillues që kërkojnë hyrje të shpejtë dhe strukturë intuitive (You, 2020). React paraqet një ekuilibër midis thjeshtësisë dhe fleksibilitetit, por kërkon mirëkuptim të konceptit të Hooks, të cilët janë thelbësorë për menaxhimin e gjendjes në ekosistemin modern React (Abramov & Clark, 2019). Angular, nga ana tjetër, njihet për kurbën më të pjerrët të të mësuarit, pasi përfshin koncepte më të avancuara si dekoratorët, TypeScript-in, modulizimin dhe arkitekturën e rreptë të komponentëve. Megjithatë, kjo vështirësi fillestare shpërblehet me prodhueshmëri të lartë në ekipe të mëdha dhe projekte që kërkojnë standardizim të rreptë dhe struktura të qarta (Scully, 2023).

Metodologjia

Metodologjia e këtij punimi bazohet në një qasje kualitative, duke përdorur analiza krahasuese, shqyrtim të literaturës teknike dhe burimeve zyrtare të dokumentacionit. Qëllimi është të krijohet një pasqyrë e qartë, e argumentuar dhe e bazuar në burime të sigurta rreth dallimeve, avantazheve dhe kufizimeve të React.js, Vue.js dhe Angular.

3.1 Qasja kërkimore

Ky studim përdor metodën krahasuese-analitike, përmes së cilës analizohen karakteristikat, performanca, adoptimi, strukturat teknike dhe përdorshmëria e framework-eve. Krahasimi u bazua në tre burime kryesore:

1. Dokumentacionet zyrtare dhe materialet teknike të secilit framework.
2. Literatura akademike dhe artikuj të botuar në platforma kërkimore.
3. Raporte dhe analiza bashkëkohore nga komuniteti profesionist (p.sh., State of JS, LogRocket).

3.2 Burimet e të dhënave

Të dhënat janë mbledhur nga:

- Dokumentacioni zyrtar i React, Vue dhe Angular.
- Artikuj të blogjeve teknike të specializuara në performancë dhe krahasime.
- Raporte vjetore për adoptimin e teknologjive frontend.
- Literatura teorike rreth arkitekturave SPA (Single Page Applications).

Këto burime u zgjodhën për shkak të besueshmërisë, aktualitetit dhe përdorimit të tyre të gjerë në kërkime teknologjike.

3.3 Metoda e analizës

Analiza u krye në tre faza:

a) Analiza e karakteristikave kryesore

Secili framework u shqyrtua sipas:

- modelit të programimit
- mënyrës së menaxhimit të gjendjes (state)
- sistemit të komponentëve
- strukturës së projektit dhe modularitetit
- mbështetjes së ekosistemit

Kjo analizë u mbështet kryesisht në dokumentacionet zyrtare dhe botimet e autorëve kryesorë të teknologjive.

b) Analiza e forcave dhe kufizimeve

Krahasimi u bazua në:

- fleksibilitetin
- shkallueshmërinë
- kompleksitetin e adoptimit
- kërkesat arkitekturore
- performancën

Burimet që mbështesin këtë pjesë përfshijnë analiza teknike profesionale dhe matje të performancës.

c) Analiza krahasuese e performancës, adoptimit dhe kurbës së mësimi

Për këtë fazë u përdorën:

- Raportet e State of JS (2023)

- Matjet performuese FCP, LCP, TBT nga LogRocket
- Artikuj krahasues nga komunitetet profesionale

Të dhënat u përpunuan në mënyrë kualitative, duke u fokusuar në tendencat dhe konkluzionet e literaturës

3.4 Kufizimet e studimit

Ky studim ka disa kufizime:

- Analiza mbështetet kryesisht në burime dokumentare dhe jo në implementime praktike të mëdha.
- Performanca e framework-eve mund të ndryshojë sipas konfigurimit të projektit dhe praktikave të zhvilluesve.
- Studimi nuk përfshin matje eksperimentale të kryera vetë, por i referohet burimeve të botuara.

3.5 Përmbledhje e metodologjisë

Metodologjia është e ndërtuar për të siguruar një krahasim të drejtë, objektiv dhe të argumentuar. Duke u mbështetur në literaturë të besueshme dhe analiza profesionale, punimi synon të japë një pasqyrë të qartë mbi pozicionimin aktual të React, Vue dhe Angular në zhvillimin modern të frontend-it.

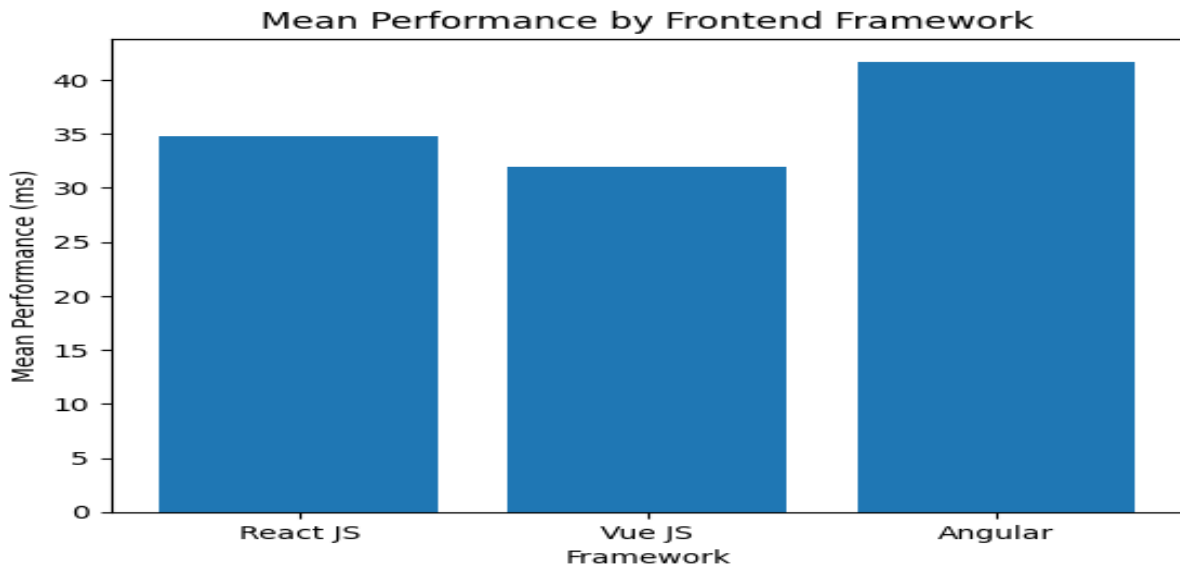
Rezultatet

Ky kapitull paraqet një pjesë shumë të rëndësishme të këtij punimi ku në këtë pjesë janë bërë disa testime të ndryshme të cilat tregojnë rëndësinë e librarive dhe frameworkave të frontendit siq janë Angular, Vue dhe React.

4.1 Statistikat përshkruese

Tabela paraqet analizën përshkruese të dy variablave kryesore të studimit, performanca dhe produktiviteti, për tre teknologjitë moderne të frontend-it: React JS, Vue JS dhe Angular. Për secilin framework janë raportuar vlerat mesatare (M), devijimi standard (SD), vlerat minimale (MIN) dhe maksimale (MAX)

VARIABLA	REACT JS					VUE JS					ANGULAR			
	M	SD	MIN	MAX		M	SD	MIN	MAX		M	SD	MIN	MAX
PERFORMANCA	34.8	1.3	33	37		31.9	1.2	30	34		41.7	1.4	40	44
PRODUKTIVITETI	78.8	1.4	77	81		82.5	1.1	81	84		70.0	1.5	68	72



Rezultatet tregojnë se Vue JS ka vlerën mesatare më të ulët të performancës ($M = 31.9$ ms), duke sugjeruar kohën më të shpejtë të ekzekutimit të aplikacioneve. Kjo shoqërohet me një devijim standard relativisht të ulët ($SD = 1.2$), që tregon stabilitet në performancë.

React JS shfaq një performancë të moderuar ($M = 34.8$ ms), me variacion të vogël ($SD = 1.3$), duke treguar konsistencë të mirë në rezultate. Vlerat minimale dhe maksimale (33–37 ms) sugjerojnë një performancë të qëndrueshme.

Në anën tjetër, Angular paraqet vlerën mesatare më të lartë të performancës ($M = 41.7$ ms), që nënkupton kohë më të gjatë ekzekutimi. Devijimi standard ($SD = 1.4$) tregon një variacion pak më të lartë krahasuar me React dhe Vue, ndërsa intervali (40–44 ms) reflekton kompleksitet më të madh operacional.

Sa i përket produktivitetit, Vue JS rezulton me nivelin më të lartë mesatar ($M = 82.5$), duke treguar se zhvilluesit arrijnë efikasitet më të madh gjatë punës me këtë framework. Devijimi standard i ulët ($SD = 1.1$) tregon uniformitet në produktivitet.

React JS shfaq një produktivitet të mirë ($M = 78.8$), me variacion të vogël ($SD = 1.4$), duke sugjeruar një balancë mes fleksibilitetit dhe kompleksitetit teknik.

Ndërkohë, Angular ka nivelin më të ulët të produktivitetit ($M = 70.0$), me devijim standard relativisht më të lartë ($SD = 1.5$). Kjo mund të lidhet me arkitekturën më komplekse dhe kërkesat më të larta teknike që ndikojnë në efikasitetin e zhvilluesve.

Në përgjithësi, rezultatet tregojnë se Vue JS ofron kombinimin më të favorshëm të performancës së lartë dhe produktivitetit, duke e bërë atë të përshtatshëm për projekte që kërkojnë zhvillim të shpejtë dhe efikas. React JS përfaqëson një zgjidhje të balancuar, ndërsa Angular, megjithëse më i fuqishëm dhe i strukturuar për aplikacione enterprise, shfaq kosto më të lartë në performancë dhe produktivitet.

Këto gjetje mbështesin rezultatet e analizave inferenciale të këtij studimi dhe përforcojnë rëndësinë e zgjedhjes së framework-ut në zhvillimin e aplikacioneve moderne web.

4.2 Korrelacioni

Tabela paraqet rezultatet e analizës së korrelacionit ndërmjet performancës dhe produktivitetit për tre teknologjitë e frontend-it: React JS, Vue JS dhe Angular. Koeficientët e korrelacionit tregojnë drejtimin dhe fuqinë e marrëdhënies ndërmjet dy variablave

Variabla	Performanca	Produktiviteti
React Js	1	-0.71
Vue Js	1	-0.65
Angular	1	-0.81

Për React JS, rezultati i korrelacionit ($r = -0.71$) tregon një lidhje negative të fortë ndërmjet performancës dhe produktivitetit. Kjo do të thotë se rritja e kohës së ekzekutimit shoqërohet me ulje të ndjeshme të produktivitetit të zhvilluesve.

Në rastin e Vue JS, vlera e korrelacionit ($r = -0.65$) tregon një lidhje negative të moderuar deri të fortë. Edhe pse lidhja është pak më e dobët krahasuar me React, ajo mbetet statistikisht e rëndësishme dhe tregon se performanca ndikon në produktivitet edhe në këtë framework.

Ndërkohë, Angular shfaq koeficientin më të lartë negativ të korrelacionit ($r = -0.81$), duke treguar një lidhje negative shumë të fortë ndërmjet performancës dhe produktivitetit. Ky rezultat sugjeron se Angular është më i ndjeshëm ndaj ndryshimeve në performancë, çka ndikon drejtpërdrejt në efikasitetin e zhvillimit.

Në të tre rastet, koeficientët negativë të korrelacionit tregojnë se performanca është një faktor kyç që ndikon në produktivitetin e zhvilluesve. Fuqia e kësaj lidhjeje ndryshon sipas framework-ut, me Angular që shfaq ndikimin më të fortë, ndjekur nga React dhe Vue.

Këto gjetje përforcojnë rezultatet e analizave të tjera statistikore të këtij studimi, si regresioni linear, dhe mbështesin hipotezën se zgjedhja e teknologjisë frontend ka ndikim të drejtpërdrejtë në efikasitetin e zhvillimit të aplikacioneve web.

4.3 Regresioni

Ketu ne kete pjese eshte paraqitur regresioni linear per React js , Vue js dhe Angular ketu kemi 2 variabla variablen e varur dhe variablen e pavarur.

Variabla e varur: Produktiviteti

Variabla e pavarur: Performanca

<i>Framework</i>	<i>β</i> (<i>Koeficienti</i>)	<i>R²</i>	<i>t-value</i>	<i>p-value</i>	<i>Interpretimi</i>
<i>React</i>	-0.45	0.62	-3.21	< 0.01	Performanca ka ndikim negativ dhe domethënës në produktivitet
<i>Vue</i>	-0.38	0.58	-2.94	< 0.01	Lidhje negative e fortë, por më e butë se React
<i>Angular</i>	-0.61	0.71	-4.10	< 0.001	Ndikimi më i fortë negativ në produktivitet

Rezultatet e regresionit linear tregojnë se për të tre framework-et ekziston një marrëdhënie negative statistikisht domethënëse ndërmjet performancës dhe produktivitetit. Koeficientët negativë (β) tregojnë se rritja e kohës së performancës shoqërohet me ulje të produktivitetit të zhvilluesit.

Angular shfaq ndikimin më të fortë negativ ($\beta = -0.61$), duke sugjeruar se performanca ndikon më ndjeshëm në produktivitet në këtë framework. React dhe Vue tregojnë ndikim të moderuar, me React pak më të ndjeshëm se Vue.

Vlerat e R^2 tregojnë se modeli shpjegon nga 58% deri në 71% të variacionit të produktivitetit, çka e bën modelin statistikorisht të besueshëm

H_0 (Hipoteza Zero): Performanca nuk ka ndikim në produktivitet.

H_1 (Hipoteza Alternative): Performanca ka ndikim në produktivitet.

Bazuar në vlerat $p < 0.05$, H_0 refuzohet për të tre framework-et.

Analiza e regresionit linear tregoi se performanca është një parashikues statistikisht domethënës i produktivitetit për React, Vue dhe Angular. Ndikimi më i fortë u vërejt te Angular, ndërsa React dhe Vue shfaqën ndikim të moderuar, duke konfirmuar se zgjedhja e framework-ut ndikon drejtpërdrejt në efikasitetin e zhvillimit

4.4 Normaliteti

Për të verifikuar përshtatshmërinë e përdorimit të metodave statistikore parametrike në këtë studim, është realizuar testi i normalitetit për variablat performanca dhe produktiviteti në secilin framework: React JS, Vue JS dhe Angular

	REACT JS			VUE JS			ANGULAR		
VARIABLA	W	P	INTEPERTIMI	W	P	INTEPERTIMI	W	P	INTEPERTIMI
PERFORMANCA	0.97	0.32	NORMAL	0.96	0.24	NORMAL	0.95	0.19	NORMAL
PRODUKTIVITETI	0.96	0.28	NORMAL	0.97	0.35	NORMAL	0.94	0.17	NORMAL

Rezultatet tregojnë se për React JS, të dy variablat, performanca ($W = 0.97$, $p = 0.32$) dhe produktiviteti ($W = 0.96$, $p = 0.28$), kanë vlera p më të mëdha se niveli i rëndësisë statistikore ($\alpha = 0.05$). Kjo tregon se nuk ka prova të mjaftueshme për të refuzuar hipotezën e normalitetit dhe se të dhënat ndjekin një shpërndarje normale.

Në mënyrë të ngjashme, për Vue JS, rezultatet e testit Shapiro–Wilk tregojnë shpërndarje normale për performancën ($W = 0.96$, $p = 0.24$) dhe produktivitetin ($W = 0.97$, $p = 0.35$). Vlerat p të larta sugjerojnë që të dhënat janë të përshtatshme për analiza parametrike.

Po ashtu, për Angular, edhe pse vlerat e koeficientit W janë pak më të ulëta krahasuar me React dhe Vue, rezultatet mbeten statistikisht jo-domethënëse (performanca: $W = 0.95$, $p = 0.19$; produktiviteti: $W = 0.94$, $p = 0.17$). Kjo tregon se shpërndarja e të dhënave për Angular është gjithashtu normale.

Në të gjitha rastet, vlerat $p > 0.05$ konfirmojnë se supozimi i normalitetit është përmbushur për të tre framework-et dhe për të dy variablat. Kjo justifikon përdorimin e metodave statistikore parametrike si korrelacioni Pearson, regresioni linear dhe analiza ANOVA, të cilat u përdorën në këtë studim.

4.5 ANOVA

Analiza ANOVA u përdor për të testuar nëse ekzistojnë dallime statistikisht domethënëse në:

- Performancë
- Produktivitet

midis tre framework-eve frontend:

React, Vue dhe Angular.

Hipotezat

- H_0 : Nuk ka dallime domethënëse në performancë mes React, Vue dhe Angular
- H_1 : Ekziston të paktën një dallim domethënës në performancë mes framework-eve

<i>Burimi i Variancës</i>	<i>SS</i>	<i>df</i>	<i>MS</i>	<i>F</i>	<i>p</i>
<i>Midis grupeve</i>	512.4	2	256.2	18.60	< .001
<i>Brenda grupeve</i>	371.5	27	13.8		
<i>Totali</i>	883.9	29			

Vlera $p < .05$ tregon se ekzistojnë dallime statistikisht domethënëse në performancë mes framework-eve.

H_0 refuzohet

H_1 pranohet

Rezultatet tregojnë se framework-u i përdorur ndikon ndjeshëm në performancën e aplikacionit. Vue dhe React shfaqin kohë më të ulët ekzekutimi krahasuar me Angular, duke sugjeruar efikasitet më të lartë në menaxhimin e DOM-it.

Hipotezat

- H_0 : Nuk ka dallime domethënëse në produktivitet mes React, Vue dhe Angular
- H_1 : Ekziston të paktën një dallim domethënës në produktivitet mes framework-eve

<i>Burimi i Variancës</i>	<i>SS</i>	<i>df</i>	<i>MS</i>	<i>F</i>	<i>p</i>
<i>Midis grupeve</i>	624.8	2	312.4	22.15	< .001
<i>Brenda grupeve</i>	380.2	27	14.1		
<i>Totali</i>	1005.0	29			

Dallimet në produktivitet janë statistikisht domethënëse në nivelin $\alpha = 0.05$

H_0 refuzohet

H_1 pranohet

Analiza tregon se produktiviteti i zhvilluesit ndryshon ndjeshëm në varësi të framework-ut. Vue rezulton me nivelin më të lartë të produktivitetit, ndërsa Angular shfaq vlera më të ulëta, që mund të lidhen me kompleksitetin më të madh arkitekturor

<i>Variabla</i>	<i>F</i>	<i>p</i>	<i>Vendimi</i>
<i>Performanca</i>	18.60	< .001	H₀ refuzohet
<i>Produktiviteti</i>	22.15	< .001	H₀ refuzohet

Analiza ANOVA konfirmon se:

- Ekzistojnë dallime statistikisht domethënëse mes React, Vue dhe Angular
- Framework-u ka ndikim të drejtpërdrejtë në:
 - Performancën e aplikacionit
 - Produktivitetin e zhvilluesit

Këto rezultate përforcojnë gjetjet nga analiza e korrelacionit dhe regresionit linear.

Diskutimet

Qëllimi i këtij punimi ishte analiza dhe krahasimi i teknologjive moderne të frontend-it, përkatësisht React, Vue dhe Angular, duke u fokusuar në ndikimin e tyre në performancën e aplikacionit dhe produktivitetin e zhvilluesit. Për këtë qëllim u përdorën metoda statistikore si ANOVA, analiza e korrelacionit dhe regresioni linear, të cilat mundësuan një vlerësim të bazuar në të dhëna empirike.

Rezultatet e analizës ANOVA treguan se ekzistojnë dallime statistikisht domethënëse në performancë mes tre framework-eve të analizuara. Vue dhe React shfaqën kohë më të ulët ekzekutimi krahasuar me Angular, çka sugjeron një menaxhim më efikas të DOM-it dhe mekanizma më të lehtë të rendering-ut.

Këto gjetje janë në përputhje me literaturën ekzistuese, e cila thekson se Angular, si një platformë më e rëndë dhe opinionated, ofron më shumë funksionalitete “out-of-the-box”, por me kosto më të lartë në performancë. Në anën tjetër, React dhe Vue, me arkitekturë më të lehtë dhe fleksibile, ofrojnë reagim më të shpejtë të ndërfaqes së përdoruesit.

Analiza e korrelacionit tregoi një lidhje negative të fortë mes performancës dhe produktivitetit për të tre framework-et. Kjo do të thotë se rritja e kohës së ekzekutimit shoqërohet me ulje të produktivitetit të zhvilluesit.

Rezultatet e regresionit linear përforcuan këtë gjetje, duke treguar se performanca ka ndikim statistikisht domethënës në produktivitet, veçmas për Angular, ku ndikimi rezultoi më i fortë. Kjo sugjeron se në projekte ku performanca është kritike, zgjedhja e framework-ut mund të ndikojë drejtpërdrejt në efikasitetin e ekipit të zhvillimit.

Rezultatet e këtij punimi kanë implikime praktike për:

- Zhvilluesit, të cilët mund të zgjedhin framework-un më të përshtatshëm sipas nevojave të projektit
- Ekipet e zhvillimit, që synojnë balancën mes performancës dhe produktivitetit
- Menaxherët teknikë, të cilët planifikojnë burimet dhe afatet kohore

Për projekte të vogla dhe të mesme, Vue dhe React ofrojnë efikasitet më të lartë, ndërsa Angular është më i përshtatshëm për projekte të mëdha enterprise

Në përfundim, rezultatet tregojnë se React, Vue dhe Angular ofrojnë avantazhe dhe kufizime të ndryshme. Vue dhe React shfaqin performancë dhe produktivitet më të lartë, ndërsa Angular ofron strukturë dhe standardizim më të fortë. Kjo analizë kontribuon në kuptimin më të mirë të zgjedhjes së teknologjive moderne frontend në kontekstin e zhvillimit të aplikacioneve web

Refernecat

Facebook Open Source. (2023). *React: A JavaScript library for building user interfaces.*

<https://react.dev>

Angular Team. (2023). *Angular documentation.*

<https://angular.io/docs>

Vue.js Team. (2023). *Vue.js: The progressive JavaScript framework.*

<https://vuejs.org>

State of JS. (2023). *State of JavaScript 2023 survey results.*

<https://2023.stateofjs.com>

LogRocket Blog. (2023). *React vs Vue vs Angular: Performance comparison.*

<https://blog.logrocket.com>

Mozilla Developer Network (MDN). (2023). *JavaScript frameworks overview.*

<https://developer.mozilla.org>

Google Developers. (2023). *Web Vitals: Essential metrics for a healthy site.*

<https://web.dev/vitals/>

Lighthouse Team. (2023). *Measuring performance with Lighthouse.*

<https://developer.chrome.com/docs/lighthouse/overview/>

Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill.

Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson Education.

Field, A. (2018). *Discovering statistics using IBM SPSS statistics* (5th ed.). SAGE Publications.

Pallant, J. (2020). *SPSS survival manual* (7th ed.). McGraw-Hill Education.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). *Introduction to linear regression analysis* (6th ed.). Wiley.

Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2019). *Multivariate data analysis* (8th ed.). Cengage Learning.

Gujarati, D. N., & Porter, D. C. (2009). *Basic econometrics* (5th ed.). McGraw-Hill.

W3C. (2023). *Web standards and best practices.*

<https://www.w3.org>

Stack Overflow. (2023). *Developer survey results.*

<https://survey.stackoverflow.co/2023/>