UDC: 004.4:004.7 Info M: str. 43-49

PROGRESIVNE WEB APLIKACIJE: IZMEĐU NATIVNIH I MOBILNIH WEB APLIKACIJA
PROGRESSIVE WEB APPS: BETWEEN NATIVE AND MOBILE WEB APPLICATIONS

Slavimir Vesić

REZIME: U današnjem svetu mobilni uređaji su postali neizostavni deo naše realnosti. Prema nekom nepisanom pravilu većina ljudi se oslanja na mobilne tehnologije kada su u pitanju kontakti, poruke, email poruke, obaveštenja, kalendari, socijalne mreže i td., koje omogućavaju gotovo trenutnu razmenu informacija. Usled sve većeg broja mobilnih uređaja i veće pokrivenosti mobilnih mreža, potražnja korisnika za mobilnim sadržajem je uvećana. Iako postoji veliki broj mobilnih aplikacija za određenu mobilnu platformu tzv. nativnih aplikacija, prema nekim istraživanjima je uočeno da korisnici iako preuzimaju mobilne aplikacije sa online prodavnica, besplatno ili za određenu nadoknadu, instaliraju ih i konfigurišu, zapravo koriste veoma mali broj istih. Sa druge strane, Web ne zahteva instaliranje, konfigurisanje i dodeljivanje prava aplikacijama, već jednostavnim unosom adrese u pretraživač sadržaj postaje dostupan i upravo zbog brzine kojom korisnik dolazi do novog sadržaja njegovo zadovoljstvo je veće. Google je uočio opisni trendi za tržište mobilnih uređaja napravio je novi tip aplikacija, progresivne Web aplikacije. Pomenute aplikacije kombinuju karakteristike nativnih aplikacija i mobilnih Web aplikacija i time dobijaju najbolje od oba sveta čime nude bolje korisničko iskustvo u odnosu na dosadašnji mobilni Web, što kao posledicu ima veće provedeno vreme na sajtu od strane korisnika.Iako je pomenuti trend još u fazi razvoja, rezultati velikih on-line kompanija koje su implementirale PWA je navelo autora da opiše ovu vrstu aplikacija, zajedno sa promenama u korisničkom iskustvu.

KLJUČNE REČI: progresivne Web aplikacije, nativne aplikacije, mobilne Web aplikacije, korisničko iskustvo

ABSTRACT: In today's world, mobile devices have become an indispensable part of our reality. According to an unwritten rule, most people rely on mobile technology when it comes to contacts, messages, email messages, notifications, calendars, social networks, etc., which allow them almost instant exchange of information. With the growing number of mobile devices and the increasing coverage of mobile networks, user demands for mobile content is increased.Although there is a growing number of mobile applications for specific mobile platform,so-called native applications, some researches has found that even though users download mobile apps from online store, free or for a fee, install and configure them, they actually use a very small number of them.On the other hand, the Web does not require installation, configuration, or setting permissions for applications, but simply by entering the address into the browser, the content becomes available. The speed of content delivery to the end user is the reason for customer satisfaction. Google has noticed the aforementioned trend and created a new type of application called Progressive Web apps. It combines characteristics of native applications, as well as mobile web applications, in order to have best of both worlds, and offer better user experience than the current mobile Web, which increases the time spent on the site by users.Although the above-mentioned trend is still under development, the results of large on-line companies that implemented Progressive Web Apps has led authorto describe this type of application, along with the changes in the user experience.

KEY WORDS: Progressive Web Apps, native applications, mobile Web applications, user experience

1. UVOD

Još od 1983. godine i pojave prvog mobilnog telefona Motorola DynaTAC 8000X, u žargonu poznatijeg kao „cigla", uporedo sa proizvodnjom mobilnih uređaja razvijaju se i aplikacije za te uređaje. Poučeni iskustvom iz industrije personalnih računara, gde se u početku smatralo da je hardware važniji i manje pažnje se posvećivalo software-u, što se kasnije pokazalo kao netačno i da ukoliko ne postoji operativni sistem i aplikativni softver, mašine su neupotrebljive, proizvođači mobilnih telefona razvijaju aplikacije uporedo sa kreiranjem novih modela mobilnih telefona. U tom periodu prve generacije mobilnih telefona, konkurencija između proizvođača je bila izražena i nije bilo razmene iskustva i inovacija, tako da je kompletan software za mobilne telefone razvijan in-house od strane proizvođača sa naglaskom na zaštiti od konkurenata. Veći deo populacije programera je radio izvan kompanija koje su proizvodile mobilnih telefona , tako da nisu bili uključeni u proces razvoja softvera za mobilne uređaje [1]. U tom periodu su dominirale video igre, od kojih su poznatije Snake, Pong i Tic-Tac-Toe, a korisnici su koristili jednostavne, a ponekad teške za upotrebu, aplikacije kao što su kalkulator, alarm i kalendar.

Kako su vremenom cene mobilnih telefona opale, sve veći broj korisnika je krenuo da ih koristi i porasla je potražnja za novim mogućnosti koje proizvođači mobilnih telefona sa svojim resursima nisu mogli da obezbede. Kao odgovor na ove izazove pojavio se Wireless Application Protocol, skraćeno WAP, kao uprošćena verzija Hyper Text Transfer Protocol-a, skraćeno HTTP-a. Proizvođači mobilnih telefona su smatrali da je to dobro rešenje, preko kojeg će se privući programere i ljude koji kreiraju sadržaj i da će na taj način ispuniti potrebe korisnika, a mobilni operateri su videli šansu da obezbede nove usluge i pri tome dobro ih naplate. Korisnici nisu bili zadovoljni previše WAP-om pre svega zbog lošeg korisničkog iskustva, jer većina WAP sajtova je isporučivala jedinstven sadržaj bez prilagođavanja karakteristikama mobilnih telefona sa jedne strane, a sa druge visokom cenom usluge. Često se čula kritika da je WAP, zapravo „Wait-and-Pay", tj. „čekaj i plati", koja odnosila na činjenicu da zbog male veličine ekrana

u kojem je korisnik čitao deo rečenice i kada je želeo da pročita drugi deo rečenice morao je da sačeka da se sadržaj učita i pri tome dodatno naplati. Iz pomenutih razloga WAP nije mogao da doživi veći komercijalni uspeh [1].

Cena hardware-a je opadala i dalje i pojavili su se Personal Digital Assistant, skraćeno PDA, uređaji koji su u sebi imali kompaktne verzije Windows i Linux operativnih sistema. U tom trenutku, sve veći broj programera Desktop aplikacija se uključio u razvoj softvera za ove uređaje. Proizvođači mobilnih telefona su shvatili da ukoliko žele da nastave sa daljom prodajom svojih proizvoda moraju da do određenog nivoa izlože mehanizme rada njihovih uređaja. Pojavljuje se veliki broj vlasničkih platformi i programeri kreiraju aplikacije za njih. Među prvima su Palm OS i RIM Blackberry OS. U to vreme Sun Microsystems je ponudio Java Micro Edition, skraćeno Java ME. Pored toga Nokia, Sony Ericsson, Motorola i Samsung su razvili Symbian OS. Apple je 2007. godine razvio iPhone iOS, a samo godinu dana kasnije je nastao Google Android [1].

Kako je tržište mobilnih telefona postalo fragmentisano sa svim platformama i udelima koje one imaju na tržištu, ista pojava je zadesila i programere, gde su oni primorani da programiraju u različitim programerskim jezicima, okruženjima i alatima i razvijaju tzv. nativne aplikacije (eng. native application). Nativne aplikacije su napisane za specifičnu platformu i one mogu da iskoriste karakteristike operativnog sistema i drugog softvera koji je instaliran na toj platformi i na taj način upotrebljavaju hardver i softver tog uređaja kao što su kamera ili sistem globalnog pozicioniranja (eng. Global Positioning System), skraćeno GPS. Nativne aplikacije žive na uređaju i njima se pristupa preko ikonica koje se nalaze na tzv. home screen-u uređaja. One se instaliraju preko application store-ova, kao što su Google Play ili Apple-ov App Store. Pored toga što mogu da koriste GPS, kameru, kompas, listu kontakata i td., one takođe mogu da uključe i pokrete. Nativne aplikacije koriste obaveštenja, tj. notifikacije (eng. notification) koje predstavljaju specifičan vid poruka koje se prikazuju korisniku na ekranu u oblasti koja se naziva notification area. Pored navedenog, nativne aplikacije mogu da rade u offline modu, tj. ne pristupaju Web-u [1].

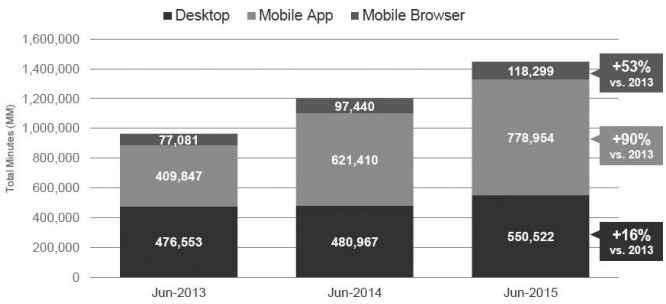
World Wide Web, skraćeno Web, se pojavio ranih 90-ih i za veoma kratko vreme je zapao za oko široj populaciji. Web je dramatično izmenio i nastavio da menja način na koji ljudi vrše interakciju. On je uzdigao Internet iz mora mreža koje su razmenjivale podatke u tom trenutku, do globalne jedinstvene mreže za razmenu podataka [2]. Za razliku od dotadašnjeg radija i televizije, gde ste zavisili od emitera programa i sadržaja koji on nudi, koji utiče na to šta i kada se neki program emituje, na Web-u kao korisnici imate mogućnost u svakom trenutku da birate sadržaj na koji želite da se pretplatite. Web je takođe poslužio kao osnova za mnoge aplikacije, koje su naglasile njegovu vrednost i za koje su korisnici bili spremni da obezbede vezu ka Internetu samo da bi ih koristili, tzv. „aplikacije ubice“ (eng. killer applications). Neke od njih su Youtube, Gmail i Facebook [2].Danas se prilikom razvoja Web aplikacija, značajno vreme posvećuje interaktivnosti i usability-u, kao i kompletnom korisničkom iskustvu koje jedna Web aplikacija donosi, što se može videti kroz razvoj SPA (eng. single-page application) aplikacija, čiji je cilj da obezbede neprekidnu interakciju između korisnika i aplikacije [3].

Mobilni Web je zapravo upotreba Web-a sa mobilnih uređaja, kroz mobilne ili druge bežične mreže. Pojavom pametnih telefona (eng. smartphone), primećena je tendencija korisnika da sve više koriste Web, preko istih. Progameri su primorani da već svoje postojeće Web sajtove i aplikacije prilagođavaju za mobile uređaje i time pruže objedinjeno korisničko iskustvo kroz Web i mobilni Web koje korisnici očekuju. Dakle, usability je veoma važan aspekt mobilnog Web-a [4].Mobilni Web se pokreće u pretraživaču kucanjem URL adrese, gde pretraživač prosleđuje zahtev serveru za određenom Web stranicom, a server kao odgovor vraća tu stranicu, najčešće u obliku HTML5 fajla i kada se taj fajl otvori iz pretraživača, korisnik pristupa pregledanju sadržaja te stranice. Mobilne Web aplikacije izvorno nemaju neke od karakteristika nativnih aplikacija, a među njima su: notifikacije, pokretanje procesa u pozadini, mogućnost obrade složenijih pokreta, kao i rad u offline modu [5].

Korisnici upotrebljavaju i nativne i mobilne Web aplikacije na svojim uređajima da bi pristupili sadržaju koji ih interesuje i oba tipa aplikacija imaju svoje jake i slabe strane. U drugom delu ćemo govoriti o istraživanju trenda upotrebe mobilnih Web i nativnih aplikacija od strane korisnika i o tome koja je tendencija kod korisnika uočena. U trećem delu ćemo govoriti kako je Google implementirao novi tip aplikacija kao odgovor na rezultate pomenutih istraživanja i šta se želi time postići. Na kraju ćemo dati zaključak.

2. ANALIZA TRENDAUPOTREBE NATIVNIH I MOBILNIH WEB APLIKACIJA

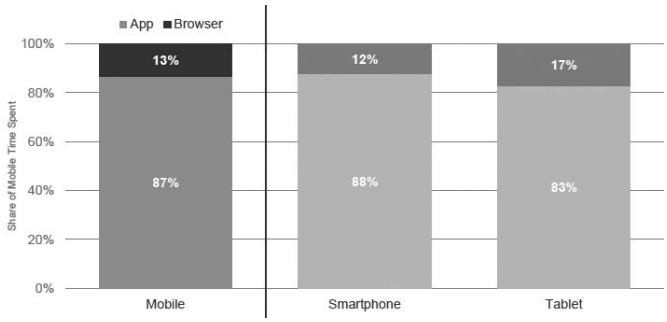
Prema izveštaju kompanije comScore [6], koja se bavi merenjem i analizom tržišta za najveće svetske kompanije, medije, marketinške agencije i izdavače, u periodu od juna 2013. godine do juna 2015. godine upotreba digitalnih medija je porasla 49% , ***slika 1***.



Slika 1. *Porast u vremenu provedenom na digitalnim medijima, preuzeto [6]*

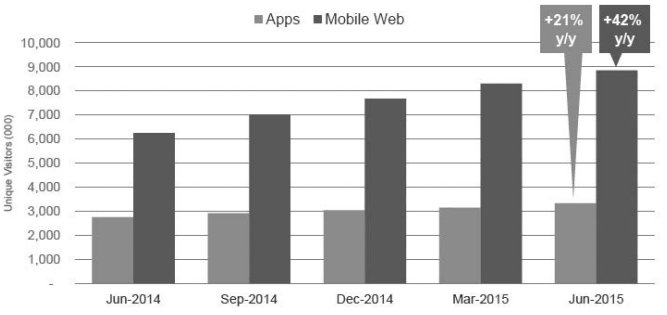
Učešće i u tom rastu najviše zauzima rast u vremenu provedenom u upotrebi nativnih aplikacije za mobilne uređaje oko 90%, pri čemu je primećen i visok rast upotrebe mobilnih Web aplikacija od 53%.Takođe je uočen i rast u vremenu od 16 %

provedenom sa Desktop računara. Pored toga, udeo vremena koji korisnici provedu na mobilnom telefonu je raspoređen na način tako da 87% vremena korisnici provedu u upotrebi nativnih aplikacija, a 13% vremena provedu u upotrebi mobilnog Web-a, kao na ***slici 2***. Na tablet uređajima u odnosu na pametne telefone, veća je upotreba mobilnog pretraživača, što se može tumačiti većom veličinom ekrana [6].



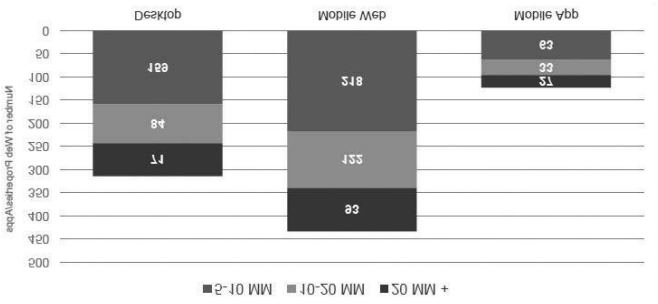
Slika 2. *Udeo vremena provedenom na mobilnim uređajima: aplikacije VS pretraživači, preuzeto [6]*

Na osnovu provedenog vremena u upotrebi, za trenutak se može izvesti zaključak da nativne aplikacije imaju dominantnu ulogu, ali to je samo deo celokupne slike, za koju je comScore dao još jednu od mnogih statistika u svojem izveštaju. Na ***slici 3***. je prikazano poređenje broja poseta korisnika nativnih aplikacija u odnosu na broj mobilnih Web aplikacija za prvih 1000 nativnih aplikacija i 1000 prvih mobilnih Web sajtova. Ova statistika prikazuje neočekivan rezultat. Ne samo da broj posetilaca mobilnih Web sajtova je veći više od 2.5 puta, već i ovaj broj posetioca raste dva puta brže u odnosu na nativne aplikacije [6].



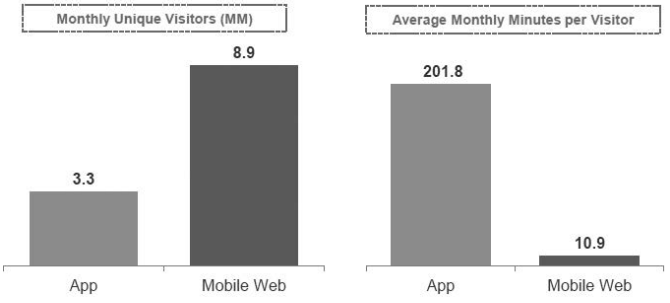
Slika 3. *Prosečni mesečni br. korisnika, poređenje 1000 nativnih i 1000 mobilnih Web aplikacija [6]*

Iako broj korisnika nativnih aplikacija raste, postojeća digitalna infrastruktura otežava prisvajanje većeg broja korisnika istih nego kod mobilnog Web-a, kao što je prikazano na ***slici 4***. Mobilni Web i dalje ima 3.5 više Web sajtova i svojstava (eng. properties) sa 5 miliona jedinstvenih posetilaca u odnosnu na nativne aplikacije.



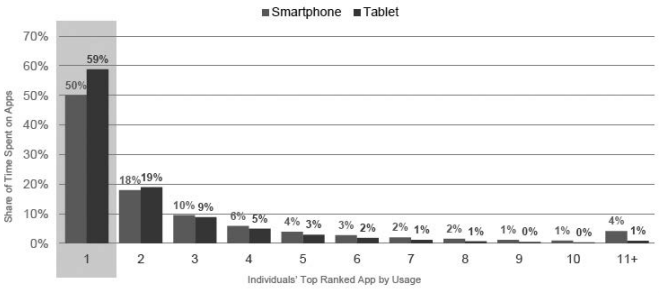
Slika 4. *Poređenje mobilnih aplikacija i Web svojstava koje su dostigle milionske korisnike [6]*

Još jedna statistika koja upoređuje 1000 nativnih aplikacija i 1000 mobilnih Web svojstava i sajtova prikazuje da je uspostavljanje korisnika nativnih aplikacija znatno teže, ali njihova prava vrednost je u tome što su lojalni i oni provode 18 puta više vremena u nativnim aplikacijama nego na mobilnom Web-u.Pomenuta statistika je prikazana na ***slici 5***., gde vidimo da korisnici nativnih aplikacija provedu na mesečnom nivou preko 3 sata, dok na mobilnim Web sajtovima provedu oko 10 minuta, dok jedinstvenih korisnika na nivou meseca izraženih u milionima ima više na mobilnim Web sajtovima.



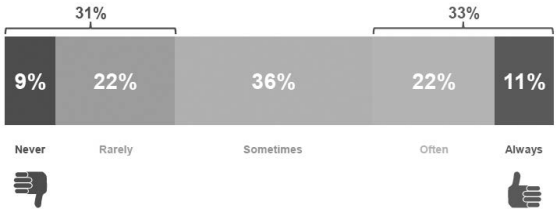
Slika 5. *Poređenje nativnih i mobilnih Web app prema br. korisnika i prosečnom vremenu provedenom [6]*

Sa aspekta korisničkih navika, preko 80% vremena korisnika nativnih aplikacija, je utrošeno na 3 najčešće korišćene aplikacije iako korisnik ima oko 25 zahteva za nativnim aplikacijama mesečno, dok je pomenuti trend još izraženiji kod aplikacija za tablet. Opisana statistika je prikazana na ***slici 6***.



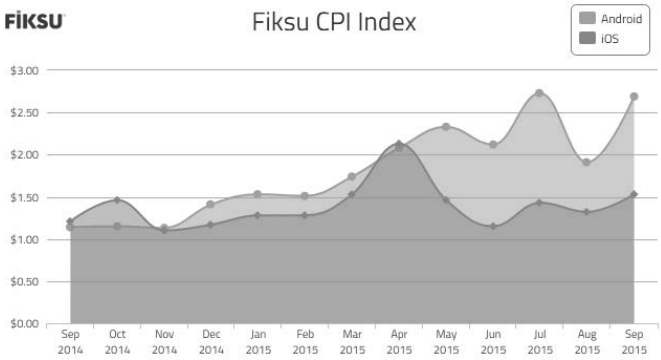
Slika 6. *Rangiranje nativnih aplikacija prema vremenu provdenom u istim, preuzeto [6]*

U pogledu prihvatanja push notifikacija (eng. push notification) na pametnim telefonima, među korisnicima vlada ravnoteža u pogledu da li ih omogućavaju ili ne, kao što je prikazano na ***slici 7***.



Slika 7. *Da li prihvatate push notifikacije od aplikacije, preuzeto [6]*

Kompanija Fiksu, koja se bavi istraživanjem tržišta mobilnih uređaja, koristi Cost per Install, skraćeno CPI, Index koji meri troškove privlačenja korisnika da instaliraju nativnu aplikaciju kao posledicu oglašavanja [7]. Tako da vidimo da troškovi privlačenja korisnika rastu i opadaju, ali su tu između 1$ do 2 $, ***slika 8***. Pored navedene, Fiksu prati i metriku koja sračunava troškove lojalnog korisnika, Cost Per Loyal User i ona u septembru 2015. iznosi oko 4$, što je za 2$ više u odnosu na godinu pre toga [7].



Slika 8. *CPI index u periodu od septembra 2014. do 2015. godine, preuzeto [7]*

3. PROGRESSIVE WEB APPS

3.1. *RAIL Performance Model*

Na osnovu prezentovanih rezultata istraživanja kompanija comScore i Fiksu, kao i internih istraživanja, kompanija Google Inc. se usmerila poslednje dve godine, na unapređivanje korisničkog iskustva mobinih Web aplikacija. To se pre svega može videti na broju konferencija[8][9][10] u poslednje 2 godine, a pogotovo u poslednjih godinu dana, na kojima se prikazuju nove mogućnosti unutar Web aplikacija. Najznačajnije među njima su Google I/O Conference, Chrome Dev Summit koji je u 2016. godini preimenovan u Progressive Web App Dev Summit, kao i O'Reilly Fluent Conference. Inženjeri Google-a su shvatili da za dobro definisanje svoje strategije moraju početi od početka, tj. od korisnika i korisničkog iskustva i zato su se usmerili na definisanje jasnih ciljeva na čijim osnovama će biti koncentrisan budući razvoj. Nielsen, vodeći ekspert za usability pitanja, je još pre više od dve decenije postavio značajna vremena odgovora jednog sistema [11], kasnije su i druga istraživanja [12] to potvdila. Vođen ovim rezultatima, Google je dodao još neke granice koje su značajne kod mobilnih aplikacija [13][14]:

- 0 – 16 milisekundi – obzirom da se ekran osvežava 60 puta u sekundi,ova granica predstavlja vreme koje potrebno da se okviri (eng. frames) prikažu. Ljudi ne vole varijabilne brzine okvira (eng. frame rate) i povremena prekidanja, jer veoma dobro prate pokrete.
- 0 – 100 milisekundi – ukoliko odgovor na akciju korisnika je unutar ovog vremenskog intervala, korisnik percipira trenutni odgovor sistema, a ukoliko je preko ove granice veza je narušena između akcije korisnika i reakcije sistema.
- 100 – 300 milisekundi – korisnici primećuju blago kašnjenje
- 300 – 1000 milisekundi – unutar ove vremenske granice korisnik percipira prirodno i neprekidno napredovanje u zadatku. Ukoliko je ovo vreme veće korisnik gubi fokus sa zadatka koji izvršva. Učitavanje Web stranice se može smatrati zadatkom.
- preko 10000 milisekundi – korisnik je nezadovoljan i veoma verovatno napušta zadatak

Ukoliko sada posmatramo jednu interakciju između korisnika mobilnog telefona i Web aplikacije, možemo primetiti da prilikom upotrebe aplikacije uvek postoje 4 aktivnosti: odgovor (eng. Response), animacija (eng. Animation), stanje mirovanja (eng. Idle) i učitavanje (eng. Load). Mapiranjem vremenskih granica sa unapred poznatim aktivnostima u interakciji između korisnika mobilnog uređaja i aplikacije na samom uređaju, Google je kreirao RAIL Performance Model [13]. RAIL model ukazuje na to koje akcije je potrebno optimizovati u aplikacijama i u kojim vremenskim granicama na način da će korisnici biti veoma zadovoljni.Na osnovu toga Web developeri se mogu koncentrisati na optimizaciju pojedinačnih aktivnosti prema ***tabeli 1***.

Tabela 1. *optimizacija RAIL performansi*

Aktivnost	Opis RAIL Performanse
Odgovor (eng. Response)	Osnovna interakcija je tapkanje (eng. tap) i kliktanje na dugme ili ikonicu. Povratna informacija korisniku je potrebno da se prikaže za manje od 100 milisekundi nakon inicijalnog inputa korisnika. Svrha ovoga jeste da se korisnik obavesti da je njegova akcija registrovana.
Animacija (eng. Animation)	Animacija može da uključi vizuelne animacije kao što su npr. ulazne i izlazne animacije i indikator učitavanja, zatimskrolovanje (eng. scrolling) i povlačenje (eng. drag). Da bi korisnik imao utisak da je animacija glatka, svaki okvir (eng. frame) je potrebno završiti za manje od 16 milisekundi, tako da se ostvari 60 okvira po sekundi (eng. frames per second), FPS.
Stanje mirovanja (eng. Idle)	Usled ograničenih resursa mobilnih uređaja je potrebno optimizovati procesiranje i korisniku prikazati na ekranu samo ono što je neophodno, dok se vreme mirovanja koristi da se završi sav odloženi posao. Odloženi posao je potrebno grupisati u blokove od 50ms, tako da ukoliko korisnik započne interakciju tada odgovor na istu ima najviši prioritet.
Učitavanje (eng. Load)	Sadržaj je važno isporučiti korisniku za 1 sekundu. Naravno ne mora se sve učitati u tom intervalu, već je potrebno omogućiti progresivno renderovanje i u pozadini učitati ostatak. Potrebno je fokusirati se optimizovanje kritične putanje renderovanja (eng. critical rendering path).

3.2. *Progressive Web apps*

Nakon što su definisali RAIL model, inženjeri Googla su razmišljali o tome kako da unaprede korisničko iskustvo na mobilnim Web aplikacijama. Obzirom da korisnici provode dosta vremena u nativnim aplikacijama, shvatili su da ukoliko mobilnim Web aplikacijama na neki način obezbede ključne karakteristike nativnih aplikacija, a pri tome očuvaju dobre strane Web aplikacija imaće pravo rešenje. Pogodnosti Web aplikacija se odnose na jednostavno otkrivanje sadržaja upotrebom search engine-a ili deljenjem linkova, zatim interakcija je veoma lagana i nema velikih promena u iskustvu već je sve udaljeno na jedan klik, zatim širok zahvat jer Web radi na svim uređajima koji imaju pretraživač, što je danas velika većina uređaja, pored toga Web je otvoren i decentralizovan, a ono što je možda najvažnije jeste da je stalno ažuran, tj. nema potrebe za update-ovim i patch-evima, tako da je njegova moć distribucije velika [15].

Sa druge strane, nativne aplikacije imaju karakteristike koje nedostaju mobilnim Web aplikacijama: pouzdano se pokreću i mogu da rade u offline modu, mogu da budu brze, koriste push notifikacije, sinhronizuju se u pozadini i pokreću se sa startnog ekrana (eng. home screen) mobilnog uređaja. Potrebno je imati u vidu da pametni telefoni i tableti su manji i imaju manje snage od Desktop računara, manje memorije, ekran osetljiv na dodir i koriste mobilne mreže. Ukoliko se pokrene Web pretraživač i ode na neki Web sajt korisničko iskustvo neće biti dobro kao u slučaju nativnih aplikacija, zbog navednih karakteristika.Da bi korisničko iskustvo na mobilnom Web-u bilo dobro kao u nativnim aplikacijama, a pri tome se zadržale sve pogodnosti Web-a, od kojih je za dostizanje velikog broja korisnika širok zahvat najzanačajnija, Google je dao 4 temelja (eng. pillars) na kojima treba da se bazira iskustvo zasnovano na mobilnim Web aplikacijama [16]:

1. ubrzavanje iskustva(eng. accelerate the experience) – Učiniti sve bržim. Učiniti da se stranice učitaju brzo, učiniti da se skroluju brzo, to je ono što veliki broj ljudi privlači.
2. prijatno iskustvo (eng. engaging experience) – Jednom kada su korisnici privučeni obezbediti prijatno i impresivno iskustvo, jer to je ono zbog čega ljudi ostaju.
3. konvertovanje (eng. convert) – Jednom kada imate brzo i prijatno iskustvo, imaćete dosta posetilaca jer Web pružaširok zahvat, tako da je potrebno konvertovati posetioce u lojalne korisnike.
4. zadržavanje (eng. retain) – Jednom kad imate korisnike, želite da ih zadržite, želite da oni ponovo dođu i da ih vratite na iskustvo u pravo vreme.

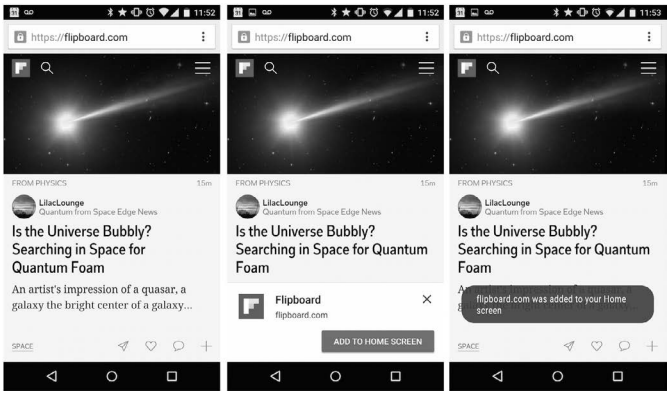
Mobilne Web aplikacije koje obezbeđuju ova 4 temelja, kombinujući najbolje karakteristike nativnih aplikacija i Web aplikacija nazivaju se Progressive Web apps [16], skraćeno PWA. One ne moraju da se instaliraju i kako korisnik napreduje u upotrebi aplikacije vremenom, aplikacija postaje sve bolja. Aplikacija se učitava brzo, čak i na mrežama sa manjim propusnim opsegom, šalje push notifikacije, ima ikonicu koja je dostupna na početnom ekranu i pruža iskustvo punog ekrana [16].

PWA imaju sledeće karakteristike [17]:

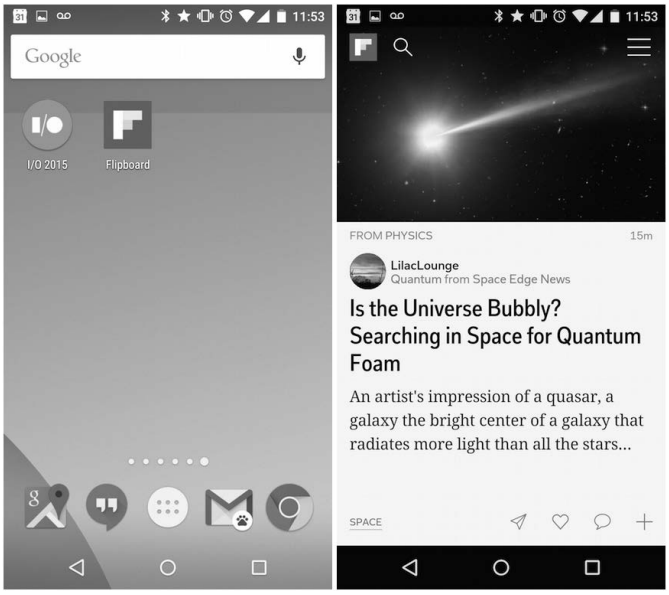
- progresivnost – moraju da rade na bilo kojem uređaju i da se progresivno poboljšavaju, tako da koriste prednosti bilo kojih svojstava dostupnih na korisnikovom uređaju i pretraživaču
- vidljivost – zato što su PWA zapravo Web sajtovi, potrebno je da budu vidljivi u search engine-ima. Ovo je glavna prednost u odnosu na nativne aplikacije, koje i dalje kaskaju u odnosu na Websajtove u kontekstu pretraživanja
- povezivost – kao i kod Web sajtova potrebno je koristiti Uniform Resource Identifier, skraćeno URI, za predstavljanje stanja aplikacije. Ovo omogućava web aplikacija da održe stanje ili da ga ponovo učitaju kada korisnik bookmark-uje ili podeli URL aplikacije.
- responzivnost – korisnički interfejs PWA mora da se prilagodi veličini ekrana
- izgled poput aplikacije – PWA je potrebno da izgleda kao nativna aplikacija i da bude izgrađena na application shell model-u, sa minimalnim osvežavanjem stranica
- nazavisno povezivanje – potrebno je da aplikacija radi u oblasti slabe veze ili offline
- ponovna upotrebljivost – korisnici vole da ponovo koriste svoje nativne aplikacije i PWA je potrebno da ostvare iste ciljeve kroz push notifikacije
- mogućnost instaliranja – PWA mogu da se instaliraju na startni ekran uređaja, tako da su odmah dostupne
- svež sadržaj – kada se novi sadržaj objavi i korisnik je povezan na Internet, taj sadržaj je potrebno da bude dostupan u aplikaciji
- sigurnost – zato što PWA imaju intenzivnije korisničko iskustvo i celi mrežni zahtevi se prosleđuju kroz service worker-e, potrebno je aplikaciju zaštitit od man-in-the-middle napada preko HTTPS-a

Jedan tok izvršenja PWA na primeru, izgleda sledeće [18]:

1. korisnik mobilnog uređaja sa Android OS preko Google Chrome pretraživača odlazi na sajt flipboard.com i stranica mu se prikazuje regularno, kao na ***slici 9***. na ekranu skroz levo
2. kada korisnik drugi ili treći put poseti sajt, i pretraživač prepozna da želi da uspostavi trajnije iskustvo, pretraživač pita korisnika da li želi da doda aplikaciju na startni ekran, kao na ***slici 9.*** na ekranu u sredini
3. ukoliko korisnik odluči da zadrži aplikaciju na startnom ekranu tada tok nije prekinut i interakcija se nastavlja, kao na ***slici 9***. na ekranu skroz desno
4. kada se PWA aplikacija pokrene sa startnog ekrana korisnik može da odabere full-screen mod, kao na ***slici 10.***



Slika 9. *Tok dodavanja PWA aplikacije na startni ekran*



Slika 10. *Pokretanje PWA aplikacije*

Da bi neki mobilni Web sajt postao PWA potrebno je dodati sledeće [19]:

- manifest (eng. Web App Manifest)
- Service Worker
- TLS protokol (eng. Transport Layer Security)
- responzivni dizajn (eng. Responsive Design)

Manifest Web aplikacije je ništa drugo do JSON fajl koji daje mogućnost programeru da kontroliše kako će aplikacija prikazati korisniku u delu u kojem on očekuje da vidi aplikaciju, kao što je startni ekran mobilnog uređaja (eng. home screen) i upravlja šta i kako korisnik može da pokrene. Manifest obezbeđuje informacije o aplikaciji kao što su ime, autor, ikonica i opis u tekstualnom fajlu [20]. Njegova svrha je da instalira Web aplikaciju na startni ekran i time obezbedi brži pristup i bogatije iskustvo. U budućnosti plan je da manifest ima još više kontrole nad aplikacijom. Prvi korak u dodavanju manifesta na Web sajt jeste njegovo kreiranje. Primer [21] manifesta je dat u ***kodu 1***.

```
{
  "lang": "en",
  "dir": "ltr",
  "name": "Super Racer 3000",
  "description": "Futuristic racing game !",
  "short_name": "Racer3K",
  "icons": [{
    "src": "icon/lowres.webp",
    "sizes": "64x64",
    "type": "image/webp"
  },{
    "src": "icon/lowres.png",
    "sizes": "64x64"
  }],
  "scope": "/racer/",
  "start_url": "/racer/start.html",
  "display": "fullscreen",
  "orientation": "landscape",
  "theme_color": "aliceblue",
  "background_color": "red"
}
```

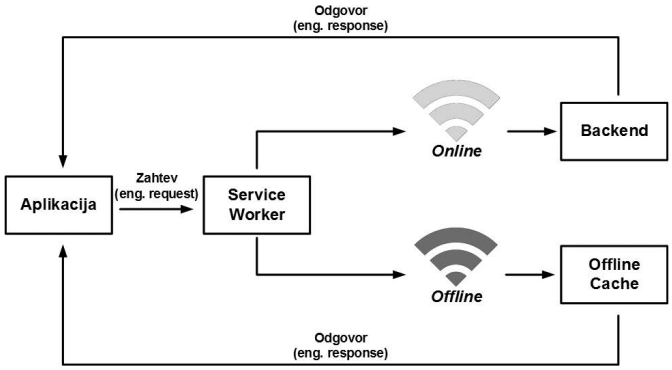
Kod 1. *Manifest Web aplikacije*

Nakon toga je potrebno kreirati HTML link tag koji će povezati [21] manifest sa HTML fajlom, ***kod 2.***

```
<link rel="manifest" href="/manifest.json">
```

Kod 2. *Referenciranje manifesta u HTML fajlu*

Service worker je HTML5 API koji obezbeđuje tehničku osnovu za bogato korisničko offline iskustvo, periodičnu pozadinsku sinhronizaciju, push notifikacije Web aplikacija, a koje su deo sveta nativnih aplikacija [22]. Pre service worker-a postojao je drugi API koji je obezbeđivao offline iskustvo i koji se naziva AppCache. Usled brojnih nedostataka [23] koji nisu prihvatljivi za mobilne Web aplikacije razvijen je service worker, koji može da presretne zahtev (eng. request) i upravlja istim uključujuću i programibilno upravljanje kešom (eng. cache) sa odgovorima (eng. response). Na taj način on deluje kao proxy server [24] koji se nalazi između Web aplikacije i pretraživača i mreže (kada je dostupna). Način na koji service worker radi je prikazan na ***slici 11***.



Slika 11. *Service worker kao proxy server*

Aplikacija šalje zahtev (eng. request) za određenom Web stranicom, tj. resursom, ukoliko je moguće uspostaviti konekciju, tada se prosleđuje odgovor (eng. response) preko backend dela sistema, a ukoliko je offline mod u pitanju, tada se podaci preuzimaju iz keša i oni prosleđuju kao odgovor. Na pomenuti način aplikacija se osigurava da prikaže sadržaj, kao kod nativnih aplikacija. Važno je napomenuti da service worker mora da radi preko HTTPS-a, da bi se izvršila prevencija tzv. „man-in-the-middle" napada. Prilikom rada sa service worker-om prvi korak je njegovo registrovanje unutar stranice [25]. U JavaScript ***kodu 3.***, vidimo način kako se to sprovodi.

```
if ('serviceWorker' in navigator) {
navigator.serviceWorker.register('/sw.js').
then(function(registration) {
console.log('ServiceWorker registration successful: ',
registration.scope);
  }).catch(function(err) {
console.log('ServiceWorker registration failed: ', err);
  });
}
```

Kod 3. *Registrovanje Service worker-a*

Nakon što je registrovan service worker u JavaScript fajlu sw.js, može nastaviti da radi. Dva važna događaja koje service worker je potrebno da osluškuje su install i fetch. Tokom faze

instaliranja se najčešće keširaju statički resursi, kao što su stilovi, skripte, fontovi i neke .html stranice. Tokom fetch faze kao odgovor se vraća keširani objekat ili se sa aktuelnog URL-a dopremaju resursi online.

Kompanije koje su usvojile PWA u ranoj fazi [15], već imaju vidljive benefite. Tako npr. Flipkart najveći indijski e-commerce sajt je produžio vreme korisnika provedeno na sajtu na 3.5 minuta, u odnosu na period ranije od 70 sekundi. Pored toga korisnici se vraćaju ponovo da koriste PWA Flipkart Lite i to 40% više nego ranije, dok je broj lojalnih korisnika porastao na 70% tako što su dodali ikonicu na startni ekran i na sve to 3 puta manje koriste podatke. Novinska agencija Washington Post je poboljšala redovno vraćanje korisnika sa 51% na 63% i znatno uvećala brzinu, na skoro 80 milisekundi po članku, kombinovanjem PWA koncepata sa drugim Google-ovim poboljšanjem za mobilne Web aplikacije Accelerated Mobile Pages, skraćeno AMP.Avio kompanija Air Berlin, druga najveća avio kompanija u Nemačkoj, je upotrebom PWA značajno unapredila korisničko iskustvo na mobilnim uređajima tokom check-in procesa. AliExpress, vodeća globalna online prodavnica, je unapredila korisničko iskustvo tako da korisnici sada provode 74% više vremena na Web sajtu.

4. ZAKLJUČAK

Progressive Web apps kombinovanjem najboljih karakteristika nativnih aplikacija i mobilnih Web aplikacija donose najbolje od oba sveta i poboljšavaju korisiničko iskustvo. One korsite potencijal Web-a kao platforme za distribuciju, koji ne zahteva instaliranje, dodeljivanje prava, konfigurisanje i ažuriranje verzija, jednostavno otkrivanjei deljenje sadržaja, takođe podržavaju rad u offline modu, push notifikacije, sinhronizaciju u pozadini i pokretanje sa startnog ekrana mobilnog uređaja, kao kod nativnih aplikacija. Skup svih navedenih karakteristika doprinosi da mobilni Web postaje sve značajniji segment, kojem se krajnji korisnici sa zadovoljstvom vraćaju i provode vreme jer imaju brzo, ispolirano i na korak dostupno iskustvo. Kompanije sada imaju mogućnost da sa mnogo manjim troškovima ponude kroz svoje proizvode i usluge jedistveno iskustvo svojim klijentima, preko bilo koje platforme i bilo kojeg tipa uređaja kojim se pristupa, jer je Web svuda, na jedan klik.

Google kompanija značajno inovira i trenutno je nekoliko relativno novih API-a dostupno, zajedno sa PWA, vidimo i AMP aplikacije koje dodatno utiču na performanse mobilnih Web aplikacija, kao i Credential Management API za pojednostavljivanje postupka autentikacije na Web-u, tzv. „one-tap sign in“, kao i Web Payments API za uprošćavanje ranije komplikovanog checkout postupaka prilikom kupovine, tzv. „one-tap checkout“. Takođe novi Fetch API je zamišljen kao zamena za XMLHttpRequest API, a Web Stream API kao trenutno najbolji način za služenje sadržaja. Svi ovi pokazatelji govore da će Web kao platforma nastaviti da evoluira i pomera granice između poznatog i nepoznatog.

LITERATURA

[1] Conder S., Darcely L. (2011). *Android Wireless Application Development*(2nd ed.). Publisher: Addison-Wesley Professional

[2] Kurose J., Ross K. (2012). *Computer Networking: A Top-Down Approach* (6th ed.). Publisher: Pearson

[3] Vesić S., Minović M. (2015). *Single-Page Applications: trend ili budućnost.* In *Info M, Broj 55.* Fakultet Organizacionih Nauka

[4] Budiu R. (2015). *The State of Mobile User Experience*.https://www.nngroup.com/articles/mobile-usability-update/ (26.08.2016)

[5] Budiu R. (2013). *Mobile: Native Apps, Web Apps, and Hybrid Apps*. https://www.nngroup.com/articles/mobile-native-apps/ (26.08.2016)

[6] Pedotto K., McElyea J.P. (2015). *The 2015 U.S. Mobile App Report*. Publisher: comScore Inc.https://www.comscore.com/Insights/Presentations-and-Whitepapers/2015/The-2015-US-Mobile-App-Report (26.08.2016)

[7] *Fiksu Indexes for September 2015*. https://fiksu.com/resources/fiksu-indexes/2015/september (26.08.2016)

[8] „Google I/O Conference 2016.“ https://events.google.com/io2016/ (26.08.2016)

[9] „Progressive Web App Dev Summit 2016“ https://events.withgoogle.com/progressive-web-app-dev-summit/ (26.08.2016)

[10] „O'Reilly Fluent Conference 2016“ http://conferences.oreilly.com/fluent/javascript-html-us (26.08.2016)

[11] Nielsen J. (1993). *Response Times: The 3 Important Limits*. https://www.nngroup.com/articles/response-times-3-important-limits/ (26.08.2016)

[12] Tolia N., Andersen D., Satyanarayanan M. (2006). *Quantifying interactive user experience on thin clients*. In *Computer*,*Volume 39, Issue 3*. IEEE

[13] Irish P., Lewis P. (2015). *Introducing RAIL: A User-Centric Model For Performance*. https://www.smashingmagazine.com/2015/10/rail-user-centric-model-performance/(26.08.2016)

[14] Kearney M. *The RAIL Performance Model*. https://developers.google.com/web/tools/chrome-devtools/profile/evaluate-performance/rail?hl=en (26.08.2016)

[15] Roy-Chowdhury R. *The Mobile Web: State of the Union.* Google I/O Conference 2016. https://www.youtube.com/watch?v=0SSI8liELJU (26.08.2016)

[16] Russell A.*Progressive Web Apps*. Chrome Dev Summit 2015. https://www.youtube.com/watch?v=9Jef9IluQw0 (26.08.2016)

[17] LePage P. *Your First Progressive Web App* https://developers.google.com/web/fundamentals/getting-started/your-first-progressive-web-app/?hl=en (26.08.2016)

[18] Russel A. (2015).*Progressive Web Apps: Escaping Tabs Without Losing Our Soul*: https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/ (26.08.2016)

[19] Marfatiya S. (2016). *Is It Right Time to Go for Progressive App Development*. https://www.addonsolutions.com/blog/is-it-right-time-to-go-for-progressive-mobile-web-app-development-2.html (26.08.2016)

[20] *Web App Manifest.* https://developer.mozilla.org/en-US/docs/Web/Manifest (26.08.2016)

[21] *Web App ManifestW3C Working Draft*. https://www.w3.org/TR/appmanifest/ (26.08.2016)

[22] *What is a Service Worker*.https://developers.google.com/web/fundamentals/primers/service-worker/ (26.08.2016)

[23] Archibald J. (2012). *Application Cache is a Douchebag*. http://alistapart.com/article/application-cache-is-a-douchebag (26.08.2016)

[24] *Service Worker API*. https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API (26.08.2016)

[25] *Register a Service Worker*. https://developers.google.com/web/fundamentals/primers/service-worker/register?hl=en (26.08.2016)

MSc Slavimir Vesić, JKP „Beogradski vodovod i kanalizacija“
Kontakt: vesic.slavimir@gmail.com
Oblasti interesovanja: Web programiranje, interoperabilnost, softverske arhitekture, interakcija čovek računar