Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/09EF04Djurovic

POSTUPAK IZRADE VEB APLIKACIJE U EMBER.JS RADNOM OKVIRU

DEVELOPMENT OF WEB APPLICATION IN EMBER.JS FRAMEWORK

Dunja Đurović, Neda Milić Keresteš, *Fakultet tehničkih nauka, Novi Sad*

Oblast – Grafičko inženjerstvo i dizajn

Kratak sadržaj – *Rad se bavi istraživanjem radnih okvira (engl. framework), za razvoj veb aplikacija. Cilj rada jeste prikazati najbolju praksu izrade veb aplikacije, način na koji rade ovakve aplikacije i kada se one koriste. Takođe, cilj je istražiti aktuelne tehnologije za razvoj veb aplikacija i napraviti komparativnu analizu tih tehnologija za konkretne primene. Rad predstavlja potpuni razvoj jedne veb aplikacije pomoću Ember.JS radnog okvira. – od samog planiranja aplikacije, preko dizajna do programiranja i puštanja u rad.*

Ključne reči: *veb aplikacije, radni okviri, veb dizajn Ember.JS*

Abstract – *This work reviews frameworks for development of web applications. The aim of this paper is to present the best practice to develop a web application and to explain how web applications work and when they are used. Also, the thesis is analyzing current technology trends for web application development and comparing those technologies for specific usage. The focus of the work is based on the web application development in Ember.js framework from the planning , web design, programming to web application deployment.*

Keywords: *web applications, framework, web design, Ember.JS*

1. UVOD

Razvoj tehnologija donosi potrebu za različitim promenama i prilagođavanjima u sferi veb aplikacija i veb sajtova. Promene koje nastaju razlog su sve veće upotrebe interneta i internet pretraživača. Sve manji broj korisnika instalira aplikacije, a razlog upravo leži u ekspanziji upotrebe veb aplikacija [1].

Veb aplikacije su lake za korišćenje i ne zahtevaju instalaciju kao u slučaju konvencionalnih aplikacija, već je potrebno samo pronaći aplikaciju pomoću veb čitača i imati pristup internetu [2].

Kako bi se izradile ove aplikacije, potrebno je imati bolji uvid šta su to veb aplikacije i način na koji one rade. Potom, bitno je prikazati procese koji dovode do konačnog proizvoda – a to su koraci poput planiranja izrade aplikacije, dizajna aplikacije, razvoja aplikacije do puštanja aplikacije u rad.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Neda Milić Keresteš, docent.

2. VEB APLIKACIJE

Kako je značaj veb aplikacija u konstantnom porastu, važno je objasniti šta su to veb aplikacije, razgraničiti ih od veb sajtova i klasičnih aplikacija koje se koriste na računarima. Takođe, bitno je pomenuti koje sve vrste veb aplikacija postoje i šta ih razlikuje, a šta im je zajedničko.

2.1. Šta su to veb aplikacije?

Veb aplikacije bi se mogle objasniti kao softverske aplikacije koje su svakodnevno u upotrebi tako da je njihov kod smešten na serveru i za prikaz koriste veb pretraživač [1].

Glavna razlika između veb sajtova i veb aplikacija jeste ta što su veb aplikacije interaktivne, a veb sajtovi imaju ulogu prezentacije. Na primer, ukoliko se poseti veb sajt nekog restorana, a tamo su prikazane informacije, u vidu jelovnika, kontakt informacija i opštih informacija može se reći da je korisnik posetio veb sajt. Međutim, ako je takav sadržaj dopunjen online naručivanjem, mogućnošću kucanja promo kodova, korišćenjem poklon kartica tada se radi o veb aplikaciji.

Sa druge strane značajno je razlikovati mobilne aplikacije (engl. *Native Applications*) od veb aplikacija. Naime, ako je reč o mobilnim aplikacijama potrebno je znati da su ovakve aplikacije izrađene posebno za mobilne uređaje. S obzirom da su one razvijene za operativne sisteme mobilnih telefona, potrebno je uvek prvo preuzeti instalaciju i instalirati ovakvu aplikaciju na uređaj.

2.2. Vrste veb aplikacije

Postoji nekoliko različitih vrsta veb aplikacija. Prva podela do koje se može doći jeste podela na statične i dinamične.

Statične su veoma jednostavne i uglavnom su to stranice koje su prezentacionog karaktera, ali opet sadrže neku vrstu interakcije sa korisnikom.

Dinamične aplikacije su kompleksnije i zahtevaju više kodiranja i razvoja. S obzirom da je njihov sadržaj izmenjiv, može se napraviti dalja podela po sadržaju na:

- E-prodavnice (engl. *E-commerce*)
- Portal Veb aplikacije
- Animirane veb aplikacije [3].

Sa aspekta tehnologije, može se napraviti podela veb aplikacija na SPA (*Single Page Apps*), MPA (*Multiple Page Apps*) i PWA (*Progresive Web Apps*) aplikacije.

- **SPA (engl. *Single Page Applications*)** su vrsta veb aplikacija koje svoj sadržaj ponovo iscrtavaju prilikom akcija navigacije (na primer, prilikom klika na link), ali bez ponovnog slanja zahteva serveru [1].
- **MPA (engl. *Multiple Page Applications*)** aplikacije su potpuno prilagođene načinu funkcionisanja veb čitača. Svaka interakcija, odnosno navigacija kroz aplikaciju, dovodi do slanja zahteva serveru i ponovnog učitavanja nove HTML stranice [2].
- **PWA (engl. *Progresive Web Applications*)** se mogu nazvati i hibridnim aplikacijama jer predstavljaju kombinaciju veb stranica i mobilnih aplikacija.

3. VOĐENJE PROJEKTA

Najznačajnija stvar na početku projekta jeste razumevanje zahteva projekta. Tek kada su svi zahtevi projekta jasni može se krenuti sa daljim fazama. Faze koje slede su procena vremena, planiranje, izrada aplikacije (dizajn, front-end, back-end), kontrola izrade i završetak projekta (utvrđuje se da li su svi delovi projekta završeni i da li su prihvatljivi, a zatim se piše završna dokumentacija) i, eventualno, kasnije održavanje projekta.

Pre početka svakog projekta, veoma je bitno isplanirati svaki korak koji će dovesti do krajnjeg proizvoda koji ispunjava sve uslove koje je klijent zahtevao. Kako bi se mogao organizovati tim ljudi koji će raditi na projektu, potrebne su različite tehnike planiranja resursa i vremena da se projekat isporuči na vreme. Ukoliko se planiranje preskoči ili se svede na minimum, moguće je da se projekat neće završiti na vreme ili neće biti zadovoljavajućih rezultata. Na početku je veoma teško odrediti koliko je vremena potrebno za svaku aktivnost, ali iskustvom i poznavanjem rada u timu, lakše se predviđa ishod.

4. PROCES RAZVOJA APLIKACIJE – DIZAJN

Kako bi veb aplikacija imala zahtevni kvalitet, potencijalni korisnik aplikacije se stavlja u fokus i vrše se brojna ispitivanja, istraživanja i konsultacije.

Proces koji je potrebno ispratiti je sledeći: empatija sa korisnicima, definisanje potreba na osnovu dobijenih rezultata, ideja ili skice i UX prototip. Na početku dizajn faze se, na osnovu prethodno izvedenih anketa i istraživanja potencijalnih korisnika, prave *User Persona* šabloni koji predstavljaju fiktivne reprezentativne korisnike. Takođe se testira funkcionalnost sajta ili aplikacije (*Usability testing*), odnosno koliko je korisniku lako i intuitivno koristiti proizvod. Prave se različiti scenariji i situacije, kao i zadaci za korisnike. Zatim se vrši identifikacija potreba, kao i sam prioritet potreba na osnovu prethodno navedenog istraživanja. Kada je ideja o samom proizvodu definisana i jasnije su potrebe klijenta, moguće je napraviti grube skice ekrana (engl. *wireframes*). Kada se usvoje početne skice, prave se detaljniji UX prototipi, na osnovu kojih se dalje izrađuje krajnji dizajn korisničkog interfejsa.

5. PROCES RAZVOJA APLIKACIJE– FRONT END

Nakon procesa dizajna, može se nastaviti dalje sa izradom front-end dela aplikacije. Ova faza obuhvata kreiranje i stilizaciju elemenata i komponenata prema odobrenom dizajnu. Za ovaj proces je, pored HTML i CSS tehnologija, potrebno poznavati *JavaScript* jezik, kao i odgovarajuće JS radno okruženje.

5.1. JS Framework

Prilikom izrade veb sajtova ili veb aplikacija, za funkcionalnost zadužen je JavaScript. S obzirom na to da se kod sajtova najviše vrši DOM manipulacija, u većini slučajeva, nije potrebno koristiti *framework* već samo neku od biblioteka. Veb aplikacije su dosta kompleksnije i pisanje čistog *JavaScript* koda dovelo bi do značajnog utroška vremena. Zbog toga se koriste JS radne okvire - skup JS biblioteka koji programerima olakšavaju utoliko što postoji osnova određenog dela koda koji se može iskoristiti. Može se reći da je postavljen temelj za dalju izradu komponenata i povezivanje sa API interfejsom [4].

5.2. Stilizacija

Posle izrade *front-end* funkcionalnosti, sledi stilizacija prema dizajnu. CSS pretprocesori predstavljaju jezik za stilizaciju, koji ima drugačiji način pisanja CSS koda i može se napraviti struktura koja će olakšati pisanje stila određenih komponenata. Na kraju se od svih datoteka koje sadrže kod pisan pretprocesorom, generiše jedna datoteka sa stilom, koju će aplikacija ili sajt koristiti za prikaz svog sadržaja. U zavisnosti od odabranog radnog okvira, koriste se različite vrste šablona u kojima je ispisan HTML (na primer, *Ember.js* koristi *Handlebars* šablon).

6. PROCES RAZVOJA APLIKACIJE – BACK END

Za razliku od *front-end* dela aplikacije, koji služi da kreira klijentski deo, odnosno izgled aplikacija ili veb sajta, *back-end* deo je softver koji se nalazi na serveru zaduženom za izvršavanje zahteva koji su mu poslati. Server sadrži kod kojim će doći do komunikacije sa bazom podataka, "kupljenja" podataka iz date baze i potom spremanja generisanih podataka za *front-end* deo. Na ovaj način se radi sa podacima koji su dinamični. Da je reč o statičnim podacima, ne bi bilo potrebe za većom ili uopšte komunikacijom sa serverom. *Back-end* deo je neophodan i kada postoje jednostavne stvari kao što je forma koju treba popuniti i sačuvati i proslediti podatke iz forme, ai kod mnogo kompleksnijih softverskih rešenja kao što je *Google search* aplikacija [5].

7. PROCES RAZVOJA VEB APLIKACIJE – PUŠTANJE APLIKACIJE U RAD

Na kraju procesa izrade aplikacije, ostaje puštanje iste u rad. Sam proces predstavlja postavljanje datoteka (u ovom slučaju paketa) na neki od javnih servera. Način na koji će kod biti postavljen na server zavisi od samog projekta. Naravno, ovaj proces se ne obavlja samo jednom, već se vrši puštanje u rad, a prilikom izmena i samog održavanja, potrebno je obaviti neke zadatke kao i pri prvom *deployment* procesu.

8. PRAKTIČNI DEO

U ovom poglavlju biće više reči o samoj izradi jedne SPA aplikacije. Reč je o aplikaciji u demo svrhe za pisanje

recepata koja omogućava dodavanje recepata, njihovo uređivanje i upravljanje. Razvoj u bilo koje druge svrhe i potrebe bio bi suštinski sličan.

8.1. Opis tehnologija

Za potrebe ove aplikacije korišćene su tehnologije *Ember.js* - *front-end* framework, *Symfony* - back-end framework i *Firebase* platforma preko koje je izvršeno puštanje aplikacije u rad. S obzirom da je fokus na *front-end* izradi aplikacije, više reči će biti o *Ember.js* radnom okviru.

Ember.js je MVC (engl. *Model-View-Controller*) klijentski radni okvir koji se koristi za izradu SPA aplikacija, koje zahtevaju kompleksne interakcije sa korisnikom. Ono što ovaj framework pruža jesu mogućnosti (engl. *features*) kojima se olakšava izrada kompleksnih operacija, kao i alat (engl. *toolkit*) koji ubrzava same procese kreiranja potrebnih datoteka za aplikaciju.

8.2 Prednosti i mane korišćenja JS framework sistema

Trenutno su najpopularniji JS *framework* sistemi *React*, Vue, *Angular*, *Ember* i *Backbone*. Kako bi se napravio što bolji izbor prilikom odluke kojom će se tehnologijom izraditi potrebna aplikacija, bitno je imati uvid u prednosti i mane svakih od njih. U ovoj listi koja je poznata i kao velika petorka (engl. *Big Five*) vršiće se odabir adekvatnog okvira na osnovu potreba aplikacije.

Prednosti Ember sistema su:

− Pouzdana i bogata arhitektura početnog stanja aplikacije

− Ember Data – sinhronizacija između modela i rute

− Uči se veoma lako i pogodan je za početnike

− Renderovanje na serverskoj strani

− Veliki broj mogućnosti (engl. *features*)

− URL podrška - (History API)

Mane Ember sistema su:

− Sporo renderovanje na početku

− Mala zajednica korisnika

− Razvoj aplikacije je složen i sporiji

− Podrška za nekoliko biblioteka.

Na osnovu prethodno istraženih mogućnosti okvira, izabran je *Ember.js* sistem. Pored toga što se lako uči, *Ember Data* omogućava lakšu komunikaciju sa serverom u odnosu na ostale sisteme.

Mogućnosti kao što su *Router* – datoteka u kojoj su definisane rute i *Computed Properties* – deklarisanje funkcija kao svojstva koja *Ember* automatski poziva, izdvajaju ovaj okvir. Doprinos izboru takođe daju mogućnosti koje se tiču kompatibilnosti sa veb čitačima [6].
S obzirom da je zadata aplikacija jednostavna, Ember.js ispunjava sve uslove za njenu izradu.

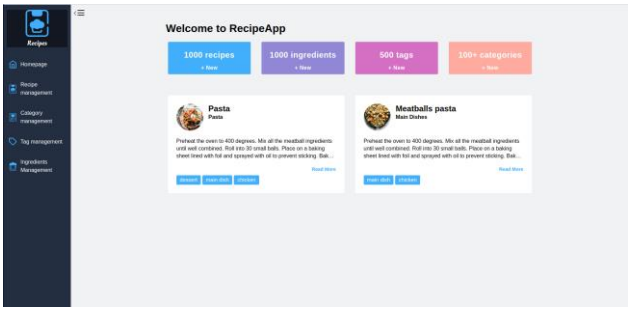
8.3 Planiranje aplikacije

Nakon utvrđivanja traženih zahteva aplikacije *Recepti*, izvršena je analiza u smislu potrebnog vremena za izradu iste pomoću metode Bottom-up. Zadaci su podeljeni na celine kao što su u ovom slučaju izrada komponenti, zatim izrada CRUD (engl. *Create, Read, Update and Delete*) funkcija, a na kraju sama prezentaciona stranica recepata.

8.4 Dizajn aplikacije

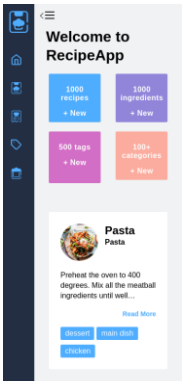
Podrazumeva se da se prilikom izrade dizajna aplikacije obuhvata i UX i UI deo. Na samom početku su izrađuju se grubi prototipi koje se nazivaju *LoFi* prototipi. Na osnovu njih izrađeni su u narednoj etapi i detaljniji, *HiFi* prototipi koji podrazumevaju i izabrane boje, kao i odnos veličine fonta (Slika 1). Nakon nekoliko iteracija, kreira se i konačni dizajn stranica (Slika 2).

Slika 1. *HiFi prototip početnog ekrana aplikacije*



Slika 2. *Dizajn početnog ekrana aplikacije*

Dizajn je za manje ekrane, odnosno responzivnost je urađena na osnovu prethodnih prototipa (Slika 3).



Slika 3. *Responzivni dizajn početnog ekrana aplikacije*

8.5 Izrada aplikacije – *Front-end* celina

U ovom poglavlju biće predstavljena izrada *front-end* dela aplikacije, pomoću *JavaScript* sistema *Ember.js*.

Kako bi bio potpuno ispraćen proces zadataka koji su definisani u procesu planiranja, prvo će biti izrađen sam kostur aplikacije pomoću *Ember CLI* interfejsa, zatim komponente i modeli koje će biti značajni za sve rute, potom izrada CRUD funkcija i prezentaciona stranica recepata.

Za početak, veoma je bitno izraditi sam kostur aplikacije. S obzirom da je moguće koristiti Ember CLI, izrada aplikacije je veoma jednostavna – vrši se kucanjem komande u terminalu **ember new <app-name>**. Posle pokretanja ove komande dobija se kostur aplikacije, sa hijerarhijom koja je prilagođena ovom sistemu.

Bitno je obratiti pažnju na deo gde se definiše *API host* u development okruženju. U ovom slučaju povezana je aplikacija sa http://127.0.0.1:8000/api sa portu na lokalnom računaru na kojem funkcioniše server. Putanja/api prikazuje lokaciju gde su definisane rute na back-end delu. Na ovaj način je *front-end* deo aplikacije povezan sa *back-end* delom i sada je konfiguracija izvršena. Može se nastaviti sa daljom izradom aplikacije.

Prilikom izrade modela, definiše se koji je tip podataka koji će biti unet (uglavnom je reč o string tipu) ili o vezi među samim modelima. S obzirom da aplikacija sadrži recepte, tagove, kategorije i sastojke, biće izrađeni modeli za svaki od njih.

Posle izrade modela, može se preći na izradu komponenti. S obzirom da su podaci koji se dodaju dinamični, moraju se definisati na osnovi komponenti.

Kada su kreirane osnovne funkcionalnosti, dodaju se funkcije poput dodavanja slika i ispisivanje grešaka. Prilikom dodavanja slika, bitno je naglasiti da se kroz šablone ubacuje posebno input polje sa atributom type=”file”. Što se tiče izbacivanja grešaka, postupak se odnosi na ispis grešaka koje su definisane u back-end delu. Ispisivanje grešaka se izvršava kada korisnik ne unese vrednosti koje su definisane kao obavezne. Tada se ispiše tekst koji korisniku označava kako grešku treba ispraviti.

9. ZAKLJUČAK

Izrada SPA aplikacija je veoma složen proces. Dato je detaljno objašnjenje šta su to SPA aplikacije i način na koji rade, zatim poređenje SPA sa MPA i PWA aplikacijama i poređenje sa veb sajtovima. Takođe, detaljno su opisani procesi izrade aplikacije – planiranje, dizajn, *back-end*, *front-end* deo (koji obuhvata korišćenje Ember.js okvira i stilizaciju), kao i plasiranje aplikacije.

Pored opisa svakog od procesa, prikazana je i aplikacija koja je izrađena na prethodnim principima. Reč je o aplikaciji koja ima ulogu dodavanja recepata, kao i njihov prikaz, izmenu i uklanjanje. Aplikacija se može razvijati i dalje uz različite mogućnosti - moguće je dodati sistem registracije korisnika, zatim vršenje nekih filtriranja i sličnih funkcionalnosti.

Izradu SPA aplikacije je prigodno podeliti na nekoliko celina u timu od nekoliko ljudi, gde će vođa projekta da izvrši planiranje, veb dizajner će izvršiti UX istraživanje i napraviti UI dizajn cele aplikacije, a potom će programeri (*back-end* i *front-end*) izraditi funkcionalnost i stil.

U budućnosti se očekuje ekspanzija SPA aplikacija. Do ovog zaključka dovode činjenice da se razvija sve više novih radnih okvira, a postojeći se konstantno ažuriraju i poboljšavaju. Takođe, u prilog SPA aplikacijama ide i činjenica da se sve više korisnika okreće korišćenju ovog jednostavnijeg rešenja koje podrazumeva korišćenje aplikacije bez instalacije.

10. LITERATURA

[1] P. Sherman, "blog.pshrmn", Blog.pshrmn.com, 2020. [Online]. Dostupno na: https://blog.pshrmn.com/how-single-page-applications-work/. [Pristupljeno u septembru 2019].

[2] Single-page application vs. multiple-page application", Medium, 2016. [Online]. Dostupno na: https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58. [Pristupljeno u septembru 2019].

[3] What are the types of web applications?, Blog.exposit.com, 2018. [Online]. Dostupno na: https://blog.exposit.com/what-are-types-web-applications/. [Pristupljeno u oktobru 2019].

[4] S. Morris, "What Is a JavaScript Framework? - Skillcrush", Skillcrush, 2018. [Online]. Dostupno na: https://skillcrush.com/blog/what-is-a-javascript-framework/. [Pristupljeno u novembru 2019].

[5] Startit, "Kako naučiti back end web programiranje", 2020. [Online]. Dostupno na: https://startit.rs/back-end-web-development/. [Pristupljeno u oktobru 2019].

[6] Pluralsight, "7 Reasons to Use Ember.js", Pluralsight.com, 2015. [Online]. Dostupno na: https://www.pluralsight.com/blog/software-development/7-reasons-to-use-ember-js. [Pristupljeno oktobra 2019].

Kratka biografija:

Dunja Đurović rođena je u Zrenjaninu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Grafičkog inženjerstva i dizajna odbranila je 2020.god.
kontakt: djurovicdunja.9@gmail.com