Sistem za kreiranje specifikacije i kontrolisanje razvoja angular komponenti

Petar Erić

Sadržaj – Predmet rada je projektovanje modela baze podataka u kojoj bi se skladištila dokumentacija o načinu povezivanja komponenti, baziranih na Angular razvojnom alatu, tokom projektovanja i razvoja veb aplikacije. Cilj je formirati setove atributa koji precizno opisuju Angular komponentu, njenu vezu sa drugim komponentama i definišu različite prioritete pristupa. Metodološki, u izradi rada kritički se porede različiti pristupi u vođenju dokumentacije tokom izrade računarske aplikacije i potom se predstavlja novo rešenje, specijalizovano za izabrani razvojni alat.

Ključne reči: *Angular komponente, REST API, specifikacija, dokumentacija, JWT, baza podataka*.

I. UVOD

RAD predstavlja prirodan nastavak istraživanja započetog u diplomskom radu, pod nazivom "Registracija članova sportskih društava", gde su opisani koraci u realizaciji veb aplikacije i naveden način kreiranja dokumentacije na osnovu koje je obavljeno njeno projektovanje. Osnovna aplikacija se bazira na klijentskoj strani realizovanoj kroz programski jezik Angular, serverskoj strani realizovanoj u programskom jeziku GoLang i bazi podataka MySql.

Nadograđena aplikacija koristi iste tehnologije, sa akcentom na povećani stepen sigurnosti. Uveden je koncept različitih nivoa iste aplikacije, koja se kombinuje sa višestrukim korisničkim rolama i različitim vrstama osetljivih dokumenata, koji zahtevaju dodatne korisničke prioritete. Nova aplikacija ima značajno uvećan broj radnih ekrana, pa se pretpostavlja da za je za izradu finalne aplikacije neophodno angažovanje programerskog tima.

U prvom delu rada detaljno su opisane tehnologije koje su korišćene u realizaciji projekta. Naročita pažnja je usmerena ka tokenima, na kojima se

bazira najosetljiviji deo provere prioriteta u projektu.

U sledećem delu opisani su programski alati preko kojih su kreirani delovi dokumentacije. Iz svakog alata, finalni dokumenti su eksportovani i potom prebačeni u namenski kreiranu bazu podataka.

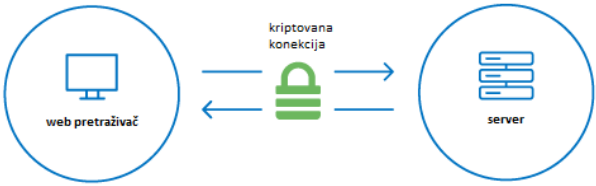
U središnjem delu rada najpre se prikazuju projektni zahtevi za početnu aplikaciju, relativno skromnih mogućnosti. Daje se kratki opis realizovane dokumentacije koja je prethodila izradi ove aplikacije. Potom je opisana specifikacija naprednije verzije iste aplikacije, koja uključuje značajno veći broj programskih modula i veće zahteve u pogledu sigurnosti.

U delu "realizacija" opisan je način skladištenja podataka osnovne dokumentacije. Potom je opisana mogućnost korišćenja osobina JWT radi postizanja efikasne kontrole prioriteta u programu. Prikazan je način podešavanja različitih nivoa aplikacije, kontrola prilagodljivog menija, kontrola korisničkih rola i pristupa dokumenata kroz kombinovanje osobina JWT i relacija između ključnih entiteta u bazi dokumentacije. Slično tome, prikazan je mehanizam upravljanja sigurnosti API poziva. Budući da je u projektu planirana upotreba najmanje četvorojezične varijante, prikazana je realizacija mehanizma za kontrolu prevoda na klijentskoj i serverskoj strani.

Nakon toga, prikazane su prednosti realizovanog rešenja u odnosu na klasičnu dokumentaciju, i to po fazama razvoja projekta: projektovanje, razvoj i ekspoloatacija.

II. PREGLED KORIŠĆENIH TEHNOLOGIJA

Radi postizanja veće sigurnosti u projektu, korišćene su tehnologije HTTPS sertifikata i JWT tokeni [9],[11]. Dodatno, svi osetljivi korisnički podaci (npr. lozinke korisnika) su kriptovane pre pamćenja u bazi.[6] Komunikacija između klijenta i servera bazira se na JSON formatu poruka (slika 2.1). Serverska strana je realizovana pomoću programskog jezika *GoLang*, a za klijentsku stranu korišćen je programski jezik *Angular* [3], [14].



Slika 2.1 Kriptovana konekcija između klijenta i servera

III. PREGLED KORIŠĆENIH ALATA

Za kreiranje različitih delova dokumentacije o projektu, postoji niz specijalizovanih alata pomoću kojih se grafički mogu predstaviti komponente ili veze između njih. U okviru ovog projekta, korišćeni su alati *Power Designer 15, Balsamiq Mockups 3, Visual Studio Code, Postman* i *MS Access*. U daljem tekstu, biće predstavljene osnovne karakteristike ovih alata.

IV. SPECIFIKACIJA APLIKACIJE I PRATEĆE DOKUMENTACIJE

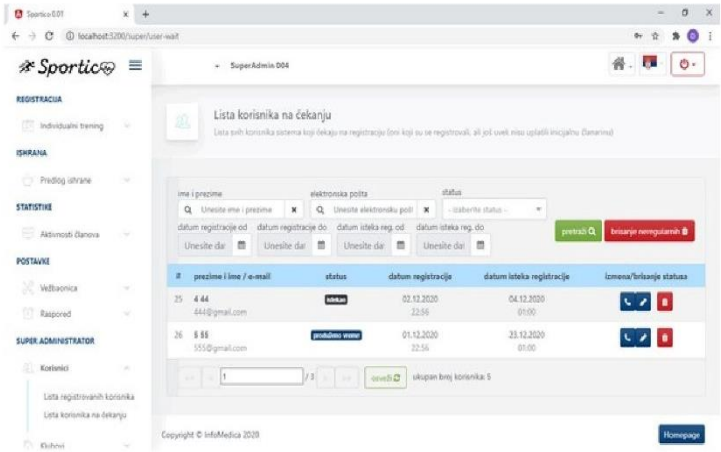
Polazna osnova rada je jednostavna aplikacija za registraciju članova sportskih društava. Za ovakvu aplikaciju, formirana je dokumentacija, kod koje je svaki deo formiran u odgovarajućem alatu. Potom je kreirana specifikacija značajno unapređene aplikacije koja ima mnogo strožije korisničke zahteve, ali i veće zahteve u pogledu sigurnosti. Uočeno je da bi dokumentacija pisana na isti način kao ranije bila glomazna, nepregledna i neefikasna za kontrolu razvoja sistema. Za ovako složenu aplikaciju postavljeni su obimniji tehnički zahtevi za pisanje dokumentacije, kako za klijentsku, tako i za serversku stranu. Dodatno su postavljeni i zahtevi za kreiranje dokumentacije za razvojni tim programera.

V. REALIZACIJA

Za potrebe praktičnog prikaza rezultata ovog rada, zakupljen je VPS (Virtual Private Server). Na njemu je podignut operativni sistem Linux Ubuntu 18.04.3. i postavljena MySql baza podataka verz. 5.7. Podignut je Apache Webserver version 2.4.29. Instalirana je kompletna podrška za GoLang.

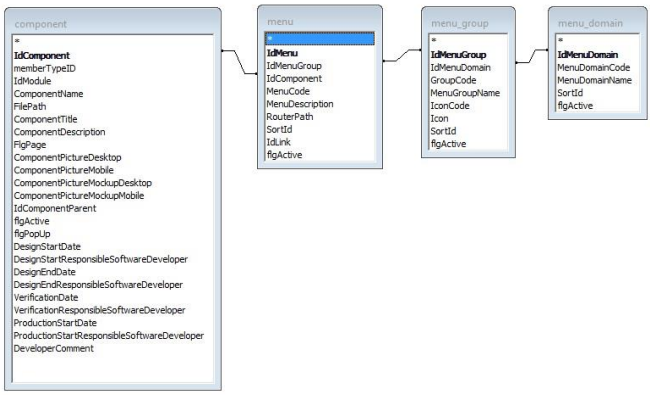
Unos podataka, njihovo ažuriranje, analiza i statistike obavljeni su kroz MS Access. Za kompleksniji unos podataka, napravljen je namenski korisnički interfejs. Preko ovog alata realizovana je i priprema najkompleksnijeg štampanog izveštaja, koji se odnosi na radni nalog programeru za izradu Angular komponente (slika 5.1) [21].

Module	**superadmin**
IdComponent	**94**
Naziv stranice	**Lista svih korisnika sistema koji čekaju na registraciju (oni koji su se registrovali, ali još uvek nisu uplatili inicijalnu članarinu)**
Component name	UserWaitComponent
File path	./modules/superadmin/user-wait/user-wait.component
Opis	
Stranica	☑
Slika desktop	



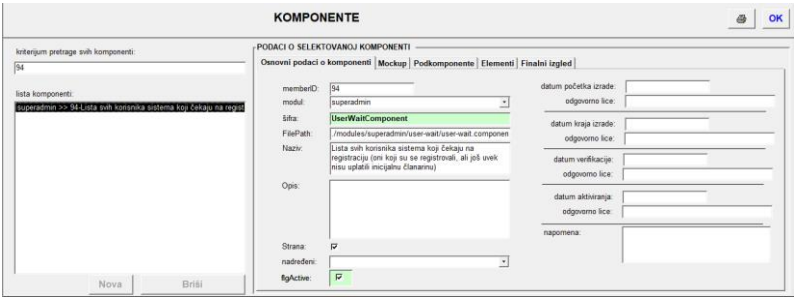
Slika 5.1. Prikaz dela štampanog izveštaja

Za potrebe prikaza i testiranja, formiran je sistem menija sa preko 110 stavki, napravljeno je oko 200 Angular komponenti, opisano oko 100 API funkcija (slika 5.2). Planirano je 9 različitih verzija iste aplikacije, pri čemu u svakoj postoji 7 različitih korisničkih rola [22].



Slika 5.2. Grafički prikaz relacija između entiteta koje opisuju meni aplikacije

Predviđeno je da se svi planirani ili realizovani elementi projekta unesu u jedinstvenu bazu podataka, po zadatoj specifikaciji koja je navedena u prethodnom poglavlju. Stoga je na osnovu ove specifikacije formirana MySQL baza podataka. Entiteti i veze su prilagođeni unosu projektne dokumentacije. Radi jednostavnijeg unosa podataka, napravljen je korisnički interfejs. Svi dokumenti o Angular komponentama, njihovim elementima i API funkcijama koje oni pozivaju uneti su u ovu bazu. Na osnovu unetih podataka, realizovan je štampani dokument sa detaljnom specifikacijom za Angular programere. Napravljen je i korisnički interfejs preko koga programer može dobiti specifikaciju on-line (slika 5.3).



Slika 5.3. Prikaz osnovnih podataka o komponenti

Na osnovu ovako organizovane dokumentacije, realizovana je test Angular

aplikacija koja u sebi sadrži sve komponente koje su navedene u dokumentaciji. Prilikom izrade komponenti, striktno su poštovane sve specifikacije iz dokumentacije.

Na strani servera, postavljena je aplikacija koja realizuje API zahteve korisničke strane [20]. Ona je pisana u programskom jeziku GoLang. Na osnovu relacija koje proizilaze iz dokumentacije, a koje su ugrađene u bazu podataka, realizovan je mehanizam provere korisničkih rola, prava pristupa i drugi sigurnosni mehanizmi.

VI. PREDNOSTI PRIKAZANOG REŠENJA U ODNOSU NA STANDARDNU DOKUMENTACIJU

Postoji mnoštvo specijalizovanih programa preko kojih je moguće formirati veoma kvalitetnu dokumentaciju. Neki od njih se baziraju na unosu teksta, a neki su više orijentisani ka grafičkoj ilustraciji radnih ekrana ili opisu programskih rešenja. U većini slučajeva, radi se o samostalnim aplikacijama, za koje uglavnom treba obezbediti licencu sa obavezom plaćanja. Budući da se radi o specijalizovanim programima, za izradu kvalitetne dokumentacije potrebno je upotrebiti nekoliko programa. Ako bismo želeli kreirati dokumentaciju koja detaljno opisuje razvoj Angular aplikacije, onda bi izbor raspoloživih alata bio veoma mali.

Predloženo rešenje bazira se na unosu svih dokumenata u bazu podataka. Podaci se unose direktno u obliku teksta, a neki delovi dokumentacije se dobijaju postavljanjem relacija između formiranih entiteta. U slučaju grafičkih ilustracija, mogu se koristiti specijalizovani programi, a finalni proizvod – grafika može se importovati u bazu i povezati sa odgovarajućim entitetom.

Na ovaj način, dobija se dokumentacija koja je veoma prilagođena potrebama programerske ekipe, pri čemu ostaje mogućnost da se za neke detalje mogu koristiti specijalizovani programi. Programeri koji dobijaju radne zadatke stoga ne moraju da koriste različite aplikacije i da uparuju podatke da bi dobili kompletnu informaciju za svoj deo posla.

A. Komparacija sa postojećim rešenjima

Na tržištu postoje mnogobrojna rešenja vezana za kreiranje dokumentacije o razvoju softverskih aplikacija. U ovom radu, korišćeni su neki od alata za generisanje dokumentacije i nakon dobijanja finalnih specifikacija, one su iz ovih alata importovane u bazu podataka. Tako na primer, Power Designer je poslužio za generisanje dijagrama slučaja korišćenja, dijagrama korisničkih

zahteva i dijagrama aktivnosti, na osnovu čega su dalje kreirane skice u programu Balsamiq Mockups. Skice su potom pretvorene u slike i kao takve prenete u bazu, u entitet "component". Na osnovu testiranja preko programa Postman, dobijeni rezultati su takođe kopirani u entitet "actions". Preko programa MS Access direktno su u bazu podataka unošene relacije između instanci entiteta, vezanih za obezbeđivanje prioriteta u programu.

Pored ovih alata, interesantno je pomenuti i neka programska rešenja koja su veoma popularna, a koja se odnose na kreiranje dokumentacije za razvoj aplikacija. Ovde su opisani programi Swagger i Visual Paradigm.

Swagger je programski paket namenjen dizajniranju API funkcija i generisanju dokumentacije [27]. Bazira se na OpenAPI Specification, koja je prvobitno imala ime Swagger Specification [28]. Ovaj stadard za specifikaciju omogućava opis, vizualizaciju i generisanje RESTful veb servisa. Sastoji se iz 3 modula: Swagger Editor, Swagger UI i Swagger Codegen.

Visual Paradigm je moćna platforma za upravljanje informacionim sistemima [29]. Iako pokriva praktično sve segmente projektovanja i ima mogućnost za generisanje dokumentacije, dobijenu dokumentaciju možemo relativno ograničeno povezati sa programskim jezikom (u našem slučaju sa Angularom). Na dijagramu komponenti možemo modelirati relacije između komponenti koje podržava programski jezik. U okviru grafičkog API dizajnera za REST moguće je veoma slikovito opisati svaku API funkciju, a potom sa grafičkog interfejsa direktno generisati API definiciju po Swagger 2 standardu. Na osnovu toga se dalje može automatski generisati polazni kod kako za serversku, tako i za klijentsku stranu.

VII. ZAKLJUČAK

U radu je prikazano rešenje preko koga se dokumentacija o razvoju Angular aplikacije vodi u bazi podataka. Tabele ove baze podataka čuvaju se na aplikativnom serveru i ravnopravne su sa svim ostalim tabelama.

U odnosu na klasični način vođenja dokumentacije, pokazuje se da je kontrola podataka mnogo bolja, a u izradi specifikacije može da učestvuje više saradnika istovremeno. Ovakva dokukmentacija je čitljivija i dostupnija programerskoj ekipi. Zahvaljujući opštim mehanizmima pretraživanja baze podataka, rukovodilac projekta ima veće mogućnosti za organizaciju projekta.

Sa druge strane, detaljno popunjavanje svih specifikacija komponenti i API poziva, kao i uspostavljanje svih veza u bazi podataka verovatno da nije prihvatljivo rešenje u slučaju da aplikacija ima relativno mali broj radnih ekrana, ako je jednostavna ili ukoliko nema potrebe za velikom bezbednošću

tokom rada. Ukoliko na razvoju ne učestvuje programerski tim, tada temeljno praćenje radnih zadataka programera i ocena njihovog učinka nema mnogo smisla.

Formiranje dokumentacije preko baze podataka ima prednosti i u fazi razvoja aplikacije. Rukovodilac projekta ima mogućnost za praćenje opterećenosti svakog programera, jednostavno može ispravljati dokumentaciju ili korigovati prioritete unutar celog projekta.

Prednosti se uočavaju i u fazi eksploatacije aplikacije. Moguće je formirati statistiku posećenosti strana i učestalost korišćenja određenih delova programa. Takođe, moguće je dobijati preciznu statistiku nastanka grešaka i upotrebiti mehanizam za njihovo lociranje. Na osnovu broja registrovanih grešaka i broja posećenih strana, moguće je napraviti ocenu učinka odgovornih programera.

LITERATURA

[1] https://en.wikipedia.org/wiki/TypeScript
[2] https://angular.io/guide/architecture
[3] https://github.com/angular/angular-cli/wiki
[4] https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)
[5] https://ng-bootstrap.github.io
[6] https://en.wikipedia.org/wiki/Base64
[7] https://cloud.google.com/blog/products/gcp/12-best-practices-for-user-account
[8] https://en.wikipedia.org/wiki/PBKDF2
[9] https://swoopnow.com/token-based-authentication/
[10] https://scotch.io/bar-talk/the-ins-and-outs-of-token-based-authentication
[11] https://scotch.io/tutorials/the-anatomy-of-a-json-web-token
[12] https://www.sohamkamani.com/blog/golang/2019-01-01-jwt-authentication/
[13] https://en.wikipedia.org/wiki/JSON
[14] https://www.callicoder.com/categories/golang/
[15] https://yourbasic.org/golang/create-error/
[16] https://blog.golang.org/godoc-documenting-go-code
[17] https://jsonapi.org/format/#error-objects
[18] https://bluxte.net/musings/2018/04/10/go-good-bad-ugly/#go-is-easy-to-learn
[19] https://www.interserver.net/tips/kb/types-http-request-methods/
[20] https://restfulapi.net

[21] https://e-nastava.vipos.edu.rs/pluginfile.php/2574/mod_resource/content/1/2010PrakBazeSkracene.pdf
[22] Natarajan Krishnan, 2005, Method And System For Developing Large Web-Based Multi-Language Applications, https://patentimages.storage.googleapis.com/b0/c0/5f/4787b58c3a5d3d/US20050204332A1.pdf
[23] Jobst Hoerentrup, 2014 Method For Generating Multi-Language menus, https://patentimages.storage.googleapis.com/29/28/c4/1523e0ad3cbdea/US8701004.pdf
[24] M. Jones, B. Campbell, C. Mortimore, 2015 JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants, https://www.hjp.at/doc/rfc/rfc7523.html
[25] Putu Arie Pratama, 2018 Token-based Single Sign-on with JWT as Information System Dashboard for Government, https://pdfs.semanticscholar.org/e178/debb761ff3114a4abe9f5232b1a40049bdef.pdf
[26] Kate M. Ferriter, Robert B. Mathis ,1989, Automated Interfacing Of Design/Engineering Software With Project Management Software, https://patentimages.storage.googleapis.com/f7/87/aa/14ea7f34c88d7f/US4875162.pdf
[27] https://swagger.io
[28] https://en.wikipedia.org/wiki/OpenAPI_Specification
[29] https://www.visual-paradigm.com

APSTRAKT

The main objectives of this paper are to form a database containing complete documentation for the development team of angular developers and integrate this database into the application database, with the creation of a mechanism for automatic priority checking, using JWT tokens. The initial goal is to design a database model in which to store documentation on how to connect components, based on the Angular development tool, during the design and development of a web application. In the purpose-created database, a set of tables was formed that would describe all the components in the Angular application, as well as API functions that are accessed on the server side. Finally, the advantages of this way of document management, which is characteristic of the design phase, are presented. In addition to this basic

advantage, good effects have been shown in the phases of program development and exploitation.

Keywords: Angular components, REST API, specification, documentation, JWT, database.

System for Creating Specifications and Controlling Development of Angular Components

Petar Erić