UNIVERZITET U BEOGRADU

MATEMATIČKI FAKULTET

Elektronske lekcije o platformi GitHub

MASTER RAD

Student

Aleksandar Davidović

Mentor

prof. dr Miroslav Marić

Septembar, 2023.

Mentor: prof. dr Miroslav Marić, redovni profesor

Predsednik komisije: dr Nina Radojičić Matić, docent

Član komisije: dr Sana Stojanović Đurđević, docent

Mojoj porodici, za podršku koja nikada nije nedostajala...

Jeli, koja me je ubedila da master privedem kraju...

Sadržaj

Uvod

Kada se govori o komunikaciji, poslovanju, plaćanju računa ili pak o zabavi prva asocijacija na sve najbrojano sve češće je internet. Sa globalnom pandemijom COVID-a 19, javlja se još veća potreba za učenjem putem interneta, što predstavlja najefikasniji (a verovatno i jedini) vid zamene „klasičnom učenju". Platforme za učenje putem interneta su sve popularnije i zbog uštede resursa, brzine razmene informacija i slično. Što se tiče učenja veb tehnologija, jedna od platformi za učenje, napisana na srpskom jeziku, je eŠkola veba i nalazi se na adresi http://www.edusoft.matf.bg.ac.rs/eskola_veba/#/home. U ovom radu će biti opisane lekcije koje su posvećene korišćenju platforme GitHub.

GitHub je platforma koja pomaže programerima da kreiraju, skladište svoj kôd i upravljaju njime, ali takođe predstavlja mesto gde svaki programer može da podeli svoj izvorni kôd. Kako je koncept otvorenog koda sve zastupljeniji, tako i ova platforma sve više dobija na značaju i popularnosti. GitHub ima preko 83 miliona registrovanih korisnika i predstavlja jednu od najpopularnijih platformi za kreiranje, isporuku i održavanje softvera. Značaj ove platforme ogleda se u tome što je koriste neke od najvećih tehnoloških kompanija u svetu, kao što su Google, Twitter, Mozilla, Facebook, IBM, Netflix i PayPal.

Platforma GitHub je vlasništvu kompanije Microsoft. To je u početku prihvaćeno sa rezervom od strane zajednice programera koji koriste koncept otvorenog koda, što se pokazalo pogrešnim, jer osnovni koncept i ideja platforme nisu narušeni. Osnovni paket na GitHub-u je besplatan za pojedince i timove, ali kompanije koje stvaraju vlasnički softver najčešće biraju neki od premijum paketa, kako kôd ne bi bio dostupan svima.

Kroz ovaj rad biće prikazane lekcije koje će čitaoca upoznati sa mogućnostima koje platforma GitHub pruža. Elektronske lekcije namenjene su svima koji se prvi put susreću sa ovim okruženjem, ali i onima koji žele da steknu uvid u dodatne mogućnosti ove platforme.

1 O Git-u

Za razumevanje načina funkcionisanja platforme GitHub potrebno je razumeti na koji način funkcioniše alat Git. Od njegovog nastanka 1991. godine Linux jezgro predstavlja prilično širok softverski projekat otvorenog koda. Tokom većine vremena održavanja Linux-ovog jezgra, softverske promene slate su kao zakrpe i arhivirani fajlovi. Projekat Linux jezgra je od 2002. godine počeo je da koristi vlasnički DVCS (eng. *Distribution version control system*), koji se zvao BitKeeper.

Partnerstvo između organizacije koja je radila na Linux jezgru i komercijalne kompanije koja je razvijala BitKeeper je prekinuto 2005. godine, pa je status ovog alata kao besplatnog bio ukinut. Iz ovog razloga je Linux-ova zajednica programera bila primorana da osmisli sopstveni alat, oslanjajući se na neke stvari koje su prethodno bili naučili prilikom korišćenja BitKeeper-a. Ono što je bilo bitno podrazumevalo je:

- brzinu,
- jednostavan dizajn,
- snažnu podršku za nelinearni razvoj (na hiljade paralelnih grana),
- potpuno distribuiran koncept,
- mogućnost da se efikasno rukuje velikim projektima kao što je Linux jezgro (brzina i veličina podataka).

Od svog nastanka 2005. godine, Git je evoluirao u alat koji je bio jednostavan za upotrebu, ali je sadržao navedene kvalitete.

Najznačanija razlika između programa Git i ostalih VCS (eng. *Version control system*) sistema (CVS, Subversion, Perforce, Bazaar itd.) je sam način na koji Git posmatra podatke. Većina drugih sistema čuva podatke kao listu promena fajlova. Ovi sistemi posmatraju podatke koje čuvaju kao skup fajlova i promena koje su napravljene nad njima tokom vremena (ovo se obično opisuje kao kontrola verzije bazirana na deltama). Program Git ne posmatra podatke na taj način, niti ih tako pamti. Umesto toga, program Git posmatra podatke kao da su skup snimaka (eng. *snapshots*) minijaturnog fajl sistema. Svaki put kada se izvrši komit (eng. *Commit*), ili kada se sačuva stanje novog projekta u Git repozitorijumu, on uzima sliku stanja fajlova u tom trenutku i pamti referencu na taj snimak. Da bi se održala efikasnost, ako se fajl nije promenio, Git ne čuva fajl ponovo, već samo vezu ka prethodnom identičnom fajlu koji je već ranije sačuvao. Program Git podatke posmatra kao tok snimaka [1].

Zbog ovih razlika, Git skoro svaki aspekt kontrole verzije posmatra na drugačiji način u odnosu na većinu ostalih sistema. Git predstavlja minijaturni fajl sistem sa ugrađenim izuzetno moćnim alatima, a ne samo običan VCS sistem.

2 Istorija platforme GitHub-a

Rad na razvoju GitHub platforme je započet oktobra 2007. godine. Platforma je 10. aprila 2008. godine pokrenuta od strane grupe programera, koju su činili: Tom Preston-Verner, Kris Vanstrat, Pi Džej Hajet i Skot Džekon. Prethodno je platforma pokrenuta kao beta izdanje. GitHub ima svoju godišnju konferenciju koja se zove *GitHub Universe*. GitHub je u početku funkcionisao kao organizacija ravnopravnih menadžera, međutim sam način plaćanja bio u nadležnosti izvršnog direktora Toma Preston-Vernera, koji je posle niza skandala 2014. godine podneo ostavku.

Februara 2009. godine GitHub je objavio da je u prvih godinu dana postojanja na platformi akumulirano 46 000 javnih skladišta, od kojih je 17 000 bilo iz prethodnog meseca. U to vreme je 6 200 skladišta račvano bar jednom, a 4 200 je spojeno. Tada je platformu koristilo nešto više od 100 000 korisnika, sa 90 000 javnih skladišta, od kojih je 12 000 račvano bar jednom [4]. Već u 2010. godini GitHub je imao preko milion skladišta, a onda je u narednoj godini broj skladišta udvostručen. Do početka 2013. godine GitHub je imao više od tri miliona korisnika sa preko pet miliona skladišta. Krajem iste godine broj skladišta je bio dupliran na 10 miliona. U junu 2022. godine GitHub je imao preko 83 miliona korsnika.

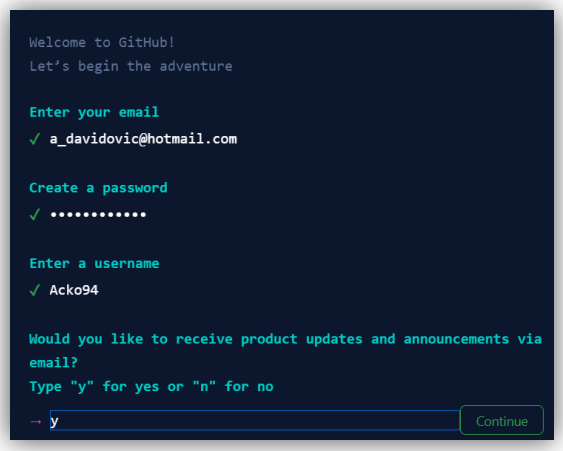
Sedište GitHub-a je u San Francisku. Prva kancelarija GitHub-a koja je otvorena van Sjedinjenih američkih država, bila je u Japanu 2015. godine. Na Forbsovoj *Cloud 100* listi za 2016. godinu, na kojoj se svake godine nalazi 100 najboljih kompanija koje koriste računarstvo u oblaku, GitHub platforma zauzimala je 14. mesto. Više informacija o istoriji platforme GitHub može se pronaći u [3].

3 Otvaranje naloga i početna podešavanja

U ovom poglavlju biće prikazano otvaranje naloga na platformi GitHub i opcije personalizacije. Takođe, biće prikazana početna podešavanja profila, korišćenje mejl adresa i aktiviranje dvofaktorske autentifikacije (eng. *Two-factor Authentication*).

3.1 Otvaranje naloga

Pre samog korišćenja platforme GitHub, budući korisnik treba da otvori korisnički nalog. To se uradi tako što se poseti sajt https://github.com, unese se mejl adresa i lozinka, izabere se slobodno korisničko ime, a zatim se klikne na dugme nastavi (eng. *Continue*). Dodatno, korisnik unosom slova y (eng. *Yes*) ili n (eng. *No*) bira da li želi da dobija dodatna obaveštenja, kao što se može videti na slici 1.

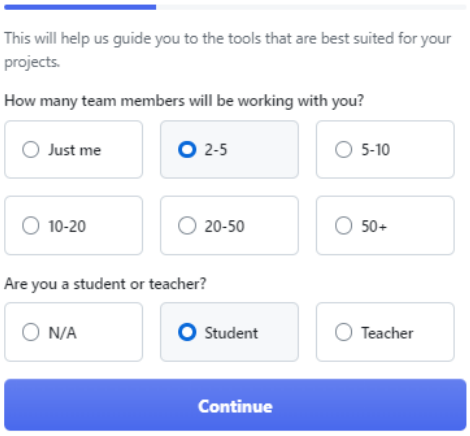


Slika 1: Unos mejl adrese, lozinke i korisničkog imena

GitHub će novom korisniku poslati mejl u kojem se nalazi kôd, čijim unosom korisnik treba da potvrdi mejl adresu koja je uneta prilikom registracije. Ovo treba uraditi odmah.

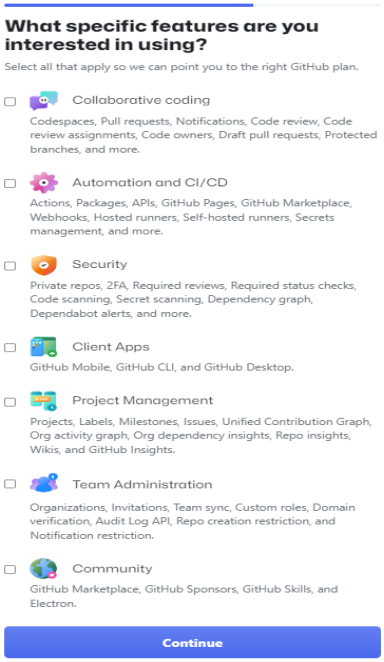
Slika 2: Unos koda za potvrdu mejla

Opcije personalizacije na početku aktiviranja naloga nisu obavezne, ali ih GitHub nudi. Na slici 3 se može videti opcija koja prikazuje izbor planiranog broja ljudi sa kojim će korisnik da sarađuje, da li je korisnik student ili nastavnik.



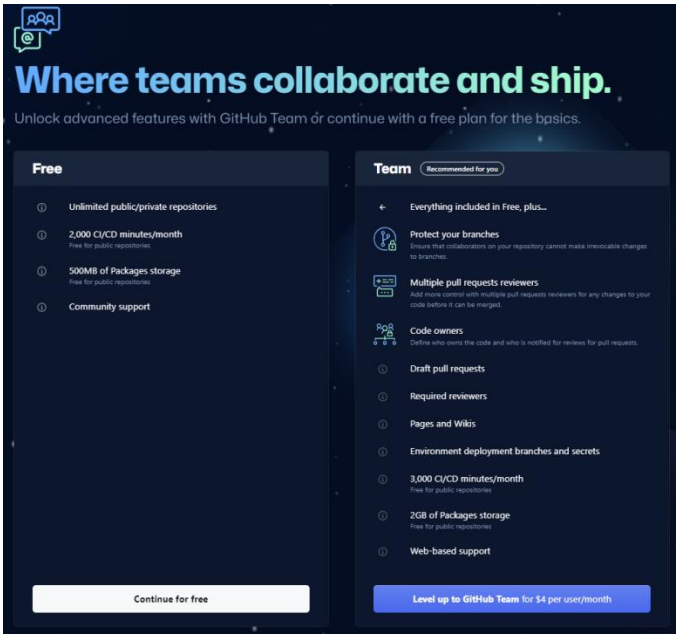
Slika 3: Osnovni podaci o korisniku

Na slici 4 prikazani su razlozi korišćenja platoforme, koje korisnik navodi prilikom registracije. Neke od mogućnosti koje se nude su zajedničko kodiranje, klijentske aplikacije, administracija timovima itd.



Slika 4: Razlozi korišćenja platforme

Naredni korak predstavlja mogućnost da se korisnik opredeli između besplatnog naloga i naloga koji se plaća. Sve opcije koje nudi besplatan nalog, nudi i nalog koji se plaća. Najvažnije razlike odnose se na to da nalog koji se plaća korisniku dodeljuje veće skladište (besplatan nalog dobija 500MB, a nalog koji se plaća dobija 2GB). Dodatno, nalog koji se plaća ima podršku putem interneta (eng. *Web-based support*). Razlike u nalozima mogu se pronaći na adresi *https://github.com/pricing.*



Slika 5: Izbor besplatnog naloga ili naloga koji se plaća

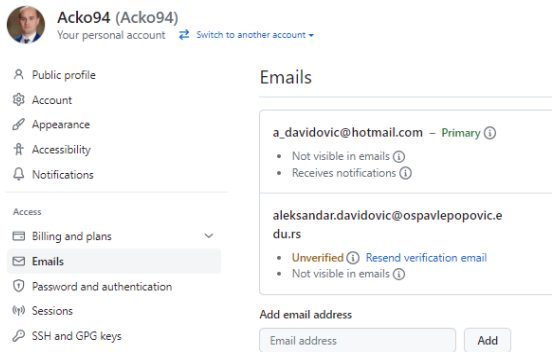
3.2 Početna podešavanja naloga

Na početku platforma GitHub svakom novom korisniku dodeljuje avatar. Dodeljeni avatar se može zameniti slikom koju korisnik izabere. Najpre treba izabrati opciju profila (eng. *Your profile*) i onda treba izabrati opciju da se postavi nova slika (eng. *Upload new picture*). Nakon izbora slike, kadgod se vrši interakcija na platformi, drugi korisnici platforme pored korisničkog imena vide i sliku korisnika.

3.2.1 Mejl adrese korisnika

Način na koji GitHub mapira Git komitove na korisnički nalog je pomoću mejl adrese. Ako korisnik u svojim komitovima koristi više mejl adresa i želi da ih GitHub poveže kako treba, neophodno je dodati sve adrese koje su prethodno korišćene u mejl odeljku (eng. *Emails*) pristupnog odeljka (eng. *Access*). U dodavanju mejl adresa mogu se videti neka od mogućih stanja. Adresa na vrhu je potvrđena i postavljena kao primarna adresa, što znači da će korisnik na

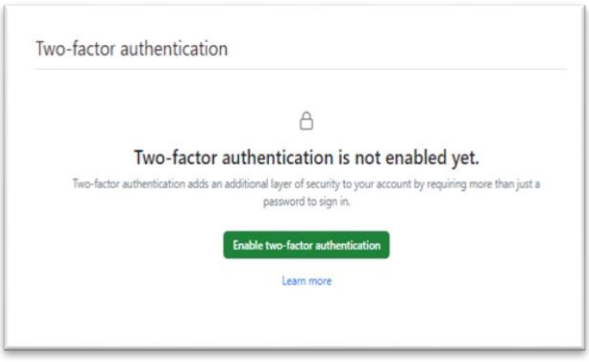
nju dobijati sva obaveštenja i račune. Ako je druga adresa potvrđena, može se koristiti kao primarna ako korisnik želi da je promeni. Međutim, ako druga adresa nije potvrđena, to znači da se ne može postaviti kao primarna adresa (videti sliku 6). Ako GitHub vidi bilo koju od ovih adresa u komit porukama u bilo kom repozitorijumu na sajtu, biće povezana sa registrovanim korisničkim nalogom [1].



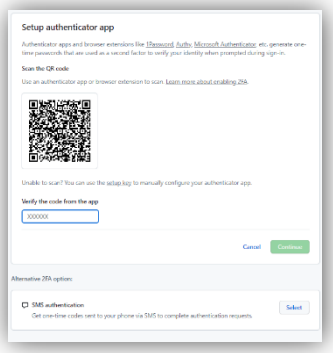
Slika 6: Potvrđena i nepotvrđena mejl adresa

3.2.2 Dvofaktorska autentifikacija

Da bi se poboljšala sigurnost naloga, korisnik ima opciju aktiviranja dvofaktorske autentifikacije (eng. *Two-factor authentication).* Podešavanje dvofaktorske autentifikacije se nalazi u kartici pristupa (eng. *Access*) u okviru podešavanja naloga (eng. *Account settings*). Ako se klikne na dugme koje omogućava dvostruku autentifikaciju (eng. *Enable two-factor authentication*), dolazi se do stranice gde se može izabrati da se na telefonu koristi aplikacija za generisanje sekundarnog koda. Dodatno, platforma GitHub pruža mogućnost da šalje kôd preko SMS poruke svaki put kada korisnik treba da se prijavi na platformu. Opcije uključivanja i aktivacije dvofaktorske autentifikacije mogu se videti na slikama 7 i 8.



Slika 7: Uključivanje 2FA autentifikacije



Slika 8: Skeniranje QR koda i unos

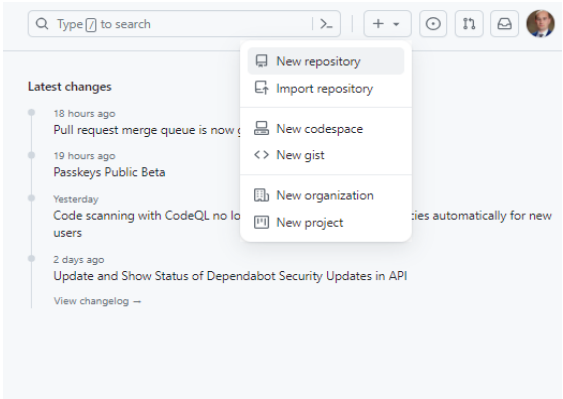
4 Kreiranje i upravljanje repozitorijumima

Da bi korisnik uspešno koristio platformu GitHub potrebno je da razume pojmove poput kreiranja repozitorijuma, njihovog povezivanja na granu i upravljanja zahtevima za povlačenje.

4.1 Kreiranje repozitorijuma

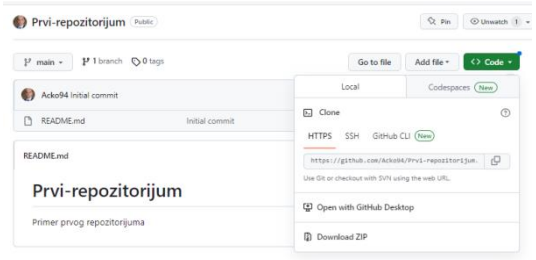
Kreiranje repozitorijuma na GitHub platformi je veoma jednostavno. Prvo se u gornjem desnom uglu klikne na „+" i izabere se opcija novi repozitorijum (eng. *New repository*), kao što se može videti na slici 9. Ukoliko se neki fajl nalazi na računaru, a korisnik želi da ga postavi na platformu, ide se na opciju uvoza repozitorijuma (eng. *Import repository*). Nakon što se pojavi opcija za imenovanje repozitorijuma, korisnik ima mogućnost da repozitorijum označi kao javni (eng. *Public*) ili kao privatni (eng. *Private*).

Opcija javnog repozitorijuma omogućava da svaki korisnik na GitHub platformi može da vidi repozitorijum, dok sa druge strane opcija privatnog repozitorijuma omogućava da se repozitorijum deli samo sa određenom grupom korisnika. Međutim, nezavisno od izabrane opcije, drugi korisnici neće moći da vrše izmene nad tim fajlovima bez odobrenja. Dodatna korisna opcija je fajl sa uputstvom (eng. *Readme file*), u kojem se mogu pisati različite instrukcije i uputstva.

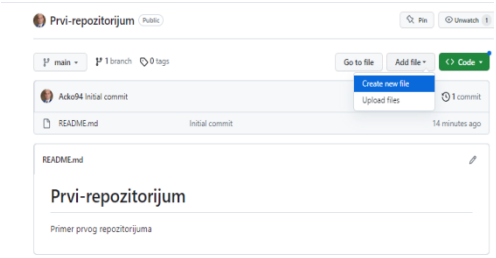


Slika 9: Kreiranje repozitorijuma

Kada je repozitorijum kreiran, otvara se nova sekcija unutar koje postoji niz različitih opcija. Za početak su najvažnije opcije koje se dobijaju klikom na dugme kôd (eng. *Code*). Unutar nje se može kopirati link do repozitorijuma korisnika, koji se kasnije može slati drugima, odnosno može se skinuti i ZIP fajl. Opcije koje se dobijaju klikom na dugme kojim se dodaje fajl (eng. *Add File*) ima dve mogućnosti. Prva je da kreira novi fajl, a druga mogućnost odnosi se na dodavanje fajlova sa računara korisnika. Kada se novi fajl kreira, otvaraju se dve mogućnosti i to da se on doda trenutnoj grani (prva opcija) ili da se kreira nova grana za ovaj komit i podnese zahtev za povlačenje (druga opcija).

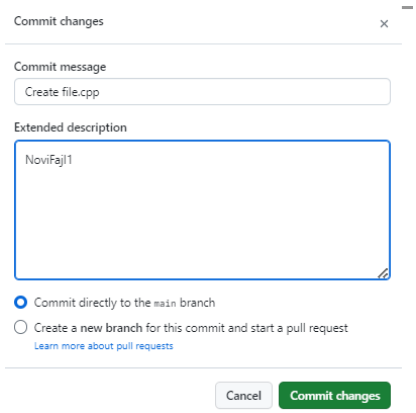


Slika10: Opcija Code



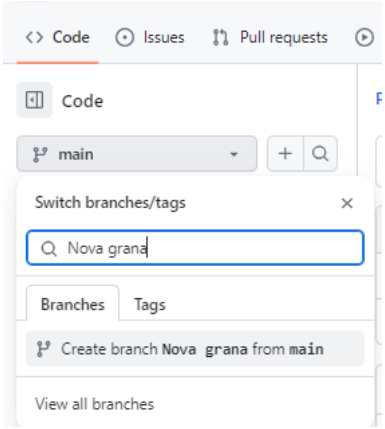
Slika 11: Opcija File

Postoji i mogućnost da se navedu neke karakteristike fajla koji je kreiran, pomoću opcije dodatnog opisa (eng. *Extended description*), kao što je prikazano na slici 12.



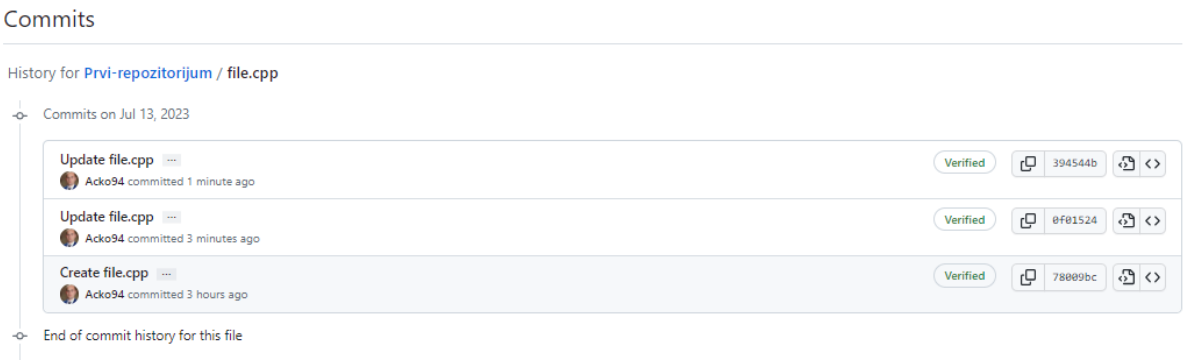
Slika 12: Kreiranje novog repozitorijuma i povezivanje na granu

Kada se kreira, fajl je deo ove main grane. Međutim, tu postoji opcija da se postojeći fajlovi dodaju nekoj novoj grani, pri čemu se nova grana kreira iz main-a. Ukoliko se kasnije izvrši određena promena u novoj grani, to neće uticati na main granu, iz razloga što main grana predstavlja raniju verziju.



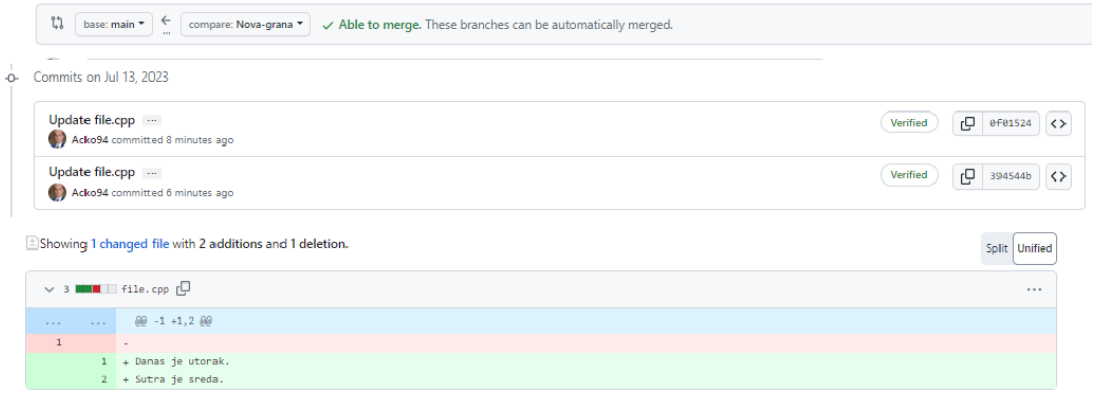
Slika 13: Kreiranje nove grane

Sve promene koje su izvršene u nekom fajlu, kao i korisnici koji su te promene izvršili, mogu se videti ako se izabere opcija istorije (eng. *History*), gde se otvara prozor prikazan na slici 14.



Slika 14: Sve izmene jednog fajla unutar repozitorijuma

Ukoliko se nešto unutar repozitorijuma obriše, pojavljuje se znak „–" i crvena boja, a prilikom dodavanja nečega pojavljuje se znak „+" i zelena boja. Generalno, najbolji pregled svih izmena unutar jednog repozitorijuma je preko opcije upoređivanja i povlačenja zahteva (eng. *Compare and pull request*), koja se nalazi u okviru sekcije koda.



Slika 15: Izmene unutar repozitorijuma

Kada se nakon izmena u repozitorijumu pojavi čekirana opcija da je spajanje moguće (eng. *Able to merge*), to znači da se sve promene koje su izvršene u drugoj grani mogu primeniti i u main grani. Ukoliko korisnik ima veliki broj grana postoji mogućnost da izabere iz koje grane će se promene primenjivati i na samu main granu. To se postiže izborom opcije kreiranja zahteva za povlačenje (eng. *Create pull request*), koju je kasnije potrebno potvrditi. Na kraju će sve biti spojeno sa granom main.

Dodatno, postoji opcija u kojoj korisnik može da ukaže na potencijalne probleme koji mogu da nastanu (eng. *Issues*).

4.2 Upravljanje zahtevima za povlačenje

Kada korisnik kreira neki projekat sa kodom, može da dobije zahtev za povlačenje. Ukoliko je korisnik dodao još nekoliko saradnika koji imaju mogućnosti da nešto menjaju, dolazi se do različitih situacija kada stignu zahtevi za povlačenje. Oni mogu da dođu ili sa grane u račvi korisničkog repozitorijuma ili sa druge grane iz istog repozitorijuma. Jedina razlika je u tome što oni iz račve obično pripadaju drugim saradnicima, gde korisnik ne može da postavi na njihovu granu i oni ne mogu na korisničku, dok kod internih zahteva za povlačenje u opštem slučaju i korisnik i saradnici imaju pristup grani.

Kada neko napravi promene u kodu korisnika i pošalje zahtev za povlačenje, korisnik dobija mejl koji ga obaveštava o novom zahtevu za povlačenje. U mejlu se nalaze link ka zahtevu za povlačenje na platformi GitHub i nekoliko URL adresa koje se mogu koristiti iz komandne linije. Autor koda može da razgovara sa korisnikom koji je otvorio zahtev za povlačenje. Svaki put kada neko prokomentariše zahtev za povlačenje, autor koda dobija mejl obaveštenja i tako je obavešten o promenama koje se dešavaju. Svako od učesnika razgovora dobiće link ka zahtevu za povlačenje gde se određena aktivnost dešava.

Kada je kôd sređen i autor želi da ga spoji, može da povuče kôd i spoji ga lokalno ili to može da uradi pomoću komandne linije i komande *git pull <url> <branch>*. Dodatno, grana može da se doda kao udaljeni repozitorijum koji se preuzme i tek onda spaja. Ako je spajanje trivijalno, može se uraditi samo klikom na dugme spajanja (eng. *Merge*) na platformi GitHub.

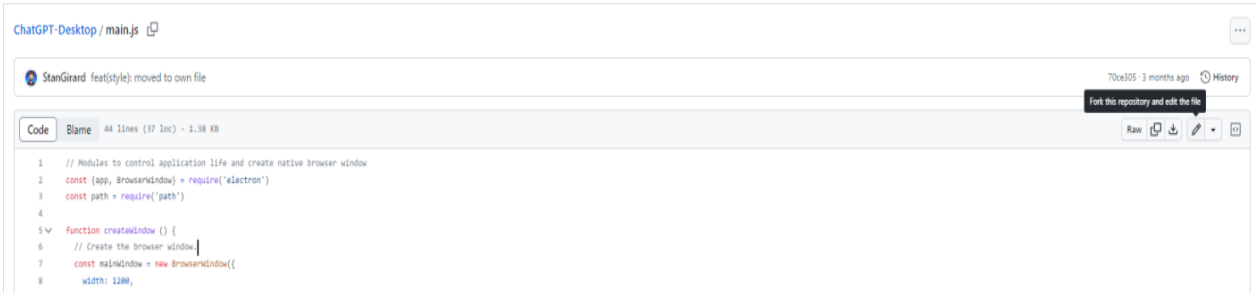
Ukoliko autor ne želi da spoji promene, može jednostavno da zatvori zahtev za povlačenje i korisnik koji je poslao zahtev za povlačenje biće obavešten da njegovo spajanje nije uspelo. Korisnik ne mora kao zahteve za povlačenje otvarati one koji kao odredište imaju glavnu ili master granu, već može da se otvori zahtev za povlačenje koji kao odredište ima bilo koju granu na mreži. Dodatno, odredište može biti i drugi zahtev za povlačenje na mreži [1].

5 Rad na postojećim projektima

U ovom poglavlju će biti opisano račvanje projekata, kao i sam GitHub proces rada. Osim osnovnih zahteva za povlačenje, biće prikazani i neki napredniji zahtevi za povlačenje i reference.

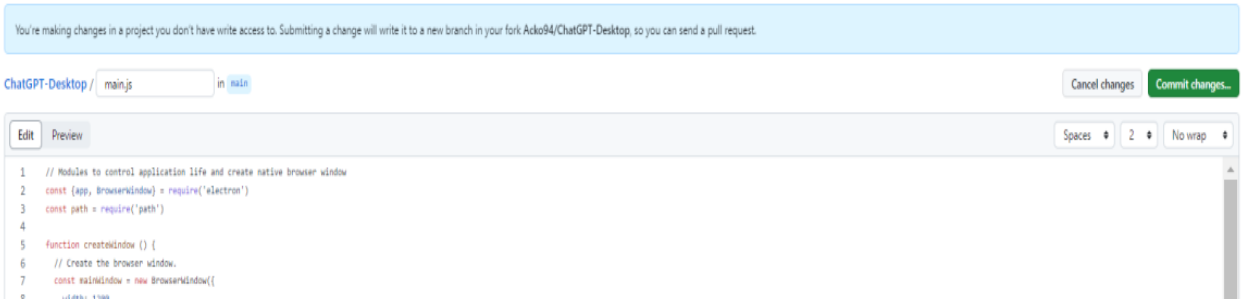
5.1 Račvanje projekata

Kada korisnik želi da da doprinos postojećem projektu za koji nema dozvolu da ga menja, uvek ima opciju „račvanja" (eng. *fork*). GitHub će napraviti kopiju projekta koja se dodeljuje korisniku i projekat je onda moguće menjati. Gledano sa ove vremenske distance, termin račvanja je imao negativno značenje jer je opisivao situaciju kada neko projekat otvorenog koda iskoristi stvarajući konkurentski projekat. Na platformi GitHub račvanje podrazumeva isti projekat, ali u korisničkom prostoru imena, što korisniku dozvoljava da javno pravi promene projekta, pa to u suštini predstavlja još otvoreniji i transparentniji doprinos.



Slika 16: Opcija račvanja

Na ovaj način kreatori projekata ne moraju da brinu o dodavanju korisnika kao saradnika da bi im dali dozvolu za menjanje. Korisnici mogu da račvaju projekat, postavljaju na račvu i daju svoj doprinos promenama nazad originalnom repozitorijumu tako što će kreirati zahtev za povlačenjem (eng. *Pull Request*). Da bi korisnik račvao projekat, treba da poseti stranicu projekta klikom na dugme račvanja. Nakon nekoliko sekundi, biće prebačen na stranicu svog novog projekta, sa svojom sopstvenom kopijom koda, koju dalje može da menja.



Slika 17: Primer račvanja, novi projekat korisnika sa sopstvenom kompijom koda

5.2 GitHub proces rada

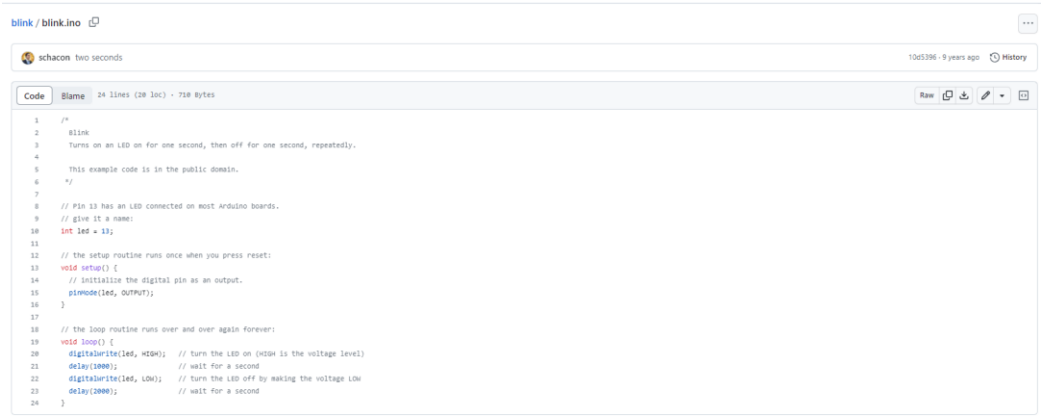
GitHub je dizajniran prema određenim procesima za saradnju, ali je pre svega fokusiran na zahteve za povlačenjem. Ovaj način rada funkcioniše bilo da je u pitanju saradnja sa dobro uvezanim timom na jednom deljenom repozitorijumu, ili sa globalno distribuiranom kompanijom ili mrežom nepoznatih korisnika koji daju doprinos projektu preko brojnih račvi. Bazira se na tematskim granama.

U opštem slučaju radi na sledeći način.

1. Račva se projekat.
2. Napravi se tematska grana iz mastera.
3. Naprave se neki komitovi da se poboljša projekat.
4. Postavi se nova grana na korisnički GitHub projekat.
5. Otvara se zahtev za povlačenjem na GitHub servisu.
6. Diskutuje se i eventulano nastavlja sa komitovanjem.
7. Vlasnik projekta spoji ili zatvori zahtev za povlačenjem.
8. Sinhronizuje se ažurirana master grana nazad u svoju račvu.

Na jednom jednostavnom primeru se može prikazati promena projekta otvorenog koda koji je hostovan na GitHub servisu, koristeći navedeni način rada. Takođe, sve ovo se može uraditi i pomoću zvaničnog GitHub CLI alata. Alat se može skinuti za bilo koji operativni sistem.

Primer. Korisnik traži kôd koji hoće da izvršava na svom Arduino programabilnom mikrokontroleru i našao je odličan fajl programa na servisu GitHub na adresi *https://github.com/schacon/blink*, kao što je prikazano na slici 18.



Slika 18: Kôd programa koji se popravlja

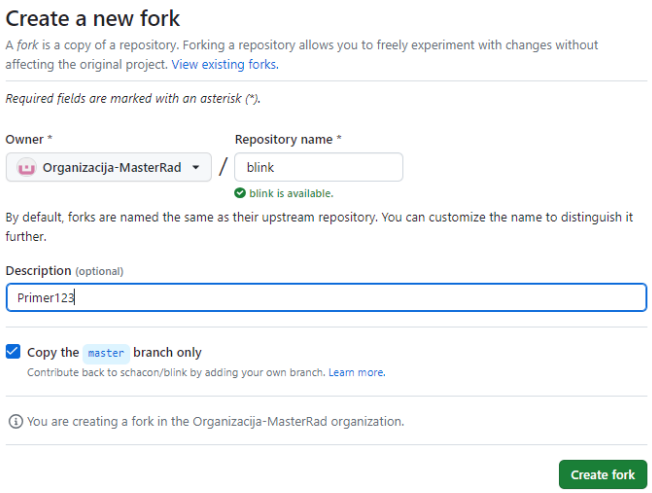
Jedini problem je što je brzina treptanja suviše velika. Korisnik smatra da bi bilo bolje da se kod svake promene stanja čeka tri sekunde umesto jedne. Zato korisnik hoće da unapredi program i pošalje ga nazad na projekat kao predloženu promenu.

Da bi korisnik dobio svoju kopiju projekta, najpre se klikne na dugme račvanja koje je ranije opisano. Ovde je korisničko ime „tonychacon” tako da je korisnička kopija projekta na adresi *https://github.com/tonychacon/blink* i tu može da se uređuje. Kloniraće se projekat lokalno, napraviti tematska grana, promeniti kôd i konačno poslati te promene nazad na GitHub. To se postiže na sledeći način.

1. Klonira se račva projekta lokalno.
2. Napravi se opisna tematska grana.
3. Izmeni se kôd.
4. Proverava se da li je promena dobra.
5. Komituje se promena na tematsku granu.
6. Postavlja se nova tematska grana nazad na korisničku GitHub račvu.

Ako se sada korisnik vrati na svoju GitHub granu, vidi se da je GitHub primetio da je korisnik postavio novu tematsku granu i prikazuje se veliko zeleno dugme da se promene objave i otvara se zahtev za povlačenje ka prvobitnom projektu.

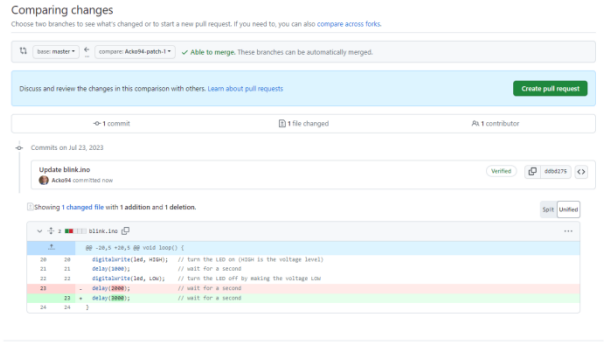
Drugi način je da korisnik ode na stranicu grananja (eng. *Branches*) *https://github.com/[korisnik]/[projekat]/branches* kako bi alocirao svoju granu i otvorio novi zahtev za povlačenje odatle.



Slika 19: Novi zahtev za povlačenje

Ako se klikne na prikazano zeleno dugme, pojaviće se ekran koji korisnika pita da njegovom zahtevu za povlačenje da naslov i opis. Ovo je veoma korisno, pošto korektan opis pomaže vlasniku prvobitnog projekta da shvati šta korisnik želi da uradi, da li su korisničke promene ispravne i da li će prihvatanje promena poboljšati prvobitni projekat.

Moguće je videti i listu komitova na korisničkoj tematskoj grani koji su „ispred” master grade i razlike svih promena koje će biti napravljene ako vlasnik projekta odluči da spoji ovu granu.



Slika 20: Prikaz izmena

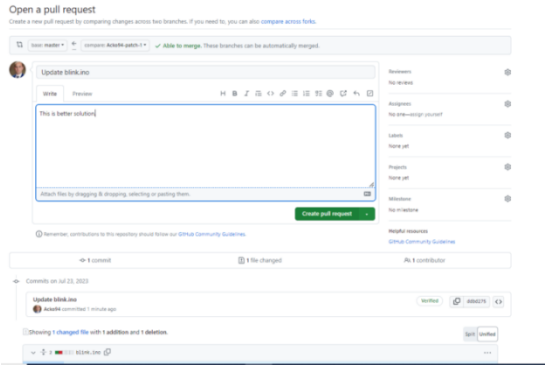
Kada na ekranu korisnik klikne na dugme kojim se kreira zahtev za povlačenje (eng. *Create pull request*), vlasnik projekta koji je korisnik račvao će dobiti obaveštenje da neko predlaže promenu i dobiće link do stranice koja ima sve informacije u vezi toga. Svi detalji u vezi sa GitHub procesom rada mogu se naći u [1].

5.3 Iteracije nad zahtevom za povlačenje

Nakon navedenog procesa, vlasnik projekta može da pogleda predloženu promenu i da je spoji, da je odbije ili da je komentariše. Ukoliko se vlasniku projekta dopada ideja iz primera, ali želi da svetlo bude isključeno malo duže, on može pisati komentar. Vlasnik projekta može da pregleda ujedinjenu razliku i ostavi komentar klikom na bilo koju liniju. Kada održavalac napravi ovaj komentar, korisnik koji je otvorio zahtev za povlačenjem (zapravo svako ko prati repozitorijum) dobiće obaveštenje. Kasnije će biti pokazan način promene ovih podešavanja, ali ukoliko je korisnik uključio mejl obaveštenja. Svako može da ostavi opšte komentare na zahtev za povlačenje.

Na stranici za diskusiju zahteva za povlačenje vidi se primer gde vlasnik projekta komentariše liniju koda, a onda ostavlja opšti komentar u odeljku za diskusiju. Može se primetiti da se komentari o kodu takođe prebacuju i u ovu konverazaciju.

Sada je korisnik u mogućnosti da vidi šta treba da se uradi da bi se promena prihvatila. Ovaj primer zahteva samo primenu jednostavnijih tehnika. Sa GitHub servisom može se jednostavno ponovo sve komitovati na tematskoj grani i onda se postavljaju promene koje će automatski ažurirati zahtev za povlačenje. U konačnom zahtevu za povlačenje takođe se može videti i to da je stari komentar na kodu sakriven u ažuriranom zahtevu za povlačenje, pošto je postavljen na liniji koja se od tada izmenila.



Slika 21: Pisanje komentara

5.4 Napredni zahtevi za povlačenje

Do sada su prikazane jednostavnije tehnike kojima se daje doprinos već postojećem projektu. Sada će biti prikazane neke efikasnije metode, koje korisnik može da koristi.

5.4.1 Zahtevi za povlačenje kao zakrpe

Važno je istaći da se u mnogim projektima ne gleda na zahteve za povlačenjem kao na niz savršenih zakrpa koje često treba da se primene tim redom, kao što većina projekata baziranih na mejling listi gleda na doprinose u nizu zakrpi. Većina GitHub projekata gleda na grane za koje postoji zahtev za povlačenje kao iterativne razgovore o predloženoj promeni, koji kulminiraju u jednistvenu razliku koja se na kraju prihvata spajanjem.

Ovo je veoma značajna razlika, zato što se u opštem slušaju promena predlaže pre nego što se smatra da je kôd savršen, što nije čest slučaj kod doprinosa baziranih na nizu zakrpa sa mejling liste. Ovo omogućava pravovremeni razgovor sa održavaocima tako da je dolazak do dobrog rešenja uglavnom plod timskog rada. Kada se kôd predloži zahtevom za povlačenje i autori ili zajednica predlože promenu, niz zakrpa se u opštem slučaju ne smotava ponovo već se razlika postavlja kao novi komit na grani, vodeći razgovor dalje tako da je kontekst prethodnog rada nedirnut.

Na primer, ako korisnik želi da se vrati nazad i opet pogleda konačni zahtev za povlačenje, primetiće se da saradnik nije rebazirao svoj komit i poslao još jedan zahtev za povlačenje. Umesto toga, dodao je nove komitove i postavio ih na već postojeću granu. Na ovaj način, ako se korisnik vrati nazad i pogleda zahtev za povlačenje, moći će lako da nađe kontekst zbog koga su takve odluke donete. Ako se na platformi klikne na dugme kojim se vrši spajanje (eng. *Merge*), pravi se komit spajanja koji ukazuje na zahtev za povlačenje tako da korisnik može da se vrati nazad i izvrši pretragu u originalnoj diskusiji, ako ima potrebe.

5.4.2 Kako održavati korak sa uzvodnim granama

Ako je korisnički zahtev za povlačenje zastareo ili se iz nekog drugog razloga spajanje ne odvija kako treba, korisnik treba da popravi greške, kako bi održavalac mogao to lako da spoji. GitHub će ovo testirati i poslati obaveštenje na dnu svakog zahteva za povlačenje o tome da li je spoj trivijalan ili ne.

Ako korisnik primeti da dobija obaveštenje poput *Zahtev za povlačenje se ne spaja glatko* (eng. *Pull Request does not merge cleanly*), treba da popravi svoju granu tako da opcija spajanja bude omogućena (sama grana postaje zelena) i da održavalac ne mora da radi dodatni posao. Ovo se može uraditi na dva osnovna načina.

- Može se rebazirati svoja grana na vrh odredišne (obično je to master grana repozitorijuma koji je korisnik račvao). Pod rebaziranjem se podrazumeva da korisnik može da preuzme sve promene koje su komitovane u jednoj grani i da ih ponovi u nekoj drugoj.
- Može da se spoji odredišna grana na korisničku granu.

Većina programera na platformi GitHub bira drugu opciju. Ono što je važno jesu prethodno opisani koraci i spajanje, tako da rebaziranje ne daje mnogo osim toga što je istorija nešto čistija, a sa druge strane je mnogo komplikovanije i podložnije greškama.

Ako korisnik želi da spoji odredišnu granu kako bi njegov zahtev za povlačenje mogao da se spoji, treba da se doda prvobitni repozitorijum kao novi udaljeni repozitorijum, da se preuzmu (eng. *fetch*) podaci sa njega, spoji se glavna grana tog repozitorijuma u svoju tematsku granu, reše svi eventualni problemi i na kraju se to postavi nazad na istu granu za koju je otvoren zahtev za povlačenje.

Jedna od najboljih stvari u vezi programa Git je to što neprestano mogu da se rade ovakve stvari. Ako postoji projekat koji veoma dugo traje, jednostavno se mogu spajati odredišne grane iznova i iznova, a rešavanje problema podrazumeva samo konflikte koji nastanu od trenutka kada je poslednji put nastalo spajanje, što proces čini podesnim za rukovanje. Ako korisnik želi da definitivno rebazira granu kako bi je obrisao, to se može uraditi, ali se strogo preporučuje da se ne forsira postavljanje preko grane na kojoj je zahtev za povlačenje već otvoren. Ako su je drugi korisnici povukli i radili nešto na njoj, dolazi se do gomile problema koji podrazumevaju opasnosti rebaziranja. Umesto toga, treba postaviti rebaziranu granu na novu granu na servisu GitHub i otvoriti potpuno novi zahtev za spajanjem koji ukazuje na stari, pa se onda zatvori prvobitni.

5.5 Reference

Postoji puno načina da se ukaže na ono što je korisnik napisao na platformi GitHub. Najpre treba pojasniti kako da se unakrsno ukaže na drugi zahtev za povlačenjem ili na tiket

(eng. *Ticket*). Svi zahtevi za povlačenjem i tiketi imaju svoje brojeve koji su jedinstveni za jedan projekat. Na primer, ne može se imati zahtev za povlačenje #3 i tiket #3. Ako korisnik želi da ukaže na zahtev za povlačenjem ili tiket sa bilo kog drugog, može jednostavno da stavi #<broj> u bilo kom komentaru ili opisu. Korisnik može da bude i određeniji ako tiket ili zahtev za povlačenjem već „živi" negde drugde. Napiše se korisničko-ime#<broj> ako se ukazuje na tiket ili zahtev za povlačenjem u račvi repozitorijuma u koje se korisnik nalazi, ili korisničko-ime/repozitorijum#<broj> da se ukaže na nešto iz nekog drugog repozitorijuma.

Ako se ponovo koristi prethodni primer gde je korisnik rebazirao granu, napravljen je novi zahtev za povlačenjem na nju, pa sada želi da se ukaže na stari zahtev za povlačenje iz novog. Takođe korisnik želi da pokaže na tiket u račvi repozitorijuma i tiketu u potpuno drugom projektu. Može da popuni opis baš onako kako je to učinjeno u unakrsnim referencama u zahtevu za povlačenjem. Kada se pošalje ovaj zahtev za povlačenje, može se videti da se sve to prikazuje kao prikazane unakrsne reference u zahtevu za povlačenje.

Može se primetiti da je pun GitHub URL koji je unet skraćen samo na neophodne informacije. Ako se sada korisnik iz primera vrati nazad i zatvori prvobitni zahtev za povlačenjem, to može da se vidi njegovim pominjanjem u novom, platforma GitHub automatski kreira događaj praćenja unazad (eng. *trackback event*) u vremenskoj liniji zahteva za povlačenjem. Ovo znači da će svako ko poseti ovaj zahtev za povlačenjem i vidi da je zatvoren lako moći da dođe do onog kojim je zamenjen.

Pored brojeva tiketa, na neke komitove se može ukazivati sa SHA-1 (eng. *Secure Hash Algorithm 1*). Ovo je kriptografska funkcija, koja se koristi za digitalni potpis. SHA-1 dobija se tako što se u okviru repozitorijuma izabere željeni fajl, uđe se u opciju pregleda istorije (eng. *History*) i onda se jednostavno dobija klikom na dugme kojim se kopira ceo SHA-1 (eng. *Copy the full SHA*). Prilikom ukazivanja mora da se navede kompletan SHA-1 sa 40 karaktera, ali ako servis GitHub to vidi u komentaru, linkovaće ga direktno na komit. Opet, može se ukazivati na komitove u račvama ili drugim repozitorijumima na isti način kao što se radi u tiketima. Više informacija o referencama može se pronaći u [1].

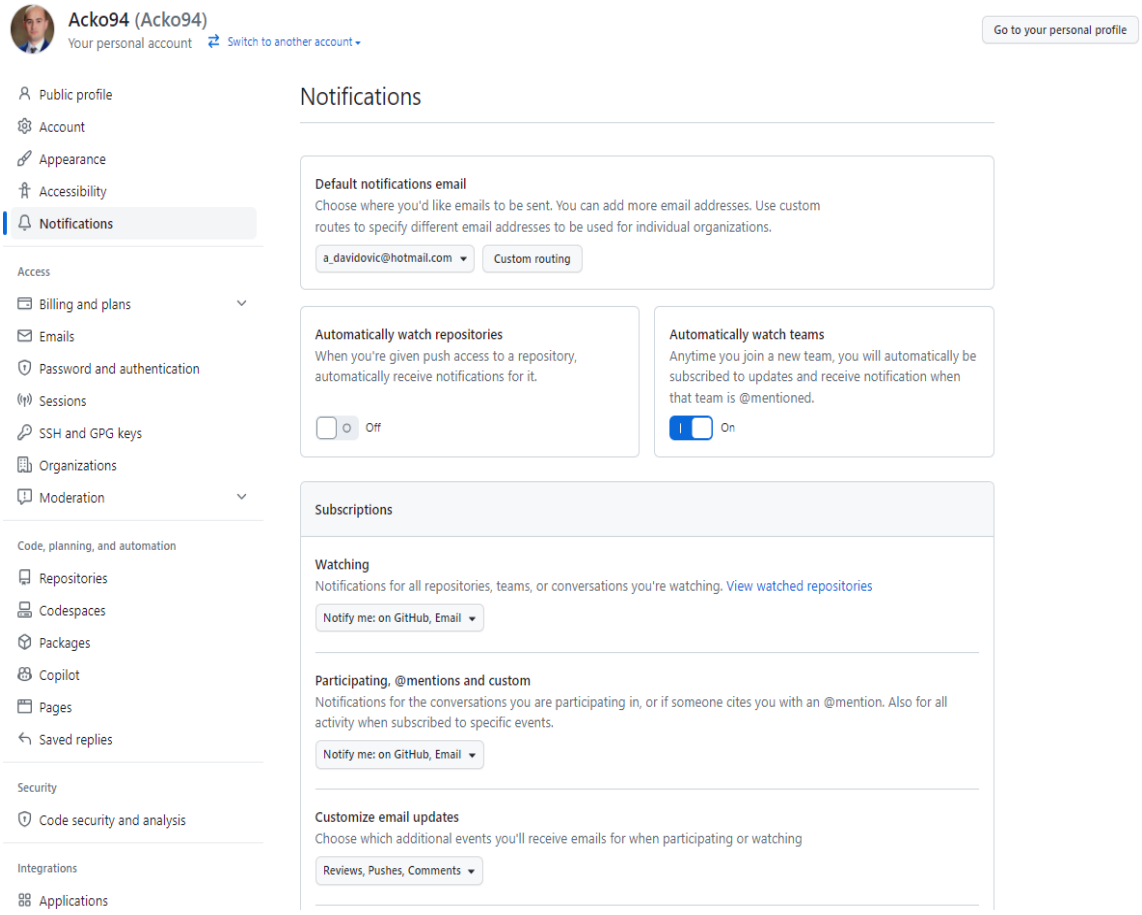
5.6 Pominjanja i obaveštenja

Platforma GitHub u sebi sadrži izuzetno funkcionalan sistem obaveštenja, koji korisniku može da koristi kada ima nekih nedoumica ili je potrebna neka povratna informacija od određenih korisnika ili od celih timova.

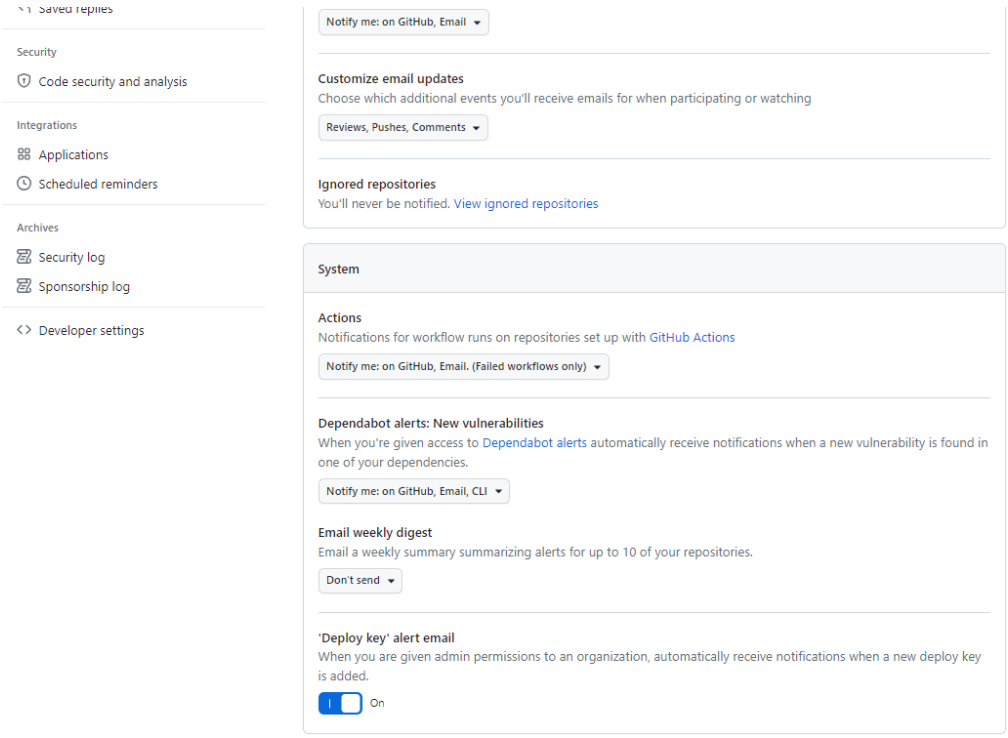
Ako se u bilo kom komentaru otkuca karakter @ pokrenuće se automatsko dovršavanje sa imenima i korisničkim imenima ljudi koji su saradnici ili daju doprinos samom projektu. Kada korisnik objavi komentar u kome se neko pominje, korisniku koji je pomenut u komentaru stiže obaveštenje. Na taj način se ljudi efikasno povezuju u razgovore. U GitHub zahtevima za povlačenje ljudi će često pozivati druge ljude iz svog tima ili kompanije da urade recenziju zahteva za povlačenje ili tiketa. Ako je neko spomenut na zahtevu za povlačenje ili tiketu,

korisnik će biti „pretplaćen” na njega i nastaviće da dobija obaveštenja svaki put kada se tu dogodi neka aktivnost. Takođe, korisnik će biti pretplaćen na sve što sam otvori, ako nadgleda repozitorijum ili se komentariše nešto. Ukoliko nema potrebe da se dalje dobijaju obaveštenja, postoji opcija kojom se prekida praćenje i isključuju obaveštenja (eng. *Unsubscribe*).

Obaveštenje predstavlja način na koji GitHub pokušava da stupi u kontakt sa korisnikom onda kada se dese neke aktivnosti ili promene. U okviru notifikacija, do kojih se stiže preko opcije podešavanja, korisnik može da izabere da li će pratiti obaveštenja o repozitorijumima kod kojih je podneo zahtev za povlačenje. Korisnik može da isključi obaveštenja o timovima čiji je član, iako je inicijalno podešeno da korisnik dobija obaveštenja o svim timovima. Svi ovi alati su veoma korisni za rukovanje velikim brojem obaveštenja (videti slike 22 i 23). Mnogi napredni korisnici platforme GitHub će isključiti obaveštenja putem mejla i u potpunosti će upravljati obaveštenjima direktno preko opcije notifikacija (eng. *Notifications*) na samoj platformi GitHub.



Slika 22: Opcije podešavanja notifikacija



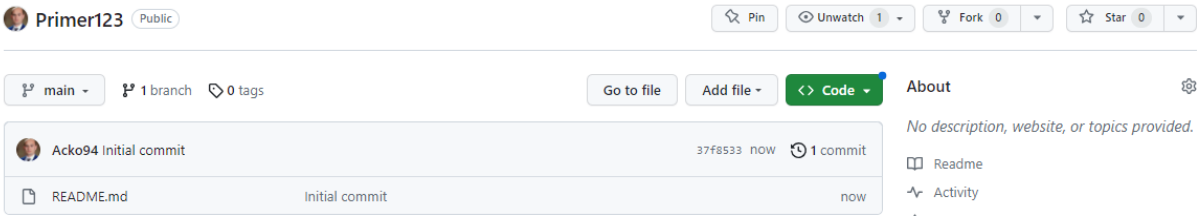
Slika 23: Dodatne opcije podešavanja notifikacija

6 Posebni fajlovi

U ovom poglavlju biće opisani fajlovi koji se nalaze u korisničkom repozitorijumu, a to su fajl za instrukcije i uputstva (eng. *Readme file*) i fajl za doprinos projektu (eng. *Contributing*).

6.1 Fajl za instrukcije i uputstva - Readme fajl

Jedan od najčešćih fajlova koji se koristi na GitHub platformi je fajl za instrukcije i uputstva, a on se nalazi u formatu koji platforma GitHub prepoznaje kao tekst, kao što su README, README.md, README.asciidoc itd. Uglavnom se ovaj fajl nalazi na početnoj stranici projekta.



Slika 24: Readme fajl u repozitorijumu

U fajlu za instrukcije i uputstva se nalaze sve bitne informacije o projektu za korisnika koji se prvi put susreće sa repozitorijumom ili projektom. Obično se pišu sledeće stvari:

- šta je namena projekta,
- kako ga konfigurisati i instalirati,
- primer koji pokazuje kako se koristi ili kako se pokreće,
- licenca pod kojom je projekat dostupan,
- kako se daje doprinos projektu.

GitHub servis prikazuje ovaj fajl, a dodatno se mogu postaviti slike ili linkovi kako bi se olakšalo razumevanje samog dokumenta [1].

6.2 Doprinos projektu - fajl Contributing

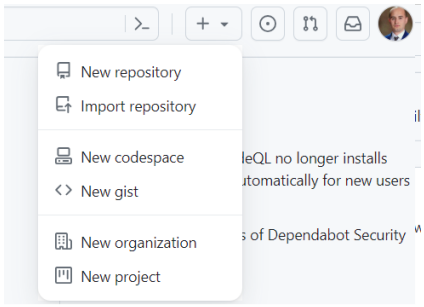
Fajl koji se bavi doprinosom projektu (eng. *Contributing*) daje osnovne informacije potencijalnim saradnicima projekta o tome kako mogu da pomognu projektu. Njega platforma GitHub prepoznaje, kao i fajl za instrukcije i uputstva. Osnovna ideja ovog fajla je da autor koda eksplicitno navede šta treba, odnosno šta ne treba da bude deo zahteva za povlačenje. Kadgod postoji fajl za doprinos projektu, GitHub će nuditi opciju otvaranja zahteva za povlačenje, ukoliko korisnik otvori zahtev za povlačenje.

7 Organizacije

Pored naloga za jednog korisnika, platforma GitHub ima dodatnu opciju koja se naziva organizacije. Kao i lični nalozi, organizacioni nalozi imaju prostor imena u kom postoje svi projekti te organizacije, ali mnoge druge stvari se razlikuju. Ovi nalozi predstavljaju grupu korisnika sa zajedničkim vlasništvom nad projektima i postoji veliki broj alata za upravljanje podgrupama tih korisnika. Obično se ovi nalozi koriste za grupe otvorenog koda ili kompanije, što se može videti u [1].

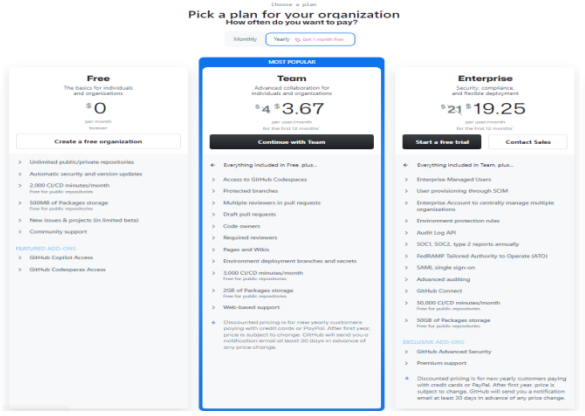
7.1 Kreiranje organizacija

Kreiranje organizacije je prilično jednostavno. U gornjem desnom uglu menija izabere se dugme „+" i izabere se opcija kreiranja nove organizacije (eng. *New organization*). Ovo se može uraditi i na bilo kojoj stranici platforme GitHub.



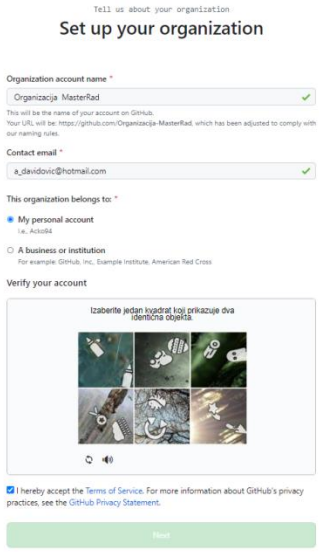
Slika 25: Kreiranje organizacije

Kao što se može videti na slici 26, prilikom kreiranja organizacije može se izabrati besplatna opcija ili neka od opcija koja zahteva plaćanje. Za korisničke potrebe, uglavnom se izabere besplatna verzija.

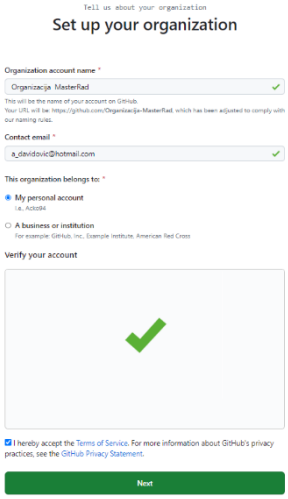


Slika 26: Izbor vrste organizacije

Nakon navedenog koraka, treba dati ime organizaciji i uneti mejl adresu kao glavnu vezu komunikacije sa grupom. Opciono, mogu se pozvati i drugi korisnici da postanu suvlasnici organizacije. Osim toga, potrebno je izabrati vrstu organizacije, verifikovati nalog i prihvatiti uslove korišćenja.

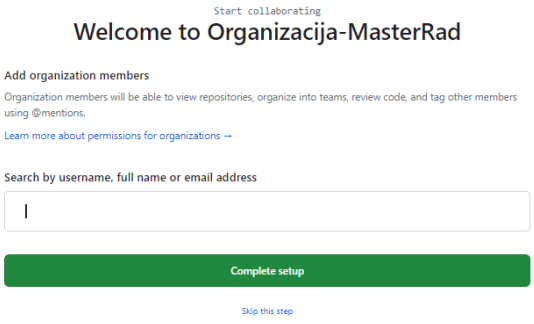


Slika 27: Proces unosa podataka



Slika 28: Uspešno uneti podaci za organizaciju

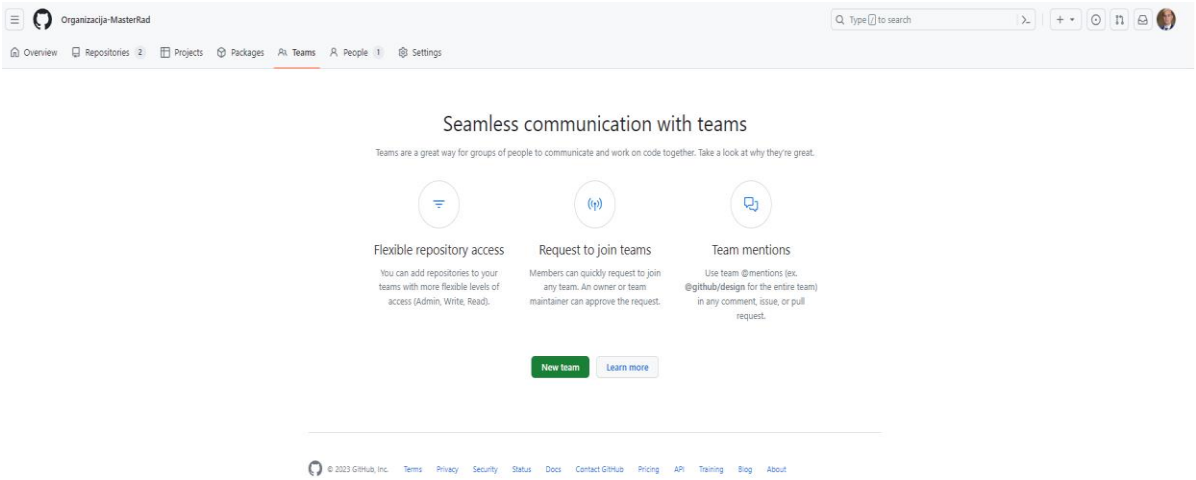
Kao vlasnik organizacije, korisnik prilikom račvanja repozitorijuma, ima mogućnost da ga račva u prostor imena svoje organizacije. Kada korisnik kreira nove repozitorijume, može ih kreirati ili pod ličnim nalogom, ali takođe i pod bilo kojom organizacijom u kojoj je jedan od vlasnika [1].



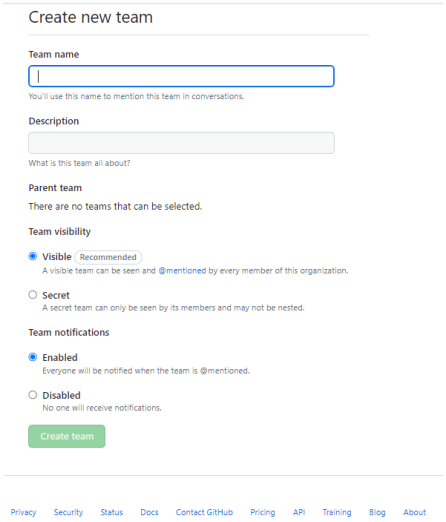
Slika 29: Uspešno kreirana organizacija

7.2 Timovi

Organizacije na platformi GitHub su sa pojedincima povezane preko timova, koji predstavljaju grupe pojedinačnih naloga i repozitorijuma u okviru organizacije, zajedno sa vrstom pristupa koji ti korisnici imaju u tim repozitorijumima. Organizacije daju vlasnicima i pristup svim informacijama o tome šta se događa u organizaciji. Postoji opcija kojom se može ispratiti sve u vezi događaja i promena na nivou same organizacije, ko je uradio to i gde su izvršene te promene (eng. *Audit log*), kao što se može videti u [1].



Slika 30: Timovi unutar organizacija



Slika 31: Kreiranje timova

Broj timova u kojima korisnik može da bude član nije ograničen. Korisnik ne mora da bude deo tima za kontrolu pristupa. Takođe, postoje timovi ljudi koji imaju zajednička interesovanja, gde korisnici razmenjuju svoja mišljenja i iskustva.

8 Pisanje skripti za GitHub

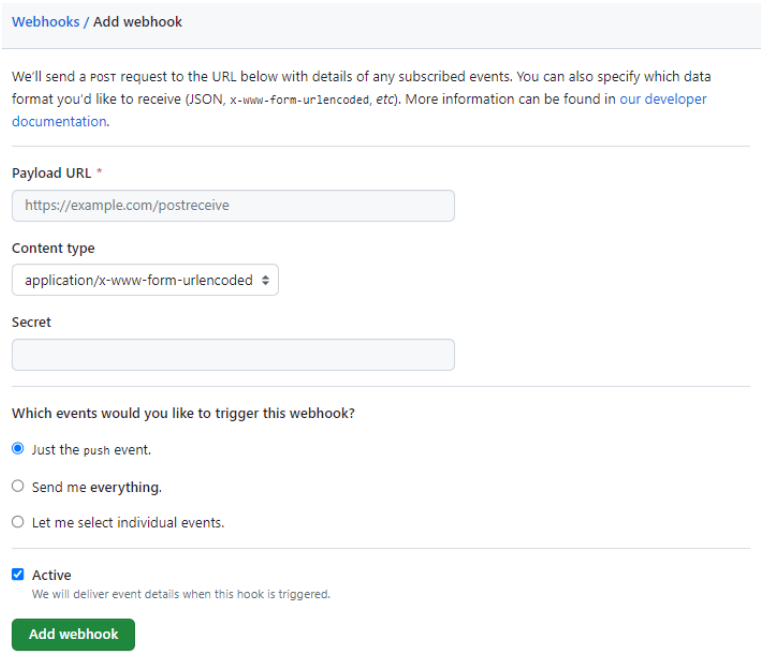
Do sada su objašnjene najznačajnije osobine i procesi rada na platformi GitHub, ali svaka veća grupa ili projekat ima potrebu da servise na platformi prilagodi svojim potrebama. Dodatno, postoji mogućnost da se integrišu neki spoljni servisi. U ovom odeljku biće prikazano kako se platforma GitHub i njen API mogu prilagoditi tako da odgovaraju potrebama korisnika.

8.1 Kuke - Webhooks

Ako je korisniku potrebno nešto što GitHub platforma nema na svom spisku servisa, postoji opcija koja se naziva kuke (eng. *Webhooks*). Kuke za neki repozitorijum na GitHub-u su prilično jednostavne. Pre svega, navodi se URL i onda će GitHub poslati HTTP tovar ka toj URL adresi kada se desi neka željena promena.

Da bi kuka bila omogućena, u okviru podešavanja postoji deo koji se naziva kuke (eng. *Webhooks*). Da bi se postavila nova kuka, izabere se opcija dodaj kuku (eng. *Add webhook*).

Konfiguracija prikazana na slici 32 je prilično jednostavna. U većini slučajeva unese se URL i ključ, pa se klikne na dugme dodaj kuku (eng. *Add webhook*). Postoji više opcija za koje događaje korisnik želi da mu servis GitHub šalje tovar (podrazumevano se tovar prima samo za *push* događaje, tj. kada neko napiše novi kôd na bilo kojoj grani korisničkog repozitorijuma) [1].

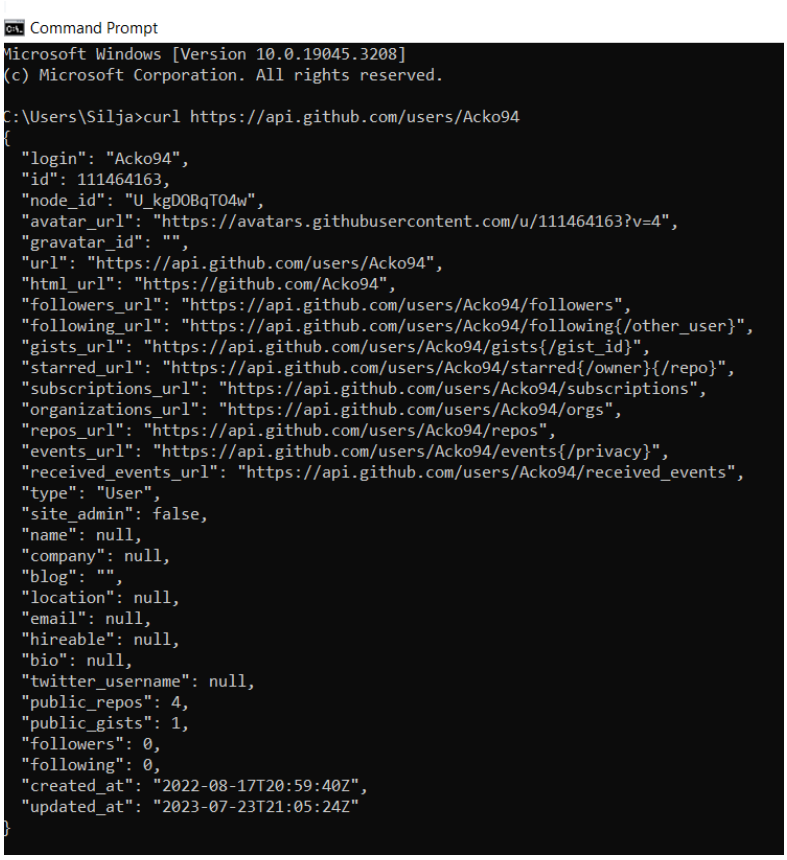


Slika 32: Konfiguracija veb kuke

8.2 Alat GitHub API

Kuke i ostali servisi predstavljaju način da korisnik dobije obaveštenja u vezi sa događajima na korisničkim repozitorijumima. Međutim, ukoliko su korisniku potrebne dodatne informacije o ovim događajima, ili ukoliko je potrebno da se automatizuje dodavanje saradnika ili dodavanje oznaka tiketima, veoma koristan alat za navedeno jeste GitHub API. GitHub ima veliki broj API krajnjih tačaka koje korisniku omogućavaju da radi skoro sve što može na veb sajtu, ali je sve automatizovano.

Najtrivijalnija stvar koja može da se uradi jeste običan GET zahtev na krajnjoj tački koja ne zahteva autenitfikaciju, kao što se može videti na slici 33. Pomoću ovog zahteva može da se dobije informacija o korisniku ili informacija koja može da se pročita u vezi sa nekim postojećim projektom.

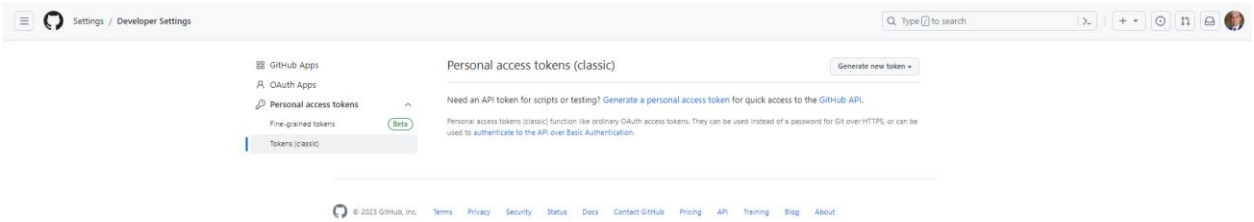


Slika 33: Proces dobijanja informacija o korisniku

Postoji veliki broj krajnjih tačaka kao što je proces dobijanja informacija o korisniku, sa kojih mogu da se dobiju informacije o skoro svemu što može da se vidi javno na platformi GitHub.

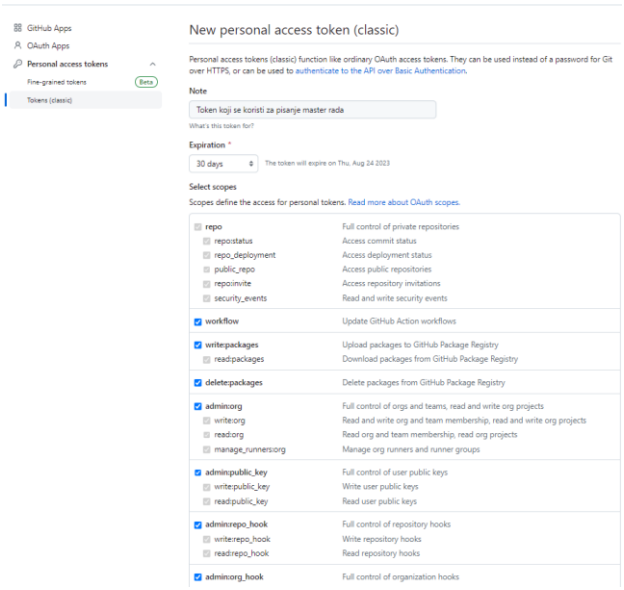
Ukoliko korisnik želi da nešto uradi na veb sajtu, poput komentarisanja na tiket ili zahtev za povlačenje, ukoliko želi da pogleda nešto ili ima interakciju sa privatnim sadržajem, mora da izvrši postupak autentifikacije. Za proces autentifikacije postoji više načina. Osnovna

autentifikacija predstavlja korišćenje korisničkog imena i lozinke, ali u opštem slučaju je efikasnije korišćenje ličnog pristupnog tokena. On se generiše tako što se u okviru stranice podešavanja (eng. *Settings*) izabere poslednja kartica o programerskom podešavanju (*eng. Developer settings*) i opcija tokena (eng. *Personal access tokens*).

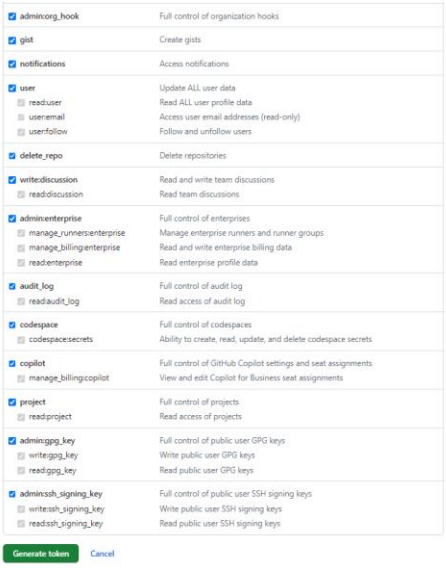


Slika 34: Mesto za generisanje tokena u okviru podešavanja

Kada se izabere opcija generisanja novog tokena (eng. *Generate new token*) otvara se veliki broj opcija za koje će važiti ovaj token, kao i za opis. Važna stvar koja se odnosi na token jeste period trajanja tokena, koji može biti ograničen ili trajan. Preporuka je da korisnik unese sadržajan opis kako bi mu bilo lakše da ukloni token iz skripte ili aplikacije kada prestane da ga koristi.

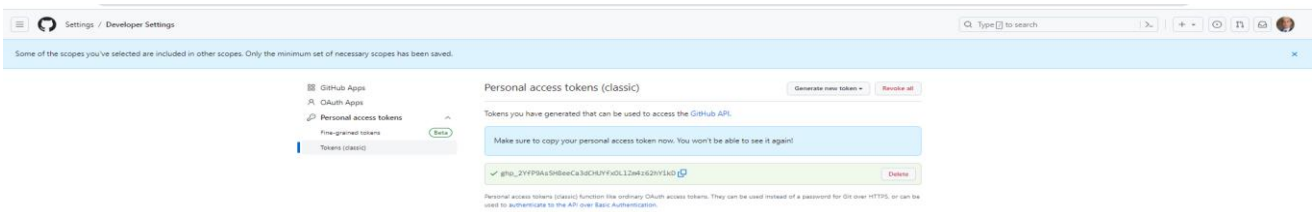


Slika 35: Kreiranje novog tokena I



Slika 36: Kreiranje novog tokena II

GitHub token prikazuje samo jednom, zato je preporuka da se odmah kopira i sačuva. Ovo se može koristiti za potvrdu identiteta korisnika u skripti umesto korišćenja korisničkog imena i lozinke. Može se ograničiti područje nad kojim skripta deluje, a sam token se može opozvati. Dodatno, ovaj vid autentifikacije povećava granicu učestalosti zahteva sa 60 zahteva po satu na čak 5 000 zahteva u istom vremenskom intervalu.

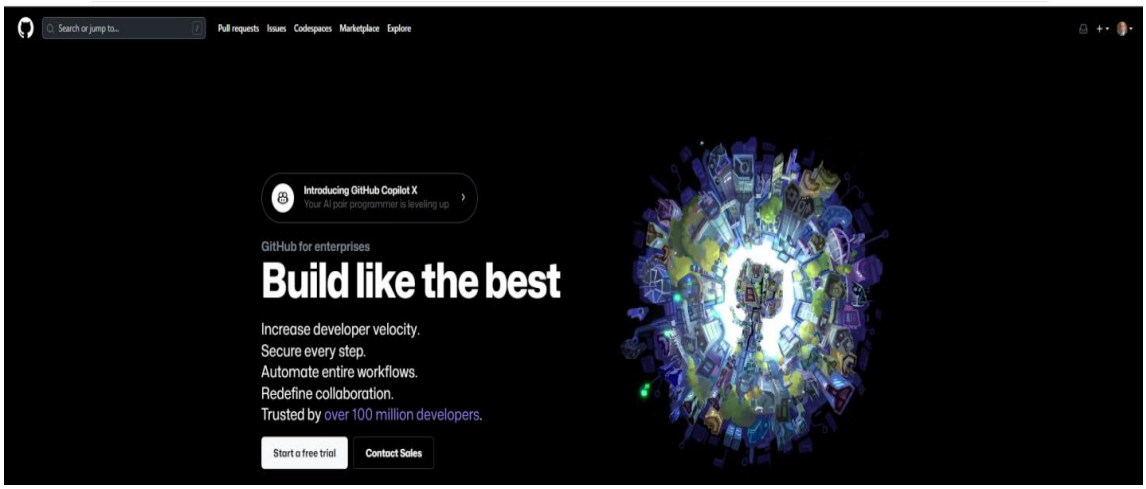


Slika 37: Kreirani token

Pomoću alata API može da se uradi skoro sve što može da se uradi na sajtu-mogu da se naprave i podešavaju prekretnice, da se dodeljuju tiketi ljudima i zahtevima za povlačenje, da se prave i menjaju oznake, da se pristupa podacima o komitu, da se prave novi komitovi i grane, da se otvaraju, zatvaraju i spajaju zahtevi za povlačenje, da se prave i menjaju timovi, da se komentarišu linije koda u zahtevu za povlačenje, da se pretražuje sajt i slično [1].

9 Razvoj softvera unutar preduzeća

Postoji i samostalna platforma za razvoj softvera unutar preduzeća (eng. *GitHub Enterprises Server).* Funkcioniše tako što omogućava velikim organizacijama da upravljaju svojim kodom na privatnim serverima, čime je omogućena bolja kontrola, ali i veća sigurnost.



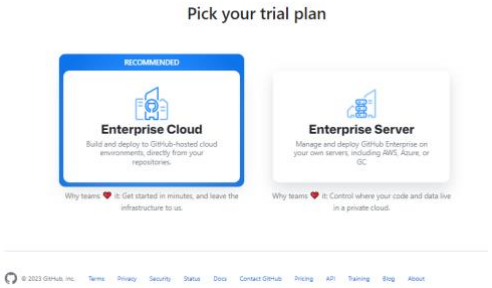
Slika 38: GitHub for enteprises

Da bi se videle sve opcije GitHub-a za preduzeća, sama platforma nudi dve opcije. Prva je probna verzija, izborom opcije započinjanja besplatnog probnog perioda korišćenja (eng. *Start a free trial),* a druga verzija se plaća, izborom opcije kojom će biti kontaktirana prodaja (eng. *Contact Sales*). Velika preduzeća biraju drugu opciju, zato što razvoj njihovog softvera predstavlja ozbiljan i odgovoran posao. GitHub za razvoj softvera unutar preduzeća omogućava:

- povećanje brzine programera,
- sigurnost u svakom koraku,
- automatizuju se čitavi procesi,
- redefiniše se saradnja.

Izborom opcije započinjanja besplatnog probnog perioda otvaraju se dve mogućnosti.

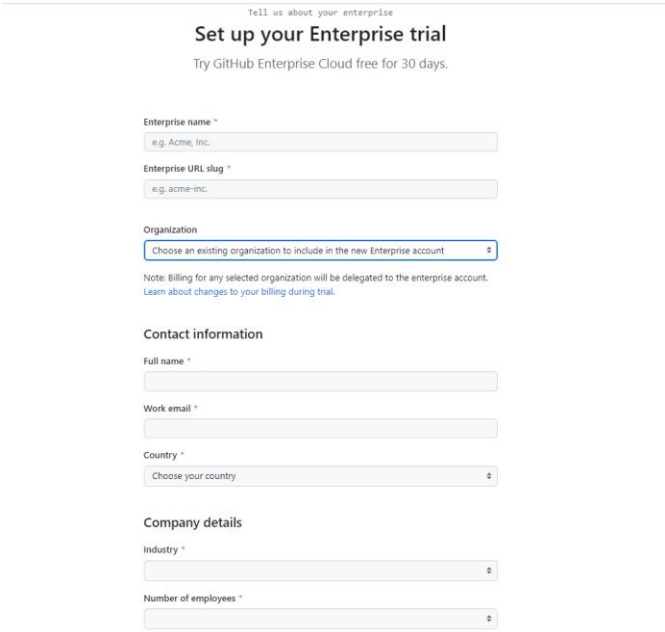
- **Enterprise cloud** se nalazi u oblaku, što znači da ne mora da se instalira na lokalnom serveru. Ovo omogućava korisnicima da pristupe svom kodu bilo gde i bilo kada.
- **Enterprise server** predstavlja mogućnost koja se koristi tako što se verzija GitHub platforme instalira na lokalnom serveru određene kompanije. Ovo omogućava organizacijama da upravljaju svojim kodom na privatnim serverima, čime je obezbeđena veća kontrola nad kodom i veća sigurnost.



Slika 39: Izbor za Enterprise: Cloud ili Sever

Izborom opcije koja se nalazi u oblaku otvara se nova kartica u kojoj se od korisnika zahteva unos podataka vezano za preduzeće i to:

- naziv preduzeća/kompanije,
- URL adresa preduzeća/kompanije,
- kontakt informacije (puno ime i prezime, poslovni mejl, država),
- podaci o kompaniji (vrsta industrije, broj zaposlenih).

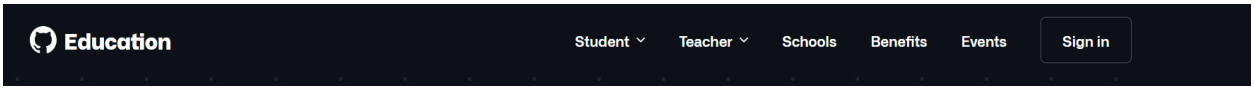


Slika 40: Unos podataka o kompaniji

Dodatno, može se dodati neka već postojeća organizacija koju je korisnik kreirao na GitHub platformi, ali nije neophodno. Kada se izvrši provera, prihvate uslovi korišćenja i kreira preduzeće počinje period od 30 dana besplatnog korišćenja ovog dela platforme, kako bi se korisnik upoznao sa svim funkcionalnostima. Ukoliko je potrebno dalje korišćenje, nakon isteka roka mora se izvršiti modifikacija naloga. Sve funkcionalnosti ovog dela platforme GitHub mogu se videti u [2].

10 Upotreba GitHub platforme u obrazovanju

Na GitHub platformu za obrazovanje (eng. *GitHub for Education*) korisnik se može prijaviti sa već kreiranim nalogom za GitHub, iz razloga što je ona deo GitHub platforme.



Slika 41: Izgled trake sa opcijama GitHub Education

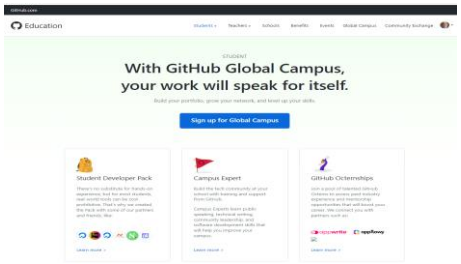
GitHub platforma za obrazovanje pomaže učenicima, nastavnicima i školama da pristupe najboljim alatima iz celog sveta, različitim programima obuka kako bi uspešno obrazovali buduće generacije.

Za studente postoje tri mogućnosti.

- **Student Developer pack** predstavlja najbolje alate za programere, koji su besplatni za studente.

- **GitHun Campus Expert** daje mogućnost uspostavljanja tehnološke zajednice u školi uz pomoć i podršku GitHub platforme.

- **Join our tim as a GitHub Intern** daje mogućnost da se studenti prijave kao GitHub pripravnici i direktno u sedištu GitHub-a u San Francisku (SAD) steknu nova iskustva.



Slika 42: Mogućnosti za studente u okviru GitHub for Education platforme

Da bi se mogle koristiti sve mogućnosti koje GitHub platforma u obrazovanju nudi za studente, u jednom trenutku će biti neophodna verifikacija naloga uz pomoć dokumenta koji pokazuje status studenta.

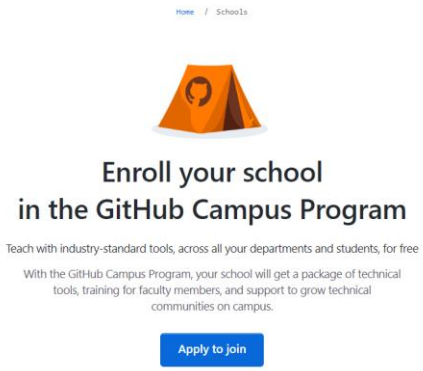
Što se tiče nastavnika, postoje sledeće mogućnosti.

- **Campus Advisors** predstavlja alat namenjen nastavnicima, koji će biti u stanju da predaju pomoću GitHub-a, nakon obuke koja objašnjava kako GitHub radi u pozadini.

- **GitHub ClassRoom** je alat koji uspešno integriše GitHub u kurs koji nastavnik predaje.

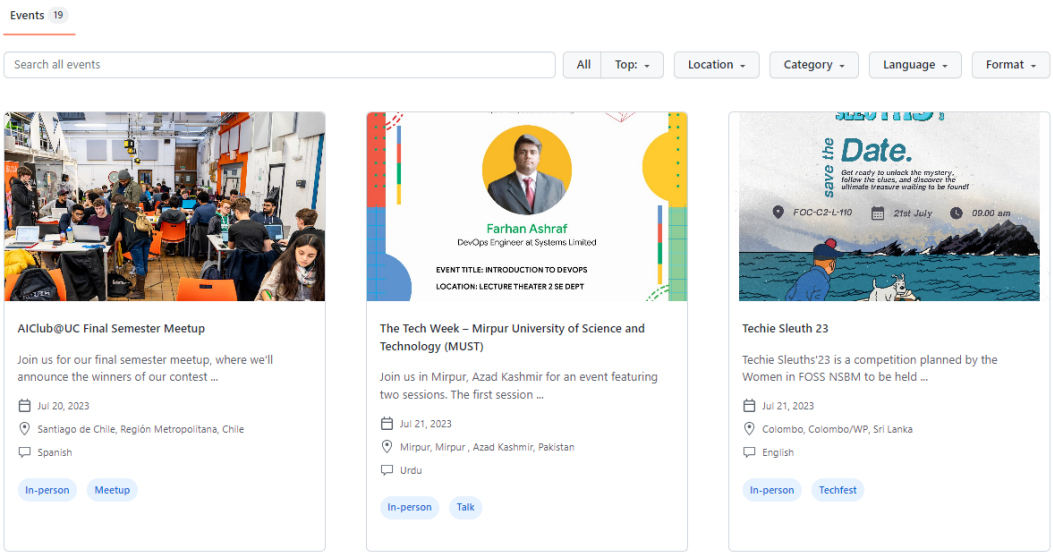
- **TeacherToolbox** jesu najbolji alati za podučavanje, besplatni za nastavnike.

Na niovu škole omogućen je **GitHub Campus Program**, koji podrazumeva učenje pomoću svih standardnih alata, u svim odeljenjima i sa svim učenicima.



Slika 43: Apliciranje za uključivanje škole u GitHub Campus Program

Jedna od novijih, ali i najkorisnijih opcija u okviru dela platforme GitHub vezano za obrazovanje, predstavlja opcija događaja (eng. *Events*). U okviru nje korisnici mogu da se upoznaju sa hakatlonima, konferencijama i događajima koji ih zanimaju. U okviru pretrage događaja korisnici mogu, koristeći različite filtere, da pretragu prilagode svojim interesovanjima. Uz svaki događaj koji predstoji, nalazi se i opis da li će biti uživo ili onlajn, kao i sama vrsta događaja. Sve informacije, kao i detaljan opis mogućnosti mogu se videti u [2].



Slike 44: Događaji koji predstoje u okviru GitHub for Education

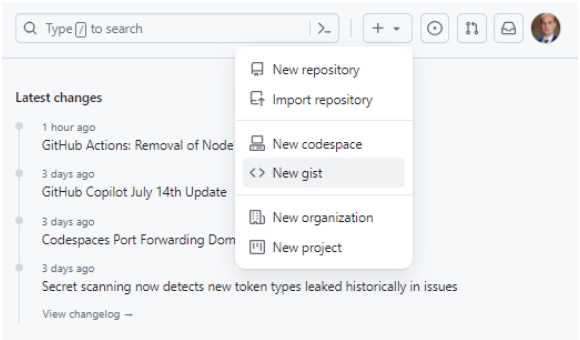
11 Deljenje isečaka koda - Gist

Deo platforme GitHub koji se naziva Gist je sličan repozitorijumu. Gist omogućava da se segmenti koda na veoma jednostavan način dele sa drugima. Svaki Gist je Git repozitorijum, što znači da može da se račva i da se klonira. Kada korisnik kreira novi Gist, on će automatski biti povezan sa nalogom i moći će da se vidi na Gist listi korisnika. Gist može biti interni ili tajni.

- Interni Gist se može naći pomoću linka *https:gist.[hostname]/discover*. Tu članovi određenog preduzeća ili kompanije mogu da pogledaju Gist fajlove koji su kreirani. Oni se mogu pronalaziti, tako da se mogu koristiti i može se videti rad sa njima.

- Tajni Gist se ne može pronaći pomoću linka *https:gist.[hostname]/discover*. Oni se ne moću naći pomoću pretraživanja, osim ako korisnik nije prijavljen i nije autor istih. Važno je napomenuti da tajni gist repozitorijumi nisu privatni. Ako se pošalje URL adresa nekog Gist repozitorijuma drugom članu preduzeća, on će moći da ga vidi. Ukoliko se pak želi sakriti sadržaj od drugih, onda je bolje napraviti privatni repozitorijum [2].

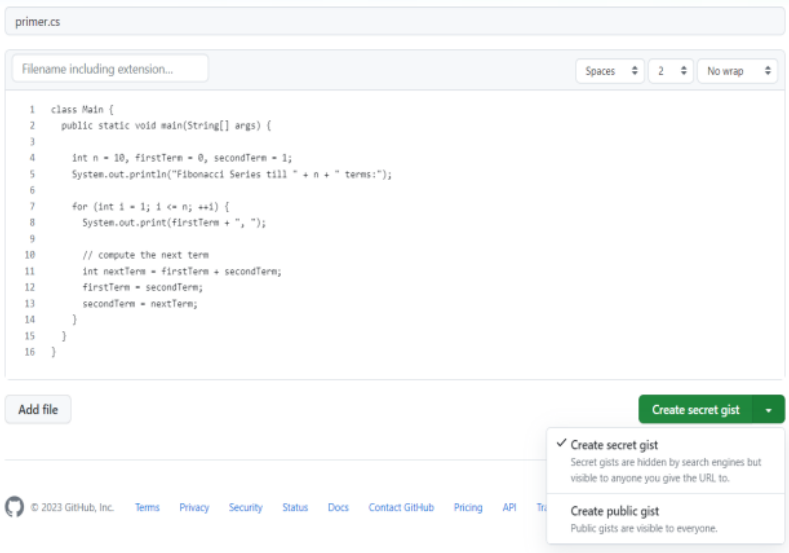
Nakon kreiranja Gist-a, on ne može biti prebačen iz internog u tajni. Međutim, tajni Gist može postati vidljiv ukoliko se u podešavanjima vidljivost prebaci na javni (eng. *Public*). Novi Gist, slično kreiranju repozitorijuma, organizacije ili projekata, kreira se izborom opcije novi gist (eng. *New Gist*), nakon čega se otvara prozor gde se mogu izabrati ime, ekstenzije i slično, a zatim se u glavnom delu dodaje sadržaj ili kôd koji treba da se sačuva. Kada se izabere opcija da se sačuva novi Gist postoji opcija da se sačuva tako da li će biti javan i vidljiv drugim korisnicima, ili samo autoru. To se kasnije može menjati. Opcija dodavanja fajlova (eng. *Add file*) služi da se u jednom Gist-u može sačuvati više različitih datoteka. Uređivanje (eng. *Edit*) i brisanje (eng. *Delete*) su veoma jednostavni.



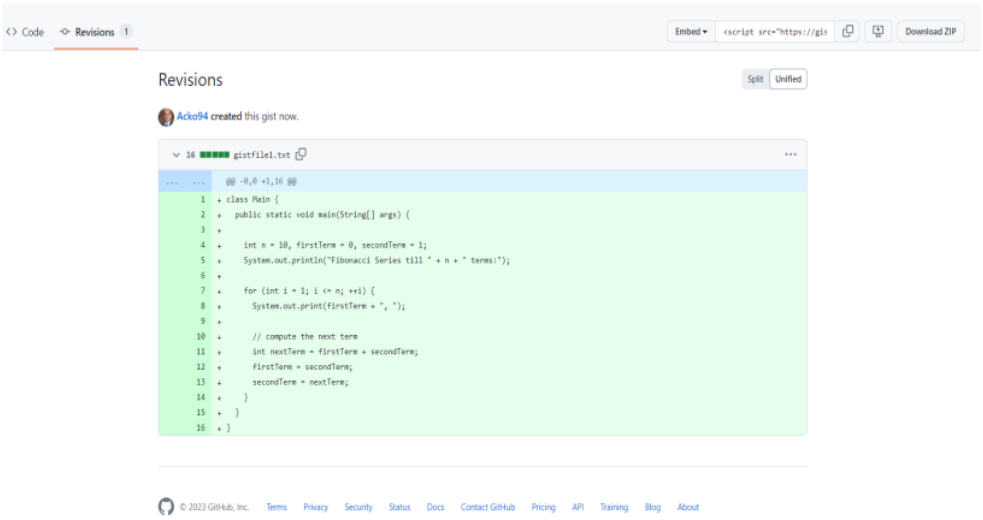
Slika 45: Kreiranje novog gista

Mogućnosti kao što su pretplata, označavanje pomoću zvezdice, komentarisanje, preuzimanje ZIP fajlova je takođe slično repozitorijumu, što ga čini veoma lakim za preuzimanje i korišćenje. GitHub platforma takođe omogućava ugradnju Gist fajla na neku veb lokaciju, korišćenjem opcije ugradnje (eng. *Embed*) bilo gde. U delu označenom kao revizija (eng.

Revision) mogu se pratiti sve promene koje su izvršene, što se može videti na slici 47. Najčešći razlog korišćenja Gist-a jesu vezani za isečke koda ili ugradnju koda u veb lokaciju. Neke naprednije tehnike vezane za Gist mogu se pronaći u [5].



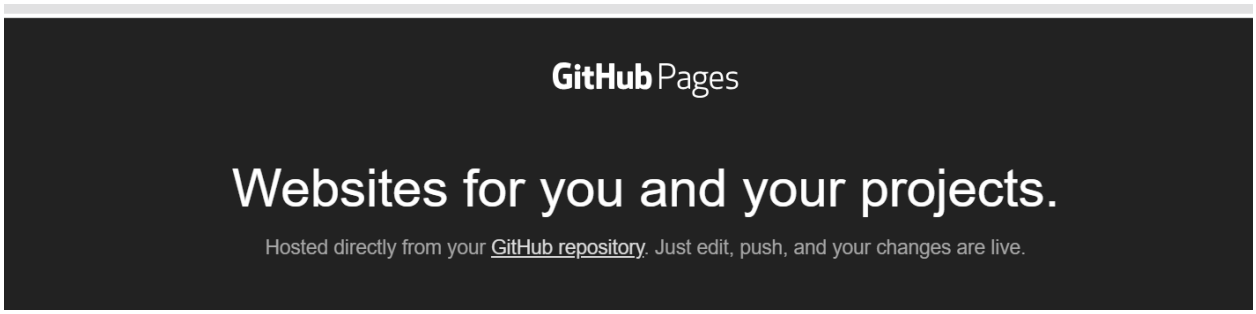
Slika 46: Imenovanje Gist fajla i izbor opcija



Slika 47: Pregled izmena Gist fajla

12 GitHub stranice

GitHub ima veoma korisnu opciju koja se zove GitHub stranice (eng. *GitHub Pages*). Ova opcija omogućava da se hostuje direktno iz skladišta sa platforme GitHub. GitHub stranice je hosting servis za statički sajt koji direktno uzima HTML, CSS i JavaScript fajlove iz repozitorijuma, opciono pokreće datoteke kroz proces izgradnje i objavljuje veb lokaciju.



Slika 48: Stranica https://pages.github.com/

Sajt se može hostovati preko GitHub domena koji glasi *github.io* ili preko sopstvenog domena. Pomoću ove opcije mogu se kreirati sajtovi koji su javno dostupni na internetu. Organizacije koje su izabrale da GitHub platformu koriste u oblaku (eng. *GitHub Enterprise Cloud*) mogu privatno da objavljuju sajtove, upravljajući kontrolom pristupa za lokaciju. Takođe, vlasnici organizacije mogu da onemoguće objavljivanje sajtova pomoću GitHub stranica iz samog skladišta organizacije.

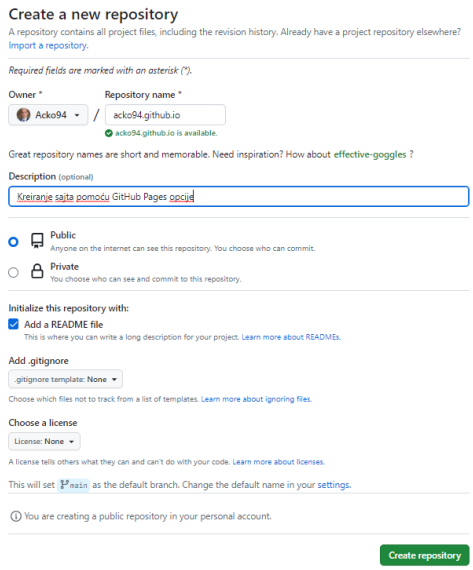
Postoje ograničenja koja se odnose na GitHub pages.

- Izvorni repozitorijumi GitHub stranica imaju preporučeno ograničenje od 1GB.
- Objavljen sajt pomoću alata GitHub stranica ne može biti veći od 1GB.
- Razvoj za GitHub stranice ističe ako traje duže od 10 minuta.
- Sajtovi koji su kreirani pomoću GitHub stranica imaju meko ograničenje opsega od 100 GB mesečno.
- Sajtovi koji su kreirani pomoću GitHub stranica imaju meko ograničenje popravki od 10 na sat vremena.

Ako veb lokacija premašuje ova ograničenja, serveri GitHub platforme možda neće biti u mogućnosti da opslužuju sajt ili će se dobiti mejl od podrške sa preporukama za ispravke. Opcija GutHub stranica nije namenjena niti je dozvoljeno da se koristi kao besplatna usluga veb-hostinga za vođenje onlajn poslovanja, sajta za e-trgovinu ili bilo koji sajt koji se koristi za komercijalne usluge, ali ni za pružanje komercijalnog softvera kao usluge (eng. *Software as a service - SaaS*). Kada se pristupi sajtu kreiranom pomoću GitHub pages, IP adresa posetioca čuva se u evidenciji iz bezbednosnih razloga. Svi detalji o GitHub pages opcijama mogu se pronaći u [2].

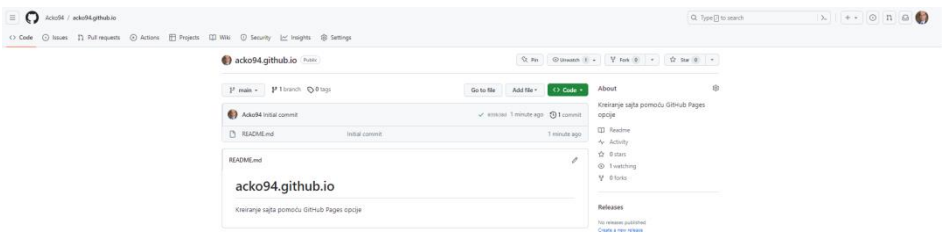
U GitHub skladištu mogu se čuvati različti kodovi. Međutim, ukoliko se koristi opcija GitHub stranica sa svim svojim funkcionalnostima, korisnički kôd bi trebalo da bude struktuiran kao tipična veb lokacija, npr. sa primarnom ulaznom tačkom HTML datoteke koja se zove *primer.html*. Dalji postupak objavljivanja stranice pomoću *GitHub Pages* servisa zahteva korišćenje komandne linije. Međutim, ukoliko korisnik želi da izbegne korišćenje komandne linije, postoji dodatna opcija korišćenja grafičkog korisničkog interfejsa.

Na jednostavnom primeru biće prikazano korišćenje korisničkog grafičkog interfejsa GitHub stranica. Neophodno je kreiranje repozitorijuma na početku, ali kreiranje mora da prati sledeću opciju kao na primeru, može se dodati opis i izabrati opcija privatnog repozitorijuma (eng. *Private*) ili javnog (eng. *Public*), kao i u većini alata na GitHub-u. Pošto je u pitanju veb stranica, biće izabrana opcija javnog repozitorijuma sa posebnim fajlom sa instrukcijama i uputstvima.



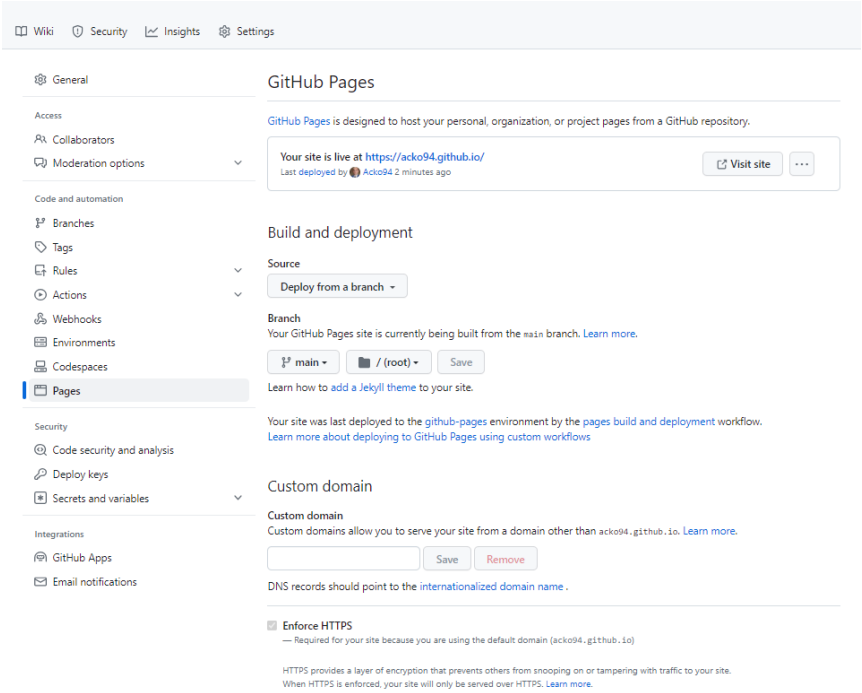
Slika 49: Kreiranje repozitorijuma za korišćenje opcije GitHub Pages

Dodatno, kao što se može videti na slici 49, nakon što se izabere naziv repozitorijuma neophodna je ekstenzija *<ime>.github.io* . U prostoru imena postoji obaveštenje korisniku da li je određeno ime koje korisnik želi za kreiranje stranice dostupno ili nije. Kada se kreira, ona je inicijalno prazna, ali se to može menjati.



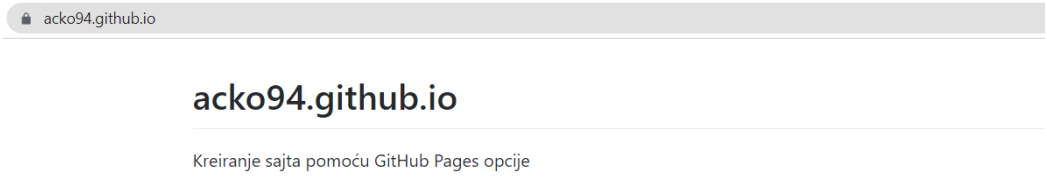
Slika 50: Repozitorijum kreiran za potrebe GitHub Pages opcije

Ako se izabere opcija podešavanja (eng. *Settings*) i dođe do opcije GitHub stranice vidi se da će stranica u primeru biti spremna za objavljivanje, kao što se može videti u [6].



Slika 51: Izgled kartice na kojoj je stranica spremna za objavljivanje

Ukoliko korisnik klikne na samu adresu ili izabere opciju da poseti sajt (eng. *Visit site*) korisniku se otvara nova kartica sa stranicom koja je prethodno kreirana. Postoje dodatne opcije podešavanja koje su prikazane na slici 51. Konačno, stranica izgleda kao na slici 52.



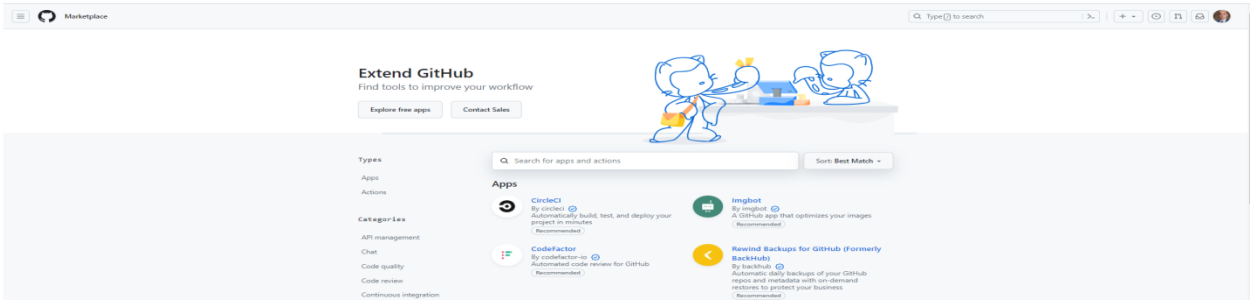
Slika 52: Izgled stranice kreirane pomoću GitHub Pages opcije

13 Softver kao usluga

GitHub platforma omogućava neke dodatne integracije poput Softvera kao usluge (eng. *Software as a Service – SaaS*). GitHub Marketplace omogućava programerima da pronađu, instaliraju i integrišu različite alate sa svojim projektima. Te usluge uključuju sledeće.

- **Vaffle.io** predstavlja upravljanje projektima za softverske timove. Automatski se vide zahtevi za povlačenje, automatizovane izgradnje, pregledi i primena u svim skladištima korisnika na GitHub platformi.
- **Rollbar** omogućava integraciju sa GitHub platformom kako bi u realnom vremenu mogle da se otklanjaju greške i dobije kompletan izveštaj o grešakama (eng. *full stack exeption reporting*). Kompatibilan je sa jezicima kao što su: JavaScript, Python, iOS, Go, C# itd.
- **Codebeat** se koristi za automatizovanu analizu koda, koja je specijalizovana za veb i mobilne programere. On je kompatibilan sa jezicima poput: Swift, JavaScript, Ruby, Kotlin itd.
- **Travis CI** služi za verodostojno testiranje i isporuku aplikacija. Takođe, omogućava potpunu kontrolu nad okruženjem za kreiranje, kako bi se prilagodilo kodu. Programski jezici koje podržava su: Java, JavaScript, ObjectiveC, Python, PHP, Ruby i Swift.
- **GitLocalize** je razvijen za timove koji svoje sadržaje prevode sa jedne tačke na drugu. GitLocalize automatski vrši sinhronizaciju sa repozitorijumom korisnika, tako da korisnik može nesmetano da nastavi da radi na GitHub platformi. Dodatne opcije daju mogućnost da korisnik dobija obaveštenja o tome šta je potrebno da se prevede.

GitHub Marketplace povezuje korisnika sa programerima koji žele da prošire i usavrše svoje GitHub tokove rada. Mogu se izlistati besplatni i plaćeni alati koje programi koriste, a nalaze se u okviru dela GitHub Marketplace. Spisak svih dostupnih alata nalazi se u [2].



Slika 53: GitHub Marketplace

GitHub Marketplace nudi dve mogućnosti programerima i to:

- **GitHub Apps** jesu aplikacije koje se mogu integrisati sa GitHub-om kako bi se olakšalo korišćenje platforme i poboljšala iskustva korisnika.
- **GitHub Actions** predstavlja uslugu koja omogućava automatizaciju različitih zadataka u GitHub-u, kao što su testiranje, izgradnja i samo objavljivanje koda.

Zaključak

Elektronske lekcije kreirane za potrebe ovog rada korisnicima omogućavaju da steknu osnovna znanja o platformi GitHub i mogućnostima koje ona pruža. Rad sadrži dosta korisnih informacija i praktičnih primera koje čitaocu mogu da posluže u procesu upoznavanja sa ovom platformom otvorenog koda, a neke naprednije opcije u okviru platforme, kojih ima mnogo, ostavljaju se korisniku da sam to istražuje.

Svaku lekciju prate slike, koje predstavljaju snimke ekrana računara sa platforme GitHub. Dat je kratak osvrt na istoriju GitHub-a, kao i uvod o samom GIT-u. Korisniku su prikazana uputstva vezano za kreiranje naloga kao i opcije koje se prilikom kreiranja naloga nude. Prikazano je kreiranje i korišćenje repozitorijuma, preuzimanje tuđih kodova sa platforme, njihovo menjanje i vraćanje promena nazad autoru. Deo rada prikazan je korišćenjem komandne linije, iako je platforma GitHub zamišljena sa svojim grafičkim interfejsom. Između ostalih prikazani su i deo platforme koji je dizajniran za razvoj softvera velikih razmera za kompanije (eng. *GitHub Enterprise*), deljenje isečaka koda-Gist, GitHub stranice, program za obrazovanje (eng. *GitHub for Education*), obezbeđivanje softvera kao usluge (eng. *GitHub Marketplace*).

Literatura za korišćenje GitHub platforme je dostupna uglavnom na stranim jezicima, pre svega na engleskom. Postoji čak i mali broj tutorijala i uputstava na internetu na srpskom jeziku. Za učenje o platformi GitHub potrebno je poznavanje engleskog jezika na naprednom nivou. Elektronske lekcije kreirane za potrebe ovog rada imaju za cilj da korisnika osposobe za korišćenje platforme GitHub i upoznavanje sa velikim brojem mogućnosti koje ona pruža, a dostupne su na sajtu Obrazovnog softvera *https://edusoft.matf.bg.ac.rs/oop.html*.

Literatura

[1] Scott Chacon, Ben Straub – *ProGit, 2nd edition,* THE EXPERT'S VOICE (prevod na srpski jezik)

[2] Zvanični sajt platforme GitHub, *https://github.com/*

[3] Istorijat nastanka platforme GitHub, *https://pslmodels.github.io/Git-Tutorial/content/background/GitHubHistory.html*

[4] Članak o razvoju GitHub platforme, *https://web.archive.org/web/20200618030357/https://wiki.dandascalescu.com/essays/pita-threshold*

[5] Video tutorijal za kreiranje Gist fajlova, *https://www.youtube.com/watch?v=xl004KsPKGE*

[6] Video tutorijal za kreiranje GitHub stranica, *https://www.youtube.com/watch?v=o5glUuFgpg*