Projektni zadatak iz predmeta

Internet softverske arhitekture

Računarstvo i automatika i Informacioni inženjering

generacija 2024/2025.

1. Namena sistema

U okviru projektnog zadatka potrebno je implementirati web aplikaciju koja predstavlja društvenu mrežu za ljubitelje zečeva, OnlyBuns, koji mogu da pregledaju i postavljaju slike svojih ljubimaca. Sem pregledanja objava, korisnici mogu i da lajkuju i postavljaju komentare, kao i da prate druge korisnike kako bi se njihove objave pojavljivale na početnoj strani aplikacije nakon prijavljivanja. Pristup sistemu imaju i administratori koji upravljaju korisničkim nalozima i pregledaju izveštaje o aplikaciji.

2. Tipovi korisnika

Informacioni sistem društvene mreže za ljubitelje zečeva razlikuje sledeće vrste korisnika:

- Neautentifikovani korisnici: imaju mogućnost da pregledaju objave korisnika i njihove profile, kao i da izvrše registraciju i prijavu na sistem.
- Registrovani korisnik: korisnici aplikacije sa svim mogućnostima neregistrovanih korisnika, i imaju mogućnost lajkovanja i komentarisanja objava, praćenja korisnika, postavljanja objava, kao i promena informacija svog profila.
- Administrator sistema: imaju mogućnosti registrovanja drugih administratora, uklanjanja objava i komentara, i pregledanja izveštaja aplikacije.

3. Funkcionalni zahtevi

3.1. Prikaz informacija neautentifikovanim korisnicima (5 bodova)

Korisnici koji nisu autentifikovani imaju prava pristupa stranici za registraciju i prijavu na sistem, i mogu da pregledaju objave korisnika automatski sortirane po vremenu nastanka (najnovije prvo), i da odu na stranicu pregleda profila korisnika koji je postavio objavu ili komentar. Nemaju mogućnost ostavljanja komentara ili lajkovanja objava, već će pri pokušaju biti obavešteni da moraju da se prijave kako bi dobili tu mogućnost. Nemaju svoj profil i nemaju prava pristupa ostalim podacima sistema. Za uspešnu implementaciju, potrebno je obezbediti zaštitu i na serverskoj i na klijentskoj strani.

3.2. Registracija korisnika i prijava na sistem (25 bodova)

Neautentifikovanim korisnicima je uvek na stranici dostupan pristup stranicama za registraciju i prijavu. Prijava na sistem se izvršava pomoći korisnikove email adrese i lozinke.

Registracija obuhvata unos email adrese, korisničkog imena, lozinke, imena i adrese, gde je potrebno implementirati neophodne validacije i poruke greške korisniku. Lozinka se unosi u dva polja da bi se otežalo pravljenje grešaka prilikom

odabira nove lozinke. Nakon popunjavanja neophodnih podataka, na datu email adresu šalje se link za aktivaciju naloga. Korisnik ne može da se prijavi na aplikaciju dok se njegov nalog ne aktivira posećivanjem linka koji je dobio u emailu.

Rešiti transakcijama problem konfliktne situacije prilikom registracije korisnika u kojem je moguće da dva korisnika u isto vreme pokušaju da se registruju sa istim korisničkim imenom. U svrhu testiranja, dodati Thread.sleep() ili napisati jedinični test kojim bi se simulirao konkurentni pristup, poput primera sa vežbi.

Napomena: potrebno je obezbediti **bilo kakav** mehanizam za autentifikaciju i autorizaciju korisnika na serverskoj strani. **Za sve funkcionalnosti treba biti implementirana autorizacija bez obzira na podelu posla po studentima!**
Takođe, potrebno je ograničiti broj pokušaja prijave na sistem na osnovu IP adrese. Sa jedne IP adrese je dozvoljeno 5 pokušaja prijava po minuti.

Napomena: Potrebno je proveru korisničkog imena implementirati/optimizovati pomoću Bloom Filtera. https://en.wikipedia.org/wiki/Bloom_filter

3.3. Periodično brisanje naloga koji nisu aktivirani (8 bodova)

Potrebno je automatizovati proces brisanja naloga koji su pokušali da se registruju ali nikad nisu aktivirali nalog putem email-a. Proces brisanja ovakvih naloga treba da se obavlja svakog poslednjeg dana u mesecu.

3.4. Profil korisnika (3 boda)

Profil korisnika bi trebalo da sadrži sledeće elemente:

- ime i prezime korisnika,
- email korisnika (nije moguće promeniti, jedinstven za nalog),
- lozinka korisničkog profila (implementirati dvostruki unos za promenu),
- adresa korisnika,
- broj pratilaca (nije moguće promeniti),
- prikaz svojih objava,
- prikaz pratilaca,
- prikaz naloga koji se prate.

3.5. Home page za korisnika (2 boda)

Na osnovnoj stranici za autentifikovanog korisnika dostupni su linkovi za:

- listu objava korisnika koji su zapraćeni,
- prikaz trendova na mreži (analitika za korisnika),
- prikaz obližnjih objava na mapi,
- čet između korisnika,
- stranica profila korisnika.

Napomena: Samo su nabrojane stranice do kojih korisnik sa svog profila treba da stigne. Studentima se prepušta organizacija stranica i izgled istih.

3.6. Home page za administratora sistema (2 boda)

Na osnovnoj stranici za administratora sistema dostupni su linkovi za:

- listu objava svih korisnika,
- prikaz trendova na mreži,
- analitiku aplikacije,
- prikaz profila svih korisnika.

Napomena: Samo su nabrojane stranice do kojih korisnik sa svog profila treba da stigne. Studentima se prepušta organizacija stranica i izgled istih.

3.7. Slanje notifikacija neaktivnim korisnicima (8 bodova)

Ukoliko korisnik nije pristupio aplikaciji u poslednjih 7 dana, potrebno je poslati korisniku email koji bi sumirao statistiku u poslednjih 7 dana na proizvoljan način (npr. obavestiti korisnika o broju novih pratilaca, lajkova, objava i slično). Notifikacija se šalje automatski od strane sistema.

3.8. Postupak kreiranja objave (20 bodova)

Registrovani korisnici su u mogućnosti da postavljaju svoje objave. Za pravljenje objave su potrebni:

- opis objave,
- slika zeca,
- lokacija objave (može da se kreira odabirom lokacije na mapi ili unošenjem adrese),
- vreme kreiranja objave.

Novokreirana objava je dostupna drugim korisnicima za pregled, lajkovanje i komentarisanje. Takođe je potrebno da bude sposobna da čuva lajkove i komentare. Informacije o lokaciji je potrebno čuvati u formatu koji je pogodan za dalji prikaz objava na mapi (npr. u formatu koordinata). Lokaciju iz objava je potrebno keširati. Strategija keširanja se prepušta studentu.

Transakcijama rešiti konfliktnu situaciju pravilnog inkrementa broja lajkovanih objava u slučaju istovremene akcije od strane više korisnika. U svrhu testiranja, dodati Thread.sleep() ili napisati jedinični test kojim bi se simulirao konkurentni pristup, poput primera sa vežbi.

.

Napomena: Za pravilnu implementaciju ove funkcionalnosti, potrebno je sliku zeca uploadovati sa računara.

3.9. Prikazivanje objave (16 bodova)

Svim korisnicima je dostupan detaljan prikaz objave koja se nalazi na njihovim stranicama. Potrebno je prikazati opis i sliku objave, kao i broj lajkova i spisak komentara. Svim registrovanim korisnicima je dostupno lajkovanje i komentarisanje objave. Korisniku koji je napravio objavu je potrebno omogućiti menjanje i brisanje objave. Obrisane i uklonjene objave ne prikazivati korisniku. Fotografije iz objava je potrebno keširati. Strategija keširanja se prepušta studentu.

3.10. Postupak komentarisanja objave (10 bodova)

Registrovanim korisnicima je dozvoljeno ostavljati komentare na objavama naloga koje prate. Komentar može da sadrži samo tekst. Komentar bi takođe trebalo da čuva vreme kreiranja i nalog koji ga je kreirao, i prikazuje se na objavi od najnovijeg do najstarijeg.

Dodatno, potrebno je ograničiti broj komentara koje može da postavi jedna osoba sa svog profila u vremenskom intervalu od sat vremena. Broj dozvoljenih komentara po nalogu je 60 po satu. Jedan nalog može da komentariše različite objave, ali ukupan broj komentara koje može da napiše je 60 u sat vremena. **Ukupan broj komentara po objavi nije ograničen.**

3.11. Postupak praćenja korisnika (13 bodova)

Registrovanim korisnicima je dozvoljeno praćenje drugog naloga navigiranjem do stranice profila korisnika i aktiviranjem praćenja tog naloga. Takođe je dozvoljen prekid praćenja naloga. Objave korisnika koji su praćeni se pojavljuju na početnoj stranici korisnika, dok se praćenja mogu videti na stranici profila naloga koji su u vezi.

Kako bi se aplikacija zaštitila od tzv. *botova,* potrebno je ograničiti broj praćenja koje korisnik može da izvrši u minutu. Korisnik može da zaprati proizvoljno mnogo naloga, ali maksimalno 50 po minuti.

Dodatno, potrebno je transakcijama rešiti konfliktnu situaciju pravilnog inkrementa broja pratilaca u slučaju ida više korisnika istovremeno zaprate isti nalog. U svrhu testiranja, dodati Thread.sleep() ili napisati jedinični test kojim bi se simulirao konkurentni pristup, poput primera sa vežbi.

3.12. Prikaz obližnjih objava na mapi (9 bodova)

Registrovanim korisnicima je dostupna mapa njihove okoline, na kojoj su obeležene obližnje objave drugih korisnika. Dozvoliti korisniku da pronalazi lokacije objave navigirajući mapu. Za potrebe ove funkcionalnosti, prilikom prikaza je potrebno koristiti informacije o lokaciji objave koji su uneti prilikom kreiranja objave. Za pravilnu implementaciju ove funkcionalnosti, potrebno je prilikom početnog učitavanja mape da se ona centrira na adresu upisanu na profilu korisnika.

3.13. Prikaz trendova mreže (16 bodova)

Registrovanim korisnicima je dostupna stranica sa prikazom trendova mreže. Na njoj se prikazuje ukupan broj objava na mreži, kao i broj objava u prethodnih mesec dana. Takođe, prikazuje se pet najpopularnijih objava (najviše lajkovanih) u prethodnih 7 dana, kao i 10 najpopularnijih objava ikada. Takođe, prikazati i listu 10 naloga koji su podelili najviše lajkova na mreži u prethodnih nedelju dana. Potrebno je implementirati keširanje objava koje su izlistane.

3.14. Prikaz analitike aplikacije (10 bodova)

Administratorima sistema je dostupna stranica sa prikazom analitike aplikacije, gde se prikazuju informacije koje mogu da pomognu administratoru prilikom upravljanja aplikacijom. Potrebno je prikazati broj objava i komentara, na nedeljnom, mesečnom i godišnjem nivou. Takođe prikazati procente korisnika koji su napravili objavu, napravili samo komentare, i nisu napravili nijedno, na radijalnom grafiku.

3.15. Prikaz registrovanih korisnika (9 bodova)

Administratoru je dostupna stranica na kojoj može da vidi prikaz svih registrovanih korisnika. Za svakog korisnika prikazano je ime i prezime, email adresa, broj objava i broj korisnika koje prati. Omogućiti pretragu korisnika po imenu, prezimenu, email adresi, kao i po broju objava (na osnovu unete minimalne i/ili maksimalne vrednosti opsega treba da se prikažu odgovarajući korisnici). Dodatno, potrebno je omogućiti sortiranje korisnika po broju korisnika koje prati i po email adresi. Prikaz korisnika treba organizovati tako da se oni prikazuju u grupama od po 5, koristeći pritom mehanizam paginacije.

3.16. Periodična kompresija slika (6 bodova)

Da bi aplikacija uštedela na masovnoj memoriji, potrebno je svaki dan pokrenuti akciju koja iterira kroz sve nekompresovane slike koje su starije od mesec dana i kompresuje ih. Biblioteku za kompresiju izaberite sami. Za potrebe projekta nemojte brisati nekompresovanu sliku.

3.17. Čet između korisnika (12 bodova)

Potrebno je omogućiti četovanje između korisnika. Razmena poruka u okviru četa treba da bude implementirana u realnom vremenu pomoću web socket tehnologije.
Potrebno je omogućiti kreiranje grupnog četa. Korisnik koji kreira grupni čet ima status admina i on može da dodaje ili uklanja ostale korisnike iz grupnog četa. Kada se doda novi korisnik, on vidi prethodnih 10 poslatih poruka u grupom četu. Na odbrani je neophodno demonstrirati komunikaciju između različitih korisnika u realnom vremenu.

3.18. Prikazivanje lokacija za brigu o zečevima (12 bodova)

Naša aplikacija je ustanovila saradnju sa organizacijom fokusiranom na brigu o zečevima, koja poseduje informacije o lokacijama koje su sposobne za njihovu brigu, kao što su azili i veterinari. Potrebno je implementirati aplikaciju organizacije za brigu o zečevima koja će našoj aplikaciji slati informacije o lokacijama na kojima korisnici mogu da nađu potrebne usluge. Slanje podataka će biti implementirano preko redova poruka, gde je potrebno omogućiti aplikaciji organizacije da se poveže i šalje poruke našoj aplikaciji društvene mreže.

Komunikaciju između dve aplikacije je potrebno implementirati preko direktnog tipa komunikacije redova poruka, i poslate poruke bi trebalo da sadrže identifikator, naziv i lokaciju na kojoj je moguće naći uslugu za brigu o zečevima. Potrebno je implementirati prijem i čuvanje ovih poruka, kao i njihov prikaz korisnicima na mapi na kojoj su prikazane objave (3.12).

Napomena: Mehanizam platforme za preuzimanje aplikacije neće biti ocenjivan, potrebno je samo implementirati potrebne funkcionalnosti za povezivanje na red poruka i slanje opisanih poruka, i demonstrirati uspešnost implementacije prilikom odbrane projekta.

3.19. Reklamiranje objava (12 bodova)

Naša aplikacija je ustanovila saradnju sa agencijama za reklamiranje. Potrebno je implementirati aplikaciju agencije za reklamiranje koja će da prima podatke o objavama naše aplikacije. Prijem podataka će biti implementiran preko redova poruka na koje je potrebno povezati aplikaciju, i prikazati primljenu poruku za potrebe demonstracije rešenja.

Administratoru sistema je dozvoljeno da označi određene objave kao pogodne za prikaz u reklamama eksterne aplikacije. Implementirati *fanout* tip komunikacije putem redova poruka gde će aplikacije agencije za reklamiranje primati informacije o označenim objavama. Poslate poruke sadrže opis i vreme objavljivanje objave, kao i korisničko ime korisnika koji je postavio objavu. Potrebno je poslati poruku svim agencijama koje imaju komunikaciju sa našom aplikacijom.

Napomena: Mehanizam agencije za reklamiranje neće biti ocenjivan, potrebno je samo implementirati potrebne funkcionalnosti za povezivanje na red poruka i prijem opisanih poruka, i demonstrirati uspešnost implementacije prilikom odbrane projekta. Kako bi se demonstrirala uspešnost *fanout* tipa komunikacije, biće potrebno pokrenuti više instanci mehanizma agencije za reklamiranje.

3.20. Praćenje aktivnosti i performansi aplikacija (12 bodova)

Kako bi se omogućio nesmetani rad aplikacije, potrebno je implementirati mehanizam nadgledanja (monitoring). Nadgledanje je potrebno implementirati pomoću alata Prometheus i Grafana. Uz pomoć ovih alata, vizuelno prikazati sledeće metrike:

- prosečno trajanje HTTP zahteva za kreiranje nove objave u toku 24h
- prosečno zauzeće CPU u određenom vremenskom intervalu
- broj trenutno aktivnih korisnika u toku 24h

3.21. Samostalna implementacija redova poruka (10 bodova)

Implementirati svoj mehanizam redova poruka za korišćenje prilikom prijema informacija o lokacijama za brigu o zečevima (3.18) po uzoru na postojeća rešenja. Potrebno je podržati čuvanje poruka i kreiranje redova poruka direktnog tipa, kao i omogućiti našoj aplikaciji i aplikaciji organizacije za brigu o zečevima da se povežu na red poruka radi komunikacije opisane u prethodnom delu specifikacije. Za pravilnu implementaciju ove funkcionalnosti, potrebno je iskoristiti ovaj mehanizam redova poruka i demonstrirati mogućnost slanja poruka o objavama aplikaciji za reklamiranje koja je opisana ranije (3.19).

3.22. Samostalna implementacija *load balancer*-a (10 bodova)

Kako bi se omogućila skalabilnost i visoka dostupnost aplikacije, potrebno je implementirati *load balancing* servis koji će raspoređivati pristigle zahteve ka različitim instancama aplikacije. Cilj *load balancera* je da ravnomerno raspoređuje zahteve kako ne bi došlo do opterećenja jedne instance aplikacije. *Load balancer* ovo postiže upotrebom nekog algoritma. Potrebno je implementirati algoritam po želji, i omogućiti *retry policy* - ponovno slanje zahteva u slučaju da odabrana instanca nije trenutno dostupna. Za uspešnu demonstraciju, potrebno je pokrenuti bar dve instance aplikacije, pri čemu će zahtevi biti preusmereni ka jednoj od pokrenutih instanci u zavisnosti od implementiranog algoritma. Potrebno je testirati rad *load balansera* nad proizvoljnim skupom zahteva. Nije potrebno podržati i distribuirani čet u ovom slučaju.

3.23. Samostalna implementacija *rate limiter*-a (10 bodova)

Potrebno je razviti jednostavan rate limiter za kontrolu broja zahteva koje korisnik može da pošalje u određenom vremenskom okviru. Ovaj rate limiter će omogućiti maksimalno 5 zahteva po minuti po korisniku. Implementacija će koristiti memoriju za skladištenje informacija o broju poslatih zahteva i vremenu kada su poslati. Demonstrirati samostalnu implementaciju na funkcionalnosti komentarisanja objava (3.10).

4. Nefunkcionalni zahtevi

4.1. Serverske platforme

Za realizaciju projekta može se izabrati serverska platforma po želji. Neke od platformi mogu biti:

- Java + Spring Boot (koristi se na vežbama)
- Java + Play framework
- Java + Spark framework
- NodeJS + Express
- Python + Django
- Ruby on Rails
- .NET
- …

4.2. Klijentske platforme

Za realizaciju projekta može se izabrati klijentska platforma po želji:

- Klasična web aplikacija
- Single-page interface aplikacija (npr. Angular + REST servisi)
- Mobilna aplikacija (Android ili iOS)

Vizuelni izgled aplikacije aplikacije se ne ocenjuje direktno, ali lepši izgled svakako ostavlja bolji utisak.

4.3. Slanje e-maila

Za slanje emaila nije obezbeđen poseban servis. Možete koristiti sopstveni email nalog. Opciono, slanje notifikacija u vidu emaila možete da odradite korišćenjem message queue-a.

4.4. API dizajn

API glavne aplikacije treba da bude dizajniran i dokumentovan u skladu sa OpenAPI specifikacijom.

4.5. Keširanje podataka

Za potrebe keširanja podataka možete koristiti bilo koji od L2 nivoa keširanja, od kojih su neki bili pokazani na terminima vežbi.

4.6. Lokacijski servisi

Za prikazivanje lokacija i mapa mogu se koristiti servisi poput OpenLayers, Leaflet, Google mapa, Yandex mapa, itd.

4.7. Grafički prikaz podataka

Za pravljenje različitih grafika mogu se koristiti third party biblioteke za iscrtavanje elemenata.

4.8. Kompresija slika

Za implementaciju kompresije slika mogu se koristiti third party biblioteke za kompresiju slika.

5. Raspodela zadataka

Student 1:
- 3.1, 3.2, 3.4, 3.12, 3.13, 3.18, 3.21

Student 2:
- 3.3, 3.6, 3.9, 3.11, 3.14, 3.15, 3.17, 3.22

Student 3:
- 3.5, 3.7, 3.8, 3.10, 3.16, 3.19, 3.20, 3.23

6. Kontrolne tačke

Kontrolna tačka 1 će se održati okvirno oko sredine novembra i nosiće 15 bodova. Potrebno je napraviti model celog sistema, kao i uraditi sledeće funkcionalnosti:

Student 1:
- 3.1,
- 3.2 - bez rate limitera, konfliktne situacije i bloom filtera;

Student 2:
- 3.9 - bez keširanja,
- 3.15 - bez paginacije;

Student 3:
- 3.5,
- 3.8 - bez keširanja i konfliktne situacije,
- 3.16.

Kontrolna tačka 2 će biti krajem decembra i nosiće 23 boda. Na ovu kontrolnu tačku dolaze sledeće funkcionalnosti:

Student 1:
- 3.2 - rate limiter i konfliktna situacija,
- 3.4,
- 3.13 - bez keširanja;

Student 2:
- 3.3,
- 3.11,

- 3.15 - paginacija;

Student 3:

- 3.7,
- 3.8 - konfliktna situacija,
- 3.10.

Za konačnu odbranu potrebno je implementirati sve ostale funkcionalnosti (osim 3.21, 3.22 i 3.23) koje nisu pokrivene kontrolnim tačkama. Funkcionalnosti 3.21, 3.22 i 3.23 nisu obavezne, i omogućavaju studentima da dobiju dodatnih 10 bodova kako bi potencijalno nadoknadili izgubljene bodove. Na svim odbranama je potrebno imati spremnu skriptu za popunu baze podataka, za potrebe demonstracije rešenja.

Timovi ili članovi tima koji ne budu izašli na jednu i / ili obe kontrolne tačke, mogu da urade tražene funkcionalnosti, ali će se u tom slučaju osvojeni bodovi skalirati po formuli bodovi = broj_osvojenih_bodova * 0.8. Formula se odnosi samo na funkcionalnosti koje studenti pokažu na odbrani a koje su već prethodno tražene za kontrolne tačke.

7. Akademska čestitost

Akademska čestitost podrazumeva samostalno pisanje radova (teksta, programskog koda i slično) uz striktno poštovanje tuđih autorskih prava. Ovaj pojam i obaveza su sastavni deo Zakona o visokom obrazovanju:

http://www.parlament.gov.rs/upload/archive/files/lat/pdf/predlozi_zakona/3048-14Lat.pdf

Više o ovoj temi možete pronaći, na primer, na sledećim linkovima:

- https://en.wikipedia.org/wiki/Academic_honor_code
- http://akademsko-pisanje.sz-ri.com/akademsko-pisanje/akademska-cestitost/

U okviru našeg predmeta to podrazumeva da studenti samostalno pišu sopstveni rad. Pomoć drugih studenata u obliku direktno preuzetih delova teksta ili programa nije dozvoljena. **Kako se ovaj projekat radi u timu, odgovornost za nepridržavanje principa akademske čestitosti snose svi članovi tima.**

Sistemi za otkrivanje plagijarizma su vremenom postali prilično efektivni. Više informacija se može naći ovde: https://theory.stanford.edu/~aiken/moss/

8. Skaliranje ocena

Projekat je predispitna obaveza čija se izrada podrazumeva u toku semestra, a odbrana na kraju. U slučaju da studenti ne uspeju u toku semestra da završe izradu projekta i uspešno odbrane urađeno, u junu i septembru će biti organizovan po dodatni termin za koji će bodovi biti skalirani na sledeći način:

- januarski rok - 100% bodova
- junski rok - 90% bodova
- septembarski rok - 80% bodova

9. Projekat za stare studente

Ukoliko stari studenti samostalno rade projekat, potrebno je odraditi sledeće funkcionalnosti:

- 3.1 - u potpunosti,
- 3.2 - bez aktivacije naloga i Bloom Filter-a,
- 3.5 - u potpunosti,
- 3.7 - u potpunosti,
- 3.8 - bez rešavanja konflikta,
- 3.9 - samo prikazivanje objave,
- 3.12 - u potpunosti,
- 3.19 - u potpunosti,
- 3.23 - u potpunosti.

Za kontrolnu tačku 1 je potrebno uraditi:

- 3.1 - u potpunosti,
- 3.2 - prijava i registracija, bez rate limiter-a i rešavanja konflikta,
- 3.5 - u potpunosti,
- 3.8 - bez keširanja,
- 3.9 - samo prikazivanje objave.

Za kontrolnu tačku 2 je potrebno uraditi:

- 3.2 - do kraja,
- 3.7 - u potpunosti,
- 3.8 - do kraja.