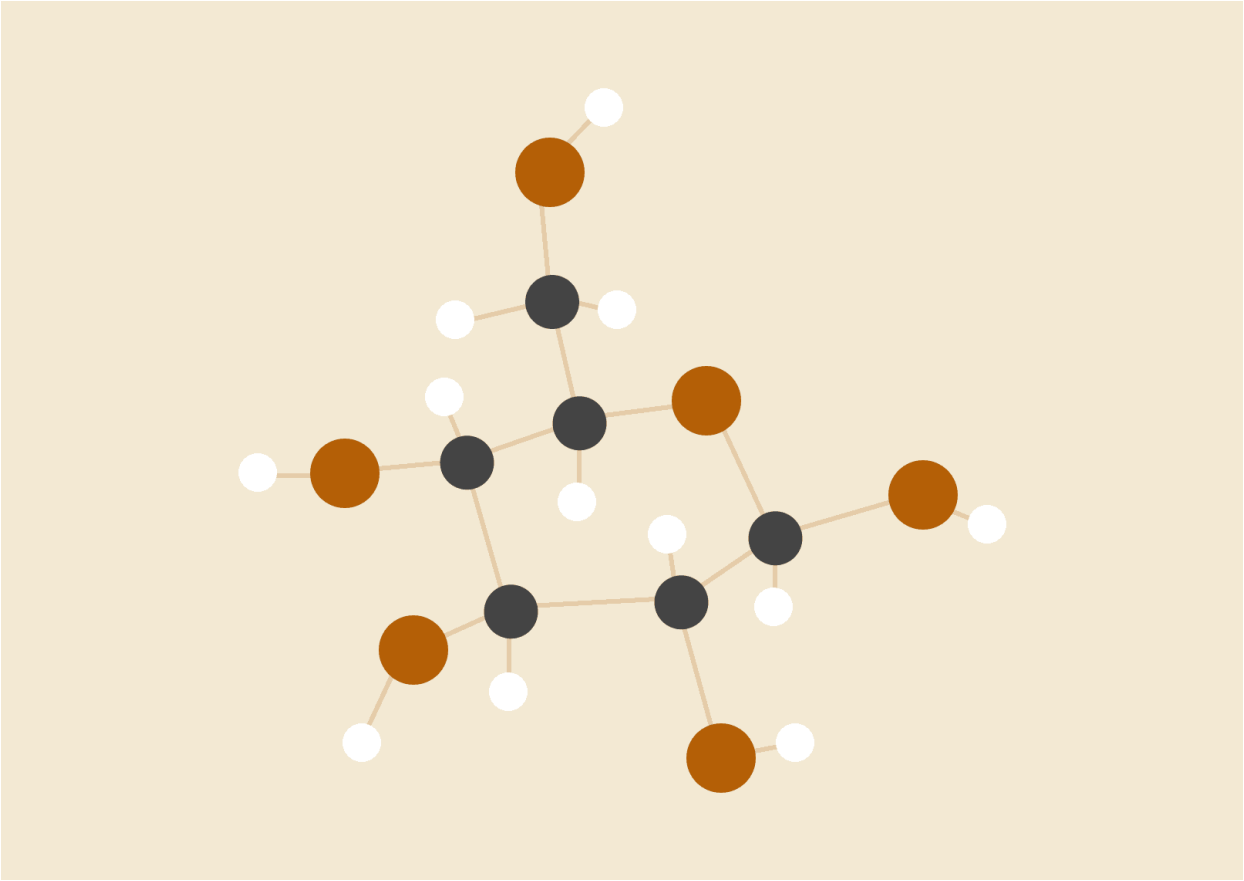
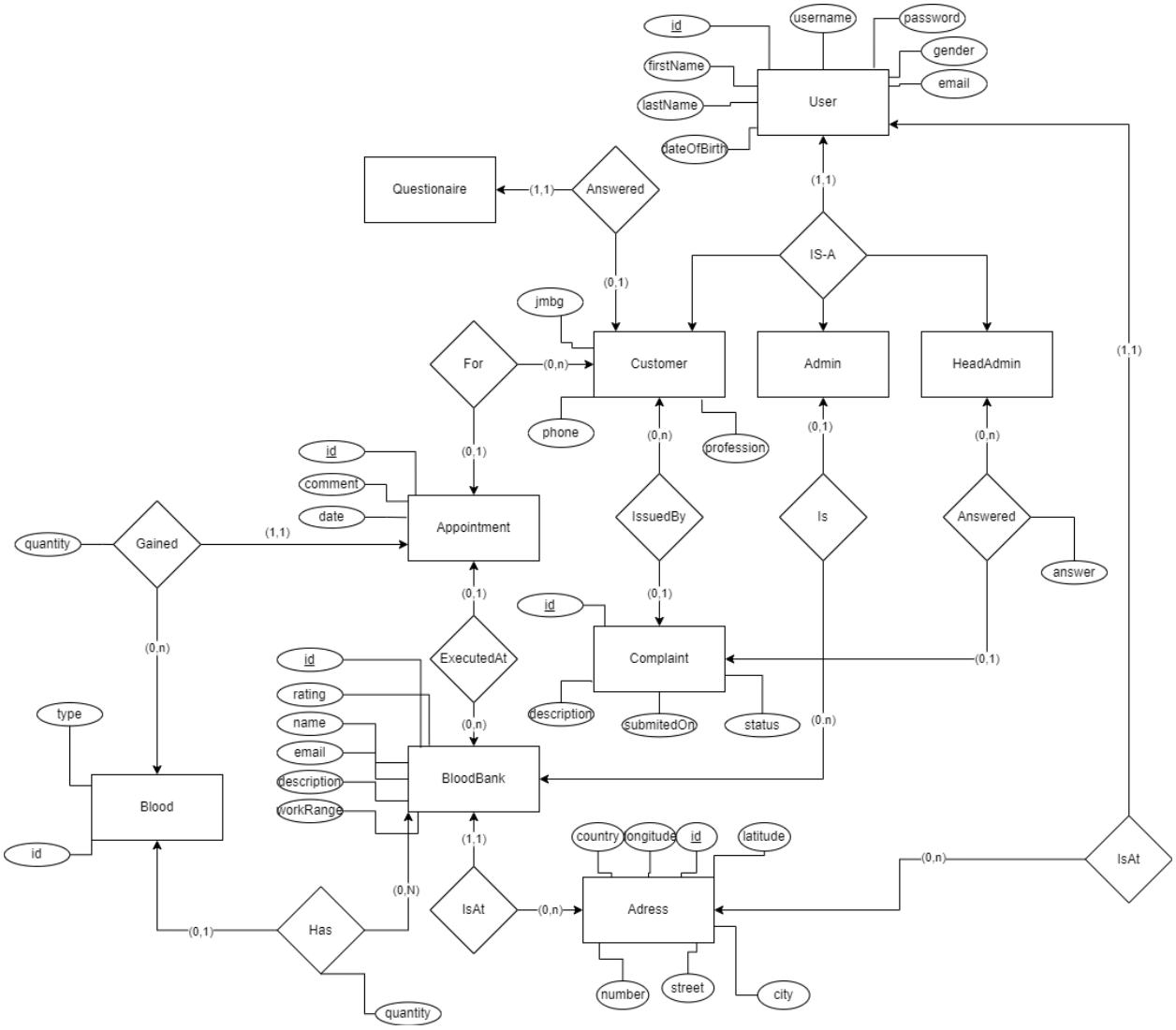
Proof of Concept (PoC) za Blood Bank System



Šema baze podataka



Strategija particionisanja podataka

Ako bismo pokušali da sve podatke skladištimo na jednom serveru vrlo brzo bi dostigli maksimum što bi ta mašina mogla da podrži.Zbog toga je potrebno vršiti **vertikalno i horizontalno particionisanje podataka**.Tabele za korisnike potrebno je horizontalnim particionisanjem podeliti na više mašina.Razlog zašto horizontalno particionišemo podatke korisnika jeste zato što imamo veliku količinu trenutnih korisnika i radi lakše skalabilnosti da mozemo da imamo i dosta više.Complaint-ove kao i adrese korisnika bi se nalazile na istim

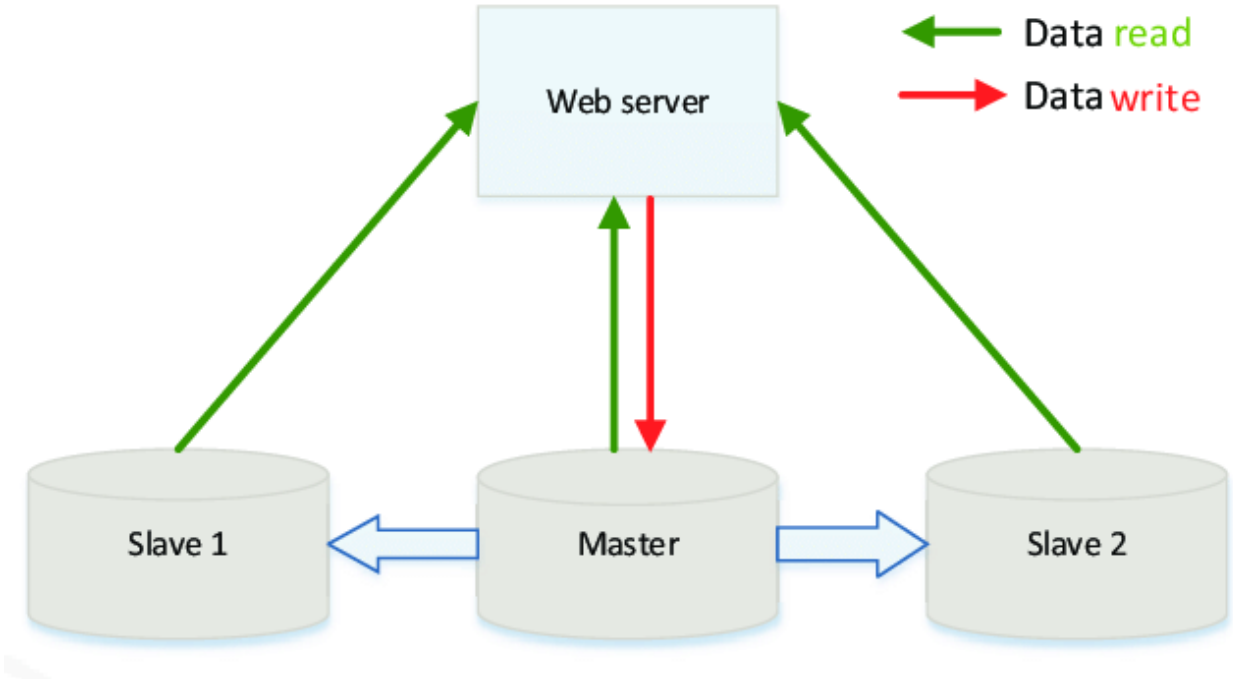
mašinama kao korisnici koji us u vezi sa njima.

Osim korisnika potrebno je horizontalno particionisati i appointmente.Appointmenti su nam core deo aplikacije i veoma nam je važno da taj deo aplikacije je skalabilan.Appointmente bi particionisali na buduće zakazane appointmente i te bi stavili na bolje mašine(**vertikalno skaliranje**), jer se mogu menjati i češće se pristupaju, dok bi prošle appointmente tj. Koji su završeni čuvali na sporijim mašinama jer se slabo pristupaju i koriste se obično samo radi statistike.

Kada bi želeli da pristupimo podacima prosledili bi id entiteta hash funkciji koja bi nam rekla na kojoj mašini se nalaze podaci datog entiteta.

Strategija za replikaciju baze i obezbeđenje otpornosti na greške

Uvodimo replikaciju baza da bi posedovali redudantnost u slučaju **katastrofalnog i permanetnog gubitka podataka**.Koristili bi sistem **master i slave baze**.Pisanje bi vršili samo u master bazi dok bi kopije podataka nalayile na slave bazama.U slučaju da se desi fail na master bazi jedna od slave baza pi tog trenutka postala master.Čitanje bi se vršilo sa master i slave baza radi raspodele zahteva.



Strategija za keširanje podataka

Za keširanje **slika,html-a i JavaScript fajlova** koji se često preuzimaju prilikom prikaza web stranice često koristili bi **CDN(Content Delivery Network).**Korisno nam je radi brzog učitavanja stranica i pošto su CDN-ovi rasprostranjeni po svetu korisniku, nezavisno od geolokacije, brzo se učitavaju webstranice.

Osim statičkih fajlova keširanje bi vršili nad **često pristupanim entitetima** u nasim bazama kao što su banke krvi koje postoje, adrese i appointmente.Banke krvi i adrese su entiteti koji se retko menjaju u našem sistemu ali se često prikazu pa zbog tog razloga ćemo keširati npr. Prvi page banki/adresa koji dobavimo iz baze kao i neke banke koje se dobijaju pretragom po čestim parametrima.

Koristimo **Read Write** strategiju keširanja jer nedostatak ili trenutna neažurnost keširanih podataka neće dovesti do pogrešnih ponašanja u sistemu pa možemo dozvoliti tu neažurnost.

Ukoliko se keš napuni, dobra strategija za izbacivanje elemenata iz keša bi bila **LRU - Last Recently Used**,kojim se izbacuju najstariji dobavljeni podaci.

Keširanje je u aplikaciji implementirano pomoć **EhCache** i implementirano je gore navedena strategija keširanja.

Procena hardverskih resursa za skladištenje

Pretpostavke sistema:

- Ukupan broj korisnika: 10 Miliona
- 5% korisnika su Admini banki, 95% Customer-i
- Broj rezervacija svih entiteta na mesečnom nivou: 500000
- Sistem mora biti skalabilan i visoko dostupan
- Customer-i sadrže jedan Questionaire, i u proseku ostave 0.5 Complaint-a godišnje
- Customer u proseku zakaže appointment jednom godišnje
- Svaki Admin kreira 1 slobodan Appointment dnevno
- 255 karaktera UTF-8 je maksimalna veličina stringova(2040B)

Zauzetost memorije za entitete koji postoje u sistemu:

- User: **1.24KB**
- Customer: **2.02KB (Questionaire uračunat)**
- Admin: **1.25KB**
- Appointment:**0.25KB**

- BloodBank:**0.84KB(Uračunato krv koju poseduje)**
- Blood:**2B**
- Complaint:**0.5KB**

Za 10 miliona korisnika potreba memorija iznosi:**19GB**

Za procenjenih 30 000 000 entiteta(500 000 mesečno) potrebno je **174GB**

Za 9.5 miliona korisnika godišnje za Appointment-e je potrebno **2.26GB**

Za 9.5 miliona korisnika godišnje za Complaint je potrebno **2.24GB**

Za kreiranje godišnje 10 000 novih banki krvi i gde svaka poseduje 3 Admin-a potrebno nam je : **0.21GB**

Ukupno potrebno memorije za održavanje u periodu od 5 godina: 200GB

Procene su rađene po principu da imamo viška memorije, kako bi izbegli da potcenimo koliko nam je potrebno resursa

Strategija load balansiranja

Algoritam koji bi bio korišćen radi ravnomerne raspodele zahteva korisnika i time povećali brzinu reagovanja našeg proizvoda jeste **Weighted Response Time**. Algoritam funkcioniše tako što uproseči response time svakog servera, i sa kombinaciom broja konekcija koje su uspostavljene sa serverom, algoritam određuje kojem se serveru šalje zahtev.Tim algoritmom dobijamo nabrzi odziv za korisnika.

Pošto koristimo autentifikaciju pomoću **JWT tokena** koji je stateless nema potrebe razmišljati čuvati sesiju kao što imamo problem kod **cookie-a** gde je cookie vezan za sesiju.Time **eliminišemo deljenu bazu** u kojoj bi čuvali cookie koji korisnici poseduju.

Operacija za nadgledanje za poboljšanje sistema

Predloge za poboljšanje našeg sistema mozemo dobiti pomsatranjem akcija koje vrše korisnici.Prikupljanjem relevatnih akcija mozemo dobiti uvid koje delove našeg sistema bi mogli nadograditi.

Najznačajniji deo koji treba nadgledati jeste akcije vezane za **zakazivanje appointment-a**.Od gledanja **količine poziva pretrage**(korisno radi informacije da li nam je algoritam za pretragu ili front kvalitetan),**Vremenske Periode pretrage za appointment-e**(informaciju koju mozemo dati bankama krvi kada je najudarnije) i druge akcije.

Takođe mozemo da nadgledamo u kojim periodima najviše posecuju korisnici sajt i prikladno s time da znamo periode kada možemo da vršimo maintenance nad opremom ili da deploy-ujemo novu verziju našeg softvera i time povećali dostupnost naše aplikacije.

Komletan crtež dizajna predloženje arhitekture