

Proof of Concept

1. Dizajn šeme baze posataka

2. Predlog strategije za particionisanje podataka

Jedan od osnovnih ciljeva prilikom particionisanja podataka jeste da ubrzao i olakša pristup podacima. Sama činjenica da će ogroman broj korisnika koristiti sistem navodi nas na ideju da particionisemo jednu na veći broj baza. U skladu sa tim treba osmisliti najbolji način za particionisanje i podataka u bazi. Smatramo da jedan od naćna particionisanja može biti particionisanje po funkcionalnostima, pa bismo u jednoj particiji mogli ćuvati podatke o pacijentima, u drugoj o zaposlenim korisnicima aplikacije, dok možemo napraviti posebnu particiju za preglede koji su održani u sklopu te bolnice. Drugi predlog je particionisanje po vremenu, pa tako možemo razdvojiti preglede u zavisnosti od meseca ili godine u kojoj su bili održani ili će biti održani. Jos jedan od pristupa bi mogao biti i particionisanje pregleda po bolnicama u kojima su održani, odnosno zaposlenih u bolnicama u kojima rade.

3. Predlog strategije za repilikaciju baze i obezbećivanje otpornosti na greske

Za implementaciju smo koristili PostgreSQL bazu podataka, koja kao sastavni deo ima repilikaciju. PostgreSQL se deli na “master” i “slave” bazu. Prilikom rada, svi podaci se prvo upisuju na master, a zatim i na sporednu bazu. Na ovaj način je ocuvana otpornost na greske. Jer prilikom prestanka rada master baze, rad se nastavlja neometano korićenjem sporedne baze. Sporedna baza takode ima znaćajnu ulogu prilikom rasterećavanja saobraćaja na glavnoj bazi, jer prilikom preopterećavanja master baze, podaci se mogu ćitati i sa slave baze.

4. Predlog strategije za kećiranje podataka

Posto je osnovni cilj kećiranja podataka poboljšanje brzine pristupa i korićenja podataka, naš predlog kećiranja se zasniva na prvenstveno na odrećivanju najćeće korićtenih podataka. Vodeći se logikom da će se pregledima koji su se obavili davno ili korisnicima koji su posećivali apoteku davno reće pristupati, koristili bismo se strategijama “least recently used” i “least frequently used” ćime bismo kećirali podatke novijih datuma i podatke koji će najćeće koriste. Samim tim što je keć ogranićenog kapaciteta, prilikom novih podataka, najstariji podaci će biti izbacivani.

5. Okvirna procena za hardverske resurse potrebne za skladištenje svih podataka u narednih 5 godina

Vođeni pretpostavkom da aplikacija ima dvesta milijona korisnika i da je broj rezervacija lekova i zakazanih pregleda kod farmaceuta i dermatologa na mesečnom nivou milion dolazimo do računice da će nam za skladištenje biti neophodno oko 42GB memorije. Posmatrajući situaciju iz šireg aspekta, najviše memorije zauzimaju pregledi, gde pregled zauzimati oko 250 byte-ova, dakle kada to pomnogžimo sa milion pregleda po mesecu i 5 godina dobijamo podatak da će pregledi zauzimati oko 15 GB samo za preglede. Dok podaci o korisnicima aplikacije otprilike zauzimaju 200 byte-ova po osobi. Dok je za skladištenje podataka o apoteka koji je približno 50 i podacima o lekovima i dijagnozama koji se u njima nalaze približno 7GB.

6. Predlog strategije za postavljanje load balansera

Load balanserom ćemo omogućiti ravnomeran raspored zahteva koji pristižu od strane korisnika, kako ni jedan server ne bi bio preopterećen, a zahtevi korisnika bili izvršavani maksimalnom brzinom. Jos jedna velika prednost ove strategije ogleda se u činjenici da prilikom otkazivanja jendog od servera load balancer preusmerava zahteve na neki od dostupnih servera. U ovom slučaju predlažemo korišćenje “round robin” tehnike.

7. Predlog koje operacije korisnika treba nadgledati u cilju poboljšanja sistema

Poboljšanje aplikacije bi se moglo ostvariti pomoću “paging”-a, jer bismo na ovaj nacin ubrzali dobavljanje podataka tako sto svi podaci ne bi bili dobavljeni istovremeno, već bismo dobavljali samo određenu količinu podataka. Uviđanje i rešavanje konfliktnih situacija do kojih bi najčeće moglo doći bi takođe značajno poboljšalo rad sistema, kao što su operacije za rezervisanje pregleda.

8. Kompletan crtež dizajna predložene arhitekture