Scalability - Proof of Concept

Team 54

1. Dizajn šeme baze podataka

Dizajn šeme baze podataka je dat kroz UML Class Dijagram koji je okačen u ovom GitHub repozitorijumu u folderu “Class Diagram”. Korišćen je objektno-relacioni maper koji prevodi objekte u entitete baze podataka.

2. Predlog strategije za particionisanje podataka

Jedan od osnovnih ciljeva prilikom particionisanja podataka jeste da ubrza i olaksa pristup podacima. Sama cinjenica da ce ogroman broj korisnika koristiti sistem navodi nas na ideju da particionisemo jednu na veci broj baza. U skladu sa tim treba osmilisti najbolji nacin za particionisanje i podataka u bazi. Smatramo da jedan od naicna particionisanja moze biti particionisanje po funkcionalnostima, pa bismo u jednoj particiji mogli cuvati podatke o pacijentima, u drugoj o zaposlenim korisnicima aplikacije, dok možemo napraviti posebnu particiju za preglede koji su odrzani u sklopu te bolnice. Drugi predlog je particionisanje po vremenu, pa tako mozemo 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 obezbeđivanje otpornosti na greske

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

4. Predlog strategije za keširanje podataka

Posto je osnovni cilj keširanja podataka poboljšanje brzine pristupa i korišćenja podataka, naš predlog keširanja se zasniva na prvenstveno na određivanju najčešće korištenih podataka. Vodeći se logikom da će se pregledima koji su se obavili davno ili koriscnicima koji su posećivali apoteku davno ređe pristupati, koristili bismo se strategijama “least recently used” i “ least frequently used” čime bismo keširali podatke novijih datuma i podatke koji ce najčešće koriste. Samim tim što je keš ograničenog kapaciteta, prilivom novih podataka, najstariji podaci će biti izbacivani.

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

Vođeni pretpostavkom da aplikacija ima dvesta milijona korisnika i da je broj rezervaijca lekova i zakazanijh pregleda kod farmaceuta i dermatologa na mesečnom nivou milion dolazimo do računice da če nam za skladištenje biti neophodno oko 42GB memorije. Posmatrajući situaciju iz šireg aspekta, najvise 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 aplikaicije 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 alikacije 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