PROOF OF CONCEPT

Skalabilnost je važna u izradi veb aplikacije. Postoji više opcija za skaliranje nivoa veb aplikacije i sloja baze podataka:

- Vertikalno skaliranje
- Horizontalno skaliranje

VERTIKALNO SKALIRANJE

Predstavlja unapređivanje komponenti jednog servera (više RAM-a, jači CPU...) ili nabavljanje jačeg servera. To je najlakši i najbrži način za skaliranje veb aplikacije. Zahteva samo premeštanje sadržaja veb aplikacije na novi server, bez promene izvornog koda. Međutim, nijedan server nema beskonačno RAM-a i beskonačno procesorske moći. Zato koristimo horizontalno skaliranje.

HORIZONTALNO SKALIRANJE

Dok se vertikalno skaliranje fokusira na pojačanje jednog servera, horizontalno se fokusira na povezivanje više njih. Na ovaj način možemo imati mnogo više RAM-a i CPU-a, ne na jednoj mašini, već na klasteru. Isto kao što procesor radi na više niti, tako i naša aplikacija radi na više servera.

Ovako skaliranje zahteva promene na nivou arhitekture same aplikacije. Konkretno, uključivanje *Load Balancer*-a.

LOAD BALANCER

Kao što mu ime nagoveštava, *Load Balancer* se koristi za distribuciju opterećenja mrežnog saobraćaja između više instanci. Možemo koristiti više algoritama za raspoređivanje:

- *Round Robin* – sekvencijalno dodeljivanje zahteva
- *Least Connection* – zahtev odlazi serveru koji servisira najmanje zahteva u datom trenutku
- *Chained Failover* – prebacuje zahtev na sledeći server samo ako prethodni ne može da ga obradi
- *Weighted Response Time* – prebacuje na server sa najkraćim vremenom obrade zahteva

Šta ako zahtev traje predugo jer pokušava da pristupi slikama, video zapisima ili bilo kom statičkom sadržaju na serveru koji je predaleko od korisnika? Rešenje ovog problema je u korišćenju *Content Delivery Network*-a.

CONTENT DELIVERY NETWORK

CDN-ovi sluše za brže dobavljanje statičkih fajlova. Nalaze se na više lokacija širom sveta tako da je moguće dobiti onaj koji je bliži lokaciji korisnika nego što su naši serveri. CDN može da kešira podatke i tako još više smanji vreme dobavljanja. Sem keširanja statičkih, biće nam potreban i neki oblik keširanja dinamičkih podataka.

MIKROSERVISI I KONTEJNERI

Mikroservisi su pristup razvoju jedne aplikacije kao skupa malih servisa, umesto jedne ogromne aplikacije. Svaki servis radi u svom sopstvenom procesu, umesto da se oslanja na jedan proces da pokrene celu aplikaciju. Ove mikroservisi komuniciraju *lightweight* mehanizmima. Pošto su oni mali delovi, ne treba im cela virtuelna mašina da bi radili. Umesto toga, oni mogu da rade na kontejneru. Kontejner je još jedan vid virtualizacije OS-a. Za razliku od VM, on ima samo minimalne resurse potrebne za pokretanje aplikacije. Ovo dovodi do male težine slike. Dakle, skalabilniji je od VM.

KEŠIRANJE I INDEKSIRANJE BAZE PODATAKA

Kada veliki broj SQL zahteva za bazu podataka daje isti rezultat, onda je bolje da se ti podaci keširaju u memoriju da bi se obezbedio brži pristup podacima i smanjilo opterećenje baze podataka. Tipični slučaj je top 10 proizvoda prikazanih na početnoj stranici (a oni su isti za sve korisnike).

Ako svakom redu u bazi pridružimo indeks, brzina dobavljanja tog reda iz baze podataka se smanjuje sa O(n) na O(1).

REPLIKACIJA BAZE PODATAKA

Možemo pisati podatke u jednu bazu podataka (READ-WRITE) a čitati iz druge (READ). Na taj način možemo smanjiti opterećenje za 50% tako što ćemo ga balansirati na 2 baze podataka, a ne samo na jednu. READ-WRITE baza preuzima odgovornost da ažurira READ bazu tako da obe imaju (skoro) iste podatke.

PARTITIONING I *SHARDING*

Partitioning predstavlja podelu tabele po vertikali. Tabelu od 20 kolona možemo podeliti na dve tabele po 10 kolona. Te dve tabele mogu biti u jednoj ili dve odvojene baze podataka.

Sharding deli tabelu na više tabela. Svaka tabela sadrži isti broj kolona, ali manje redova. Na primer, tabela korisnika može biti podeljena na 5 manjih tabela, od kojih svaka predstavlja kontinent za grupu kupaca. Poznavanje lokacije korisnika pomoći će preusmjeravanju upita na desnu particiju za obradu manje redova.