Proof of concept

Sklabilnost predstavlja mogućnost aplikacije da podnese povećanje broja korisnika i zahteva bez potrebe da se sama aplikacija mora menjati.

Rest servisi

Realizovana aplikacija nudi isključivo stateless servise, koji pogoduju skalabilnosti. Takođe autentifikacija i autorizacija realizovani u okviru aplikacije koriste JWT tokene, koji su stateless.

Relaciona baza podataka

Za veliki promet podataka relacione baze podataka ne daju baš najbolje performanse i ne mogu da održe tempo skaliranja kakav može da izdrži Spring deo aplikacije. U slučaju velikog porasta broja korisnika bilo bi poželjno preći na neku distribuiranu bazu poput Cassandra ili Redis što bi verovatno zahtevalo restrukruiranje modela podataka. Prelaskom na Cassandru dobili bismo na skalabilnosti i robusnosti.

Međutim, u trenutnom modelu su iskorišteni surogatni ključevi koji u određenoj meri poboljašavju performanse realacione baze u odnosu na upotrebu prirodnih ključeva.

Klasterovanje

Radi povećanja performansi i pouzdanosti jedan od koraka u toku sklairanja bilo bi klasterovanje servisa. Za povećanje pouzdanosti bilo bi neophodno uvesti redundante servise, tako da se izbegne Single Point of Failure problem. Smatramo da je ovakav korak neophodan, s obzirom da aplikacija rukuje sa informacijama značajnim za korisnika, koje uključuju i finansijske transakcije.

Za povećanje performansi bilo bi neophodno uvesti redundante servise i load balancer-a koji raspoređuje korisničke zahteve tako da ravnomerno optereti servise. Trenutno je upotrebljen Tomcat web server, što kao posledicu ima veliki saobraćaj. Veliki saobraćaj predstavlja problem prilikom skaliranja.

Messaging Queue

Jedan od bitnih karakteristika Messaging Queue jesu robustnost i pouzdanost, koje se postižu strategijama perzistentnosti. Pošto MQ razdavjaju procese lako je povećati brzinu kojom se poruke dodaju u red i obrađuju. U realizaciji projekta nije upotrebljen MQ mehanizam. Međutim, primena MQ mehanizma moguća je u kombinaciji sa tehnologijama koje su primenjene na projektu.

Optimističko zaključavanje

Za aktivnosti poput rezervacije karte, optimističko zaključavanje koje je implementirano predstavlja rešenje koje bolje podnosi veći broj korisnika. Optimističko zaključavanje omogućava da više korisnika rezerviše različite karte istovremeno, što smatramo da je značajno češći slučaj od pokušaja više korisnika da istovremeno rezervišu istu kartu. Pored realizacije optimistčkog zaključavanja, za sklabilnost bi bilo dobro realizovati AJAX ili WebSocket-e, tako da klijenti koji trenutno biraju mesta brže dobiju informaciju da je neko drugi zauzeo mesto. Takav pristup bi rezultovao manjim brojem uzaludno upućenih zahteva od strane klijenta ka serveru.

Implementiranje pretraga

S obzirom da je osnovna namena sistema pronalazak odgovarajuće karte, smeštaja i automobila - pretrage predstavlju jednu od osnovnih aktivnosti koju će korisnici obavljati. Rezultat pretrage je u većini slučajeva velika količina podataka, od kojih je korisniku u najčešće potreban samo jedan manji deo. Upotreba paging-a

pozitivno utiče na skalabilnost i bolje performanse. Paging je realizovan na određenom delu servisa koje aplikacija nudi. Takođe, realizacija pretraga na stateless način, gde korisnik šalje serveru informaciju o pojmu pretrage i stranici koja mu je potrebna, pozitivno utiče na skalabilnost.

Asinhroni procesi

Za aktivnosti poput slanja mejlova korisnicima (bilo nakon registracije ili rezervacije) upotrebljeni su asinhroni procesi. Takav pristup omogućava da korisnici brže dobiju odgovor nazad, te smanjuje broj nepotrebnih zahteva od strane nervoznih korisnika.

Upotreba vanjskih servisa

Jedan od prvih koraka koji bi se morali primeniti u slučaju povećanja broja korisnika je regulacija upotrebe vanjskih servisa. Da bi se aplikacija skalirala za upotrebu od strane više korisnika bilo bi neophodno izvršiti zamenu trenutno upotrebljenih besplatnih email naloga, mapa i geodecodera, baze podataka i svih ostalih vanjskih servisa sa plaćenim verzijama.

DevOps flow

Veliku pomoć pri održavanju velikih aplikacija daje uvođenje DevOps flow-a. Trenutno je podesen Codacy koji kontroliše kvalitet koda i čija je ocena istaknuta direktno na github repozitorijumu. Za pokretanje testova i automatski deploy koristi se Travis, koji neće uraditi deploy ukoliko svi testovi ne budu zadovoljeni. Ovo omogućava istovremenu upotrebu aplikacije i rad na njenom daljem razvoju. Projekte koje je lakše održavati, lakše je i skalirati.