TRENUTNA ARHITEKTURA

-Više klijenata (android, ios, winphone, WEB),

-1 monolitni poslužitelj (DRUPAL CMS, autogenerated DB model - MySQL)

-model podataka:

-entiteti: Propovijedi, Aktualno, Poziv, Odgovori, Multimedija, Duhovnost, Molitvenik, Pjesmarica, Čitanja, Izreke

-entiteti realizirani na generički način sa 2 podatkovna entiteta: Category (skupina sadržaja) i Content (sadržaj)

NOVA ARHITEKTURA

-Više klijenata (android, ios, winphone, WEB)

-po jedan mikroservis na poslužitelju za svaki entitet sa posebnom bazom podataka(Java Spring Cloud mikroservisi, PostgreSQL baza podataka)

-model podataka:

-entiteti: Propovijedi, Aktualno, Poziv, Odgovori, Multimedija, Duhovnost, Molitvenik, Pjesmarica, Kalendar, Izreke

-entiteti realizirani sa specifičnim (npr., Song, tags, chords) + generičkim modelom podataka (Content, Category)

Motivacija/Problem starog sustava (backend)/Ciljevi, rješenja novog sustava

-Drupal kod;star kod; nitko ne poznaje kod; kako sustav zapravo radi; nitko ne poznaje Drupal CMS tehnologiju

--PROBLEM

-izmjena koda je preriskantna

-sustav nije izmjenjiv

-sustav nije proširiv

--RJEŠENJE

--raspodijeljena arhitektura -> mikroservisna arhitektura sa loose coupling -> moguće što lokaliziranije izmjene ili dodavanje bez promjena

-API

-api za Category i Content; ne postoji specifični API za pojedine entitete

-star api

-nije konzistentan (semantički heterogen -> isti atributi contenta imaju različite nazive u pozivu za kategoriju i content)

-nelogičnosti u apiju

-npr., idjevi su String umjesto broj

-neizmjenjiv api (jer je sustav neizmjenjiv)

-ne postoji efektivan mehanizam verzioniranja APIja

-> izmjene u APIju nisu backwards compatible

-> promjena bilo čega u APIju zahtjeva promjenu svih klijenata (PRAKTIČNO neizvedivo, neki korisnici koriste stare verzije klijenata) -> api je zacementiran

--NUŽAN Zahtjev Izmjenjivost APIja (dodavanje entiteta, izmjena entiteta)

--NUŽAN Zahtjev Izmjenjivost APIja (dodavanje entiteta, izmjena entiteta) bez izmjene klijenata

--tj., uz backwards compatibility

--RJEŠENJE

-1) Generičan api (dopušta dodavanje novih funkcionalnosti (npr., nova vrsta dokumenata) u postojeću verziju APIja bez izmjene klijenata)

-2) Verzioniranje APIja

--bilo koja izmjena APIja koja zahtjeva promjenu klijenta je zapravo potpuno novi API (tj., nova verzija apija)

--ZAHTJEV NOVO : podrška za keširanje na klijentu

--RJEŠENJE

--Poseban dio APIja (/changes) koji vraća promjene u kategoriji-regiji od zadanog datuma

--IZMJENA: paginacijski model apija

--stari API paginaciju vrši dohvatom određenog broja vijesti novijih od zadanog datuma

--novi API uvodi standardni paginacijski model (page=number, size=number)

-sigurnost

-nema autentifikacije za pristup podacima

-nesiguran pristup podacima preko HTTPa

-ispad jedne komponente <=> ispad sustava

-NUŽAN ZAHTJEV - ispad jedne komponente je ispad samo te komponente

RJEŠENJE

--mikroservisna arhitektura sa DB per service patternom

+ne postoji single breaking point u sustavu -> ispad jednog entiteta ne vodi do ispada ostalih entiteta i pada sustava

--API GATEWAY (ZUUL) sa HTTPS i OAUTH (ili BA) autentifikacijom za siguran i autoriziran pristup sustavu

--Discovery Server (Eureka) za mikroservise -> mikroservisi se registriraju na njega

+mikroservisi ne moraju impl. HTTPS i autentifikaciju a pristup njima će biti osiguran HTTPSom i autentifikacijom preko API-GATEWAYa

-skaliranje

-ne postoji mehanizam

-NUŽAN ZAHTJEV - mehanizam skaliranja

-NUŽAN ZAHTJEV - zasebno skaliranje pojedinih entiteta u odnosu na broj zahtjeva

+optimalno iskorištavanje resursa servera

RJEŠENJE

--mikroservisi, DB per service -> moguće zasebno skaliranje mikroservisa

-Bug u broju kategorija

-sustav ima bug, nije moguće dodavati više od 1000 kategorija

-bug nije popravljiv na starom sustavu jer je sustav neizmjenjiv

-zahtjevi za novim funkcionalnostima i izmjenama postojećih funkcionalnosti

PROBLEM

-neizmjenjiv sustav

-nemogućnost dodavanja novih funkcionalnosti bez mijenjanja postojećeg sustava

--pjesmarica - NOVO: tagovi i akordi

--RJEŠENJE

--nove funkcionalnosti u entitetu (mikroservisu) pjesmarica

--kalendar - REDIZAJN (NOVO + STARO): čitanja, svetac dana, časoslov - novi,stabilniji podaci, nov način pristupa

--jako kompleksno podatkovno, logički, prava na sadržaj

--RJEŠENJE

--ovaj komad sustava je outsorcean (poseban razvojni tim, poseban server, poseban api)

--Jednim plugin mikroservisom se može integrirati taj komad sustava

--Radio streaming: NOVO

--ovo je planirana buduća funkcionalnosti

--ovaj dio sustava je outsorcean

--NUŽAN ZAHTJEV: mora se moći integrirati u sustav plugin mikroservisom

--NOVO: proširenje na druge države s posebnim kanalima za svaku državu

--RJEŠENJE

--uvode se kanali/regije sadržaja

--uvode se ADMINISTRATORSKI korisnici za svaku regiju sa pravima upravljanja sadržajem samo jedne regije

--IZMJENA: prebacivanje Molitvenika sa klijenta na poslužitelj

--PROBLEM

--trenutno je čitav molitvenik na klijentu

--sadržaj molitava nije izmjenjiv (ne mogu se dodavati nove, brisati stare,..)

--RJEŠENJE

--transformirat model molitava u generički Content model

--zaseban entitet (mikroservis) za molitve (CMS)

-centraliziran sustav

-monolitni poslužitelj i baza

-svi podaci za sve entitete su u 2 glavne tablice (kategorija i content)

-trebalo bi bit sporo (nije jer Drupal to dobro rješava)

-single breaking point za aplikaciju

RJEŠENJE

--svaki entitet -> zaseban mikroservis i baza/schema prema DB per service patternu

+brže (podaci raspodijeljeni po tablicama)

+nema single breaking point

+uvesti keširanje na poslužitelju (radi brzine)

-**potpuno generički podatkovni model za sve entitete**

-npr., pjesme i čitanja

-nemogućnoćnost specijaliziranih funkcionalnosti vezanih za samo jedan entitet

--npr., tagovi

-svi podaci su 2 tablice

RJEŠENJE

--svaki entitet zaseban mikroservis -> DB per service

+omogućuje lokalizirano dodavanja/izmjenu funkcionalnosti (specijaliziranih za pojedine entitete)

-**nesklad: prostorno/vremenski distribuiran razvoj <-> centraliziran i zacementiran (neizmjenjiv) sustav**

-tim je vremenski i prostorni distribuiran i često se mijenja (za 10g na sustavu neće radit nitko od sadašnjih ljudi i neće poznavati kod)

-novi kod će očekivano opet biti nekome nerazumljiv i neizmjenjiv kroz vrijeme

-NUŽAN ZAHTJEV Mogućnost dodavanja novih entiteta i pripadajućih funkcionalnosti bez izmjene bilo čega postojećeg u sustavu (kod, konfiguracija,...)

+Heterogen tim -> mogućnost korištenja bilo koje tehnologije

-NUŽAN ZAHTJEV Mogućnost dodavanja novih funkcionalnosti postojećim entitetima bez izmjene jezgre sustava (rušenje sustava), bez izmjene postojećih funkcionalnosti postojećih

entiteta

-NUŽAN ZAHTJEV Mogućnost izmjene postojećih funkcionalnosti postojećih entiteta

-NUŽAN ZAHTJEV Specijalizacija entiteta: Mogućnost dodavanja i izmjene funkcionalnosti pojedinom entitetu, bez mijenjanja ostalih entiteta i jezgre sustava

+što lakše dodavanje entiteta i funkcionalnosti bez proučavanja postojećeg koda

+što sigurnije dodavanje funkcionalnosti i entiteta, bez izmjene postojećeg koda, posebno jezgre (CMS) (rušenje sustava)

+Što lakše izmjene entiteta i postojećih funkcionalnosti uz minimalno proučavanja postojećeg koda i minimalne izmjene postojećeg koda

+Što sigurnije izmjene entiteta i postojećih funkcionalnosti -> lokalizirane izmjene -> minimalne izmjene postojećeg koda

RJEŠENJE

--mikroservisna arhitektura; plug in arhitektura za mikroservise

--svaki entitet je zaseban mikroservis

--jezgra sustava (CM) je enkapsulirana na jednom mjestu (npr., library)

--načelno neizmjenjiva -> mora biti napravljena dosta generički (multimedia)

DODAVANJE entiteta

--dodavanje novog mikroservisa

DODAVANJE funkcionalnosti entiteta

--dodavanje funkcionalnosti postojećem mikroservisu dodavanjem novog koda mikroservisu bez izmjene postojećeg koda tog mikroservisa

IZMJENA postojeće funkcionalnosti entiteta

--lokalizirana (samo kod te funkcionalnosti) izmjena postojećeg koda mikroservisa

--ili dodavanjem novog koda mikroservisu bez izmjene postojećeg koda tog mikroservisa (u slučaju nove verzije apija)

-prava pristupa:

--READER - ovlasti nad dohvatom sadržaja

--Administratorski korisnici (izmjena sadržaja):

--FullAdmin

--programer, potpuni pristup sustavu, upravljanje konfiguracijom mikroservisa, direktni pristup evictu keša,...;direktni pristup bazi,...

--SystemAdmin

--Šibalić; stvara administratore regija i System WRITERe;

--ovlasti nad svim sadržajem svih regija

--SystemModerator

--službeni moderatorski tim NoveEve; ovlasti nad svim sadržajem svih regija

--RegionAdmin

--stvara Region moderatore; ovlasti nad tagovima pjesama te regije

--RegionModerator

--objavljivači sadržaja pojedine regije; ovlasti nad svim sadržajem te regije

+potrebno novo ADMINISTRATORSKO sučelje za upravljanje sadržajem

-outsorceati mlađem studentu

-obrazovni ciljevi

-mikroservisna arhitektura, distr. razvoj, agilni razvoj, naručitelj, tim