
Vježba: 10 – Zadaća 3. Web sustav "Informacije o aerodromima, lokacijama i meteorološkim podacima"

Naziv projekta: **{LDAP_korisnik}-zadaca_3**

Korijenski direktorij treba biti **{LDAP_korisnik}-zadaca_3**

Sve nove klase trebaju biti u paketu **org.foi.nwtis.{LDAP_korisnik}.zadaca_3**. Za rad s postavkama treba koristiti Java biblioteke iz {LDAP_korisnik}-zadaca_3_lib_03_1 (nastao na temelju {LDAP_korisnik}-vjezba_03_1) i {LDAP_korisnik}-zadaca_3_lib_06_1 (nastao na temelju {LDAP_korisnik}-vjezba_06_1). **Projekt se isključivo treba predati u formatu Eclipse IDE projekta s maven upravljanjem.** Prije predavanja projekta potrebno je **napraviti Clean na projektu** (i svim pomoćnim projektima). Zatim cijeli projekt (i sve pomoćne projekte) sažeti u **.zip** (NE .rar) format s nazivom **{LDAP_korisnik}-zadaca_3.zip** i predati u Moodle. Uključiti izvorni kod, primjere datoteka konfiguracijskih podataka (.txt, .xml, .bin, .json) (na WEB-INF) i popunjeni obrazac za zadaću pod nazivom **{LDAP_korisnik}-zadaca_3.pdf** (u korijenskom direktoriju projekta).

Struktura .zip datoteke predane zadaće treba biti sljedeća:

```
{LDAP_korisnik}-zadaca_3
	{LDAP_korisnik}-zadaca_3.pdf
	{LDAP_korisnik}-zadaca_3_wa_1
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_3_wa_2
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_3_lib_03_1
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_3_lib_06_1
		pom.xml
		src
			main
				...
```

Nazivi klasa, nazivi atributa, nazivi metode, nazivi varijabli, komentari i sl. pišu se na hrvatskom jeziku u skladu s preporukama za Java jezik. Metode u klasama NE smiju imati više **od 35 linija programskog koda**, u što se ne broji definiranje metode, njenih argumenata i lokalnih varijabli, prazna linija, linija samo s { ili }. U jednoj liniji može biti jedna instrukcija. Linija ne može ima više od **120 znaka**. Uvlačenje je **4 znaka**.

Klase i metode trebaju biti dokumentirane u **Javadoc** formatu.

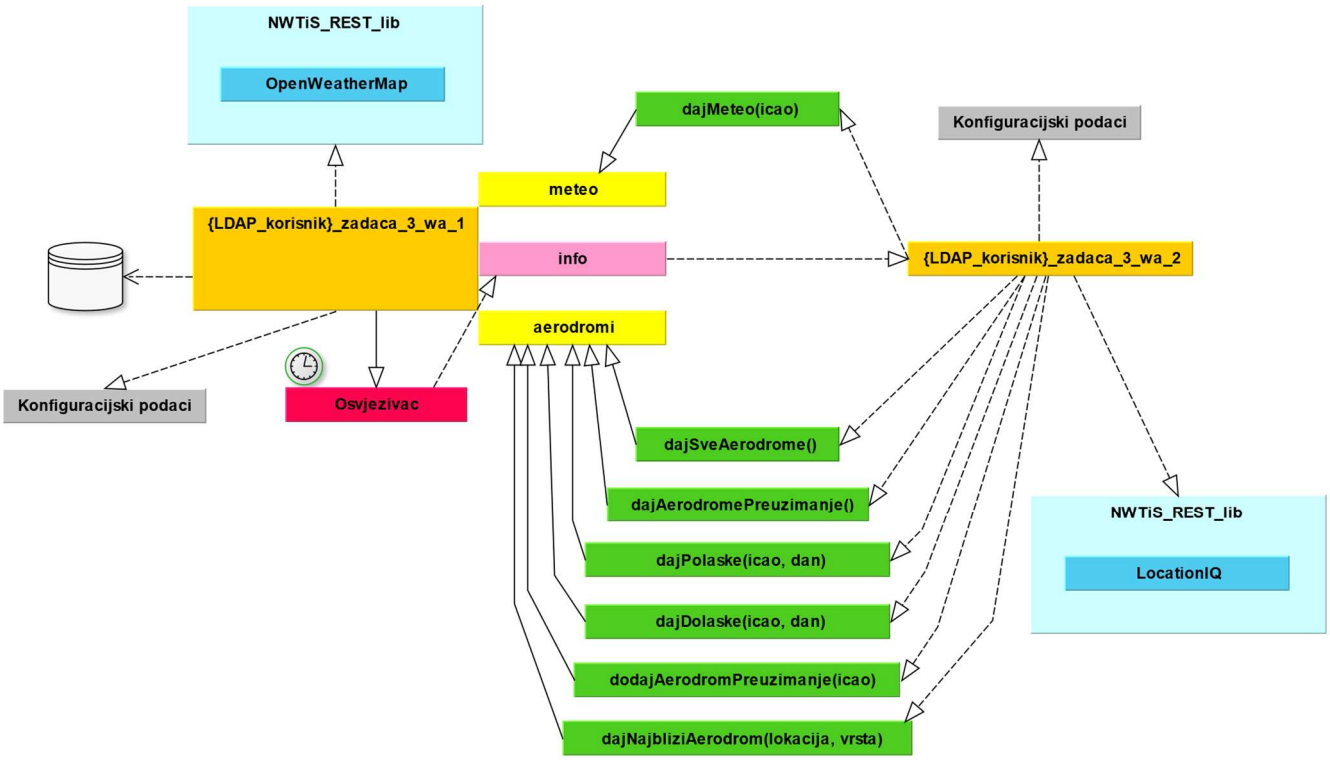
U projektu **ne smiju se koristiti klase i/ili metode koje su zastarjele** tj. anotirane su kao @Deprecated. Na Properties projekta u Java Compiler/Errors/Warnings kod Deprecated and restriced API **obavezno treba postaviti/označiti** *Signal use of depreated API inside deprecated code* i *Signal overrriding or implementing deprecated method*.

Pripazite da nazivi postavki, nazivi glavnih klasa, komande zahtjeva i odgovori na zahtjeve budu u točnom obliku kako je definirano u opisu zadaće (velika, mala slova, razmak, točka, ...)..

Boduju se dijelovi koji su rađeni nakon vježbi!

Opis rada sustava:

Sustav se sastoji od dvije web aplikacije od kojih prva (instalirana na Glassfish EE Server) pruža JAXWS (SOAP) servis, šalje zahtjeve servisu OpenWeatherMap i pruža WebSocket krajnju točku. Druga web aplikacija (instalirana na Glassfish EE Server) usmjerena je na korisničko sučelje čije podatke puni tako da šalje JAXWS pozive servisu iz prve web aplikacije i šalje zahtjeve servisu LocationIQ. Shema sustava prikazana je na slici 1.



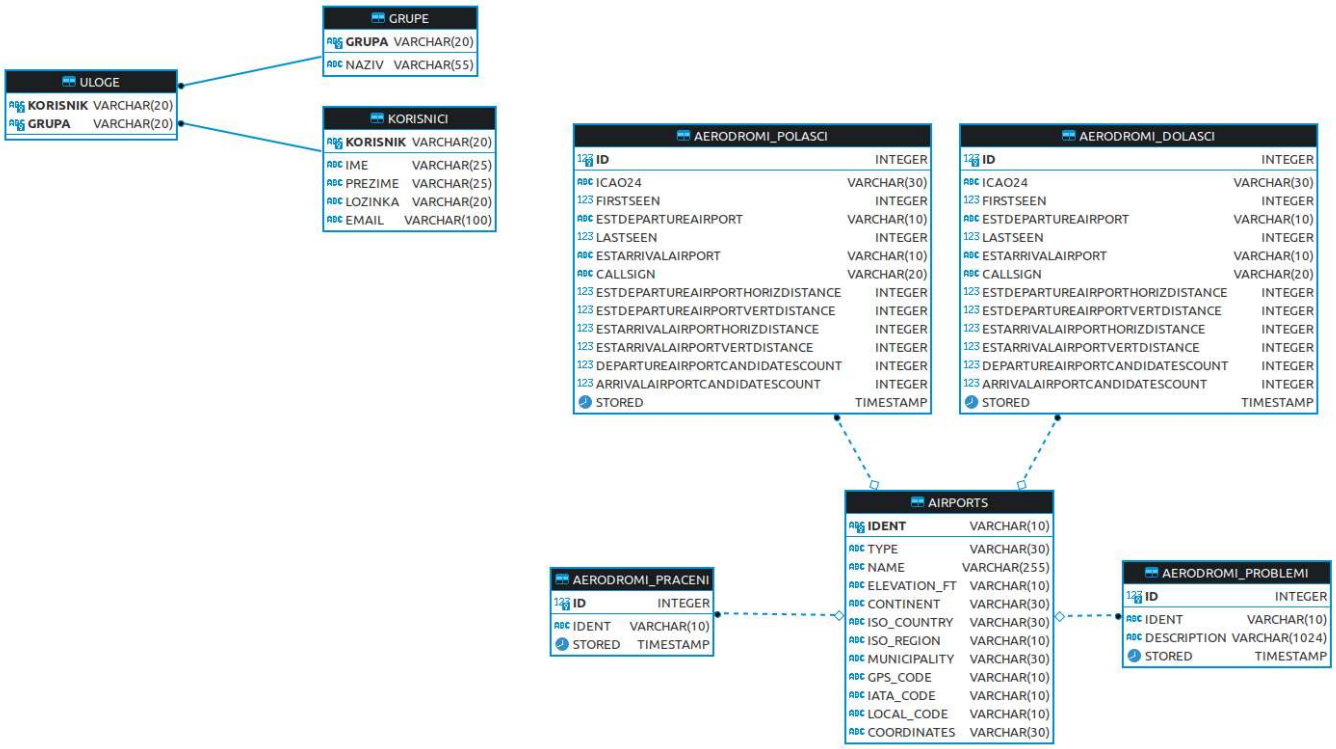
Slika 1. Shema sustava

Prvo je potrebno pokrenuti poslužiteljsku web aplikaciju `{LDAP_korisnik}-zadaca_3_wa_1`. Nakon toga može se pokrenuti korisnička web aplikacija `{LDAP_korisnik}-zadaca_3_wa_2`.

Web aplikacija `{LDAP_korisnik}-zadaca_3_wa_1` učitava konfiguracijske podatke putem slušača aplikacije kod pokretanje aplikacije i upisuje ih u atribut konteksta pod nazivom „Postavke". Naziv datoteke konfiguracijskih podataka zapisan je u web.xml kao inicijalni parametar konteksta „konfiguracija" (jedna od datoteka NWTiS.db.config_1.xml ili NWTiS.db.config_2.xml). Nakon učitavanja konfiguracijskih podataka potrebno je pokrenuti dretvu Osvjezivac. Aplikacija pruža JAXWS (SOAP) servise za aerodrome i meteorološke podatke. Njihove krajnje točke su „aerodromi" (klasa WsAerodromi) i „meteo" (klasa (WsMeteo). Aplikacija pruža krajnju točku za WebSocket pod nazivom „info". Njena uloga je da šalje informacije (opisano kod klase Osvježivac) i da prima poruke kojima se aktivira slanje informacije.

Klasa Osvjezivac je dretva koja radi u ciklusima identičnog trajanja spavanja. Zadano je spavanja za ciklus (postavka „ciklus.spavanje", u sekundama). Dretva ima zadatak da u svakom ciklusu pošalje odvojene zarezom trenutno vrijeme na poslužitelju prve aplikacije (u formatu dd.mm.gggg hh:mm:ss) i broj aerodroma koji se prate. Podaci se šalju kao poruka putem WebSocketa na krajnjoj točki „info" svim prijavljenim korisnicima (tj. sjednicama).

Vježba 10./zadaća 3. pretpostavlja da su u prethodnoj vježbi 9./zadaći 2. prikupljeni podaci u tablicama „AERODROMI_PRACENI". „AERODROMI_POLASCI" i „AERODROMI_DOLASCI". Dijagram baze podataka prikazan je na slici 2.



Slika 2. Dijagram baze podataka

Prvi JAXWS servis bavi se aerodromima. Četiri metode web servisa dohvaćaju podatke iz jedne od tablica uz primjenu filtriranja podataka na bazi primljenog/ih argumenta/anata. Argument dan je u formatu dd.mm.gggg. Metoda dodajAerodromPreuzimanje(icao) dodaje aerodrom u tablicu „AERODROMI_PRACENI". Metoda dajNajbliziAerodrom(lokacija, vrsta) traži aerodrom (vrsta: true - samo koji se prate, vrsta: false - svi aerodromi) koji je najbliži GPS lokaciji koja je zadana u argumentu. Potrebno je proći zadani skup aerodroma i za svaki od njih na temelju GPS lokacije aerodroma i GPS lokacije koja je zadana u argumentu izračunati udaljenost. Algoritam treba uzeti u obzir zakrivljenost Zemlje.

Drugi JAXWS servis bavi se meteorološkim podacima. Jedina metoda web servisa dohvaća meteorološke podatke za zadani aerodrom iz primljenog argumenta. Metoda dajMeteo(icao) dohvaća aerodrom na temelju argumenta kako bi se dobila njegova GPS lokacija. S tom lokacijom poziva se metoda getRealTimeWeather(lokacija) na objektu OWMKlijent.

Za preuzimanje meteoroloških podataka koristi se Java biblioteka NWTiS_REST_lib v 2.3.0 koja se nalazi u repozitoriju http://nwtis.foi.hr:8088/repository/NWTiS_2022/. Biblioteka se temelji na RESTful servisu koji pruža OpenWeatherMap te predstavlja njegov omotač. JavaDoc biblioteke nalazi se na adresi https://nwtis.foi.hr/NWTiS/apidocs/NWTiS_REST_lib/2.3.0/. Za korištenje OpenWeatherMap i NWTiS_REST_lib potrebno je obaviti registraciju kako bi se dobili korisnički podaci (apikey) i spremili u konfiguracijsku datoteku (postavka „OpenWeatherMap.apikey"). Detaljniji opis nalazi se u pripremi za vježbu.

Web aplikacija pruža WebSocket servis za obavještavanje svojih aktivnih korisnika o trenutnom vremenu na poslužitelju prve aplikacije i broju aerodroma za koje se prikupljaju podaci. Prethodno je kod opisa klase Osvjezivac opisan postupak slanja na bazi dretve. Ako na WebSocket stigne poruka „info" potrebno je poslati standardnu informaciju svim aktivnim korisnicima.

Tablica 1. Postavke za 1. aplikaciju

Ključ	Opis	Vrijednost
ciklus.spavanje	Vrijeme spavanja ciklusa u sekundama	300
OpenWeatherMap.apikey	korisnički ključ (apikey) za OpenWeatherMap – treba upisati vlastite podatke	xxxxxx

Web aplikacija `{LDAP_korisnik}-zadaca_3_wa_2` učitava konfiguracijske podatke putem slušača aplikacije kod pokretanje aplikacije i upisuje ih u atribut konteksta pod nazivom „Postavke". Naziv datoteke konfiguracijskih podataka zapisan je u web.xml kao inicijalni parametar konteksta „konfiguracija" (jedna od datoteka NWTiS.db.config_1.xml ili NWTiS.db.config_2.xml). Web aplikacija nema bazu podataka i podaci koje ona treba prikazivati dohvaća od web aplikacija `{LDAP_korisnik}-zadaca_3_wa_1` putem poziva JAXWS servisa ili poziva LocationIQ i servisa . Web aplikacija koristi Jakarta JSF za realizaciju korisničkog sučelja, koje ima sljedeće funkcionalnosti/poglede:

1. Pregled svih aerodroma (koristi dajSveAerodrome())
2. Pregled svih aerodroma za koje se preuzimaju podaci (koristi dajAerodromePreuzimanje()) i po potrebni unos novog aerodroma (koristi dodajAerodromPreuzimanje(icao))
3. Pregled polazaka s i dolazaka na jedan aerodroma na određeni dan (koristi dajPolaske(icao, dan) i dajDolasci(icao, dan)) uz ispis trenutnih meteoroloških podataka za aerodrom (koristi dajMeteo(icao)). Postoji izbornik u kojem se nalaze: samo polasci, samo dolasci, polasci i dolasci. Putem izbornika filtriraju se podaci koji se prikazuju. Podaci trebaju biti kronološki sređeni u prikazu.
4. Pregled najbližeg aerodroma za zadanu adresu (koristi dajNajbliziAerodrom(lokacija, vrsta)) uz ispis GSP koordinata za adresu i po potrebni (poseban gumb) ispis trenutnih meteoroloških podataka za aerodromu (koristi dajMeteo(icao)). Postoji izbornik u kojem se nalaze: praćeni, svi. Putem izbornika filtriraju se podaci koji se prikazuju. Koristi se Ajax za dohvat GPS lokacije na temelju adrese, najbližeg aerodroma i meteoroloških podataka. Koristi WebSocket za primanje podataka o vremenu na poslužitelju i broju aerodroma za koje se preuzimaju podaci o polascima i dolascima. Ima gumb kojim se šalje poruka „info" na WebSocket krajnju točku kako bi se obavilo slanje informacije svim korisnicima.

Za preuzimanje GPS lokacije koristi se Java biblioteka NWTiS_REST_lib v 2.3.0. Biblioteka se temelji na RESTful servisu koji pruža LocationIQ te predstavlja njegov omotač. Za korištenje LocationIQ i NWTiS_REST_lib potrebno je obaviti registraciju kako bi se dobili korisnički podaci (token) i spremili u konfiguracijsku datoteku (postavka „LocationIQ.token"). Detaljniji opis nalazi se u pripremi za vježbu.

Svima funkcinalnostima pristupa se putem poveznica koje se nalaze u početnoj stranici aplikacije (index.xhtml), a svaka funkcionalnost mora imati poveznicu na početnu stranicu aplikacije. Za realizaciju (1) treba se koristiti straničenje. Straničenje može biti realizirano vlastitim rješenjem (JSF, PrimeFaces ili JSF + JavaScript) pri čemu se smiju učitati svi podaci za pregled pa prikazivati po stranicama. Može se koristiti Datatables[1] za koji isto vrijedi da treba dohvatiti samo podatke stranice koja se prikazuje.

Tablica 2. Postavke za 2. aplikaciju

Ključ	Opis	Vrijednost
LocationIQ.token	Korisnički ključ za LocationIQ – treba upisati vlastite podatke	xxxxxx
stranica.brojRedova	Broj redova po stranici	15

[1] https://datatables.net/download/index