
Vježba: 9 – Zadaća 2. Web sustav "Informacije o aerodromima, letovima i problemima"

Naziv projekta: **{LDAP_korisnik}-zadaca_2**

Korijenski direktorij treba biti **{LDAP_korisnik}-zadaca_2**

Sve nove klase trebaju biti u paketu **org.foi.nwtis.{LDAP_korisnik}.zadaca_2**. Za rad s postavkama treba koristiti Java biblioteke iz {LDAP_korisnik}-zadaca_2_lib_03_1 (nastao na temelju {LDAP_korisnik}-vjezba_03_1) i {LDAP_korisnik}-zadaca_2_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_2.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_2.pdf** (u korijenskom direktoriju projekta).

Struktura .zip datoteke predane zadaće treba biti sljedeća:

```
{LDAP_korisnik}-zadaca_2
	{LDAP_korisnik}-zadaca_2.pdf
	{LDAP_korisnik}-zadaca_2_wa_1
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_2_wa_2
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_2_lib_03_1
		pom.xml
		src
			main
				...
	{LDAP_korisnik}-zadaca_2_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.

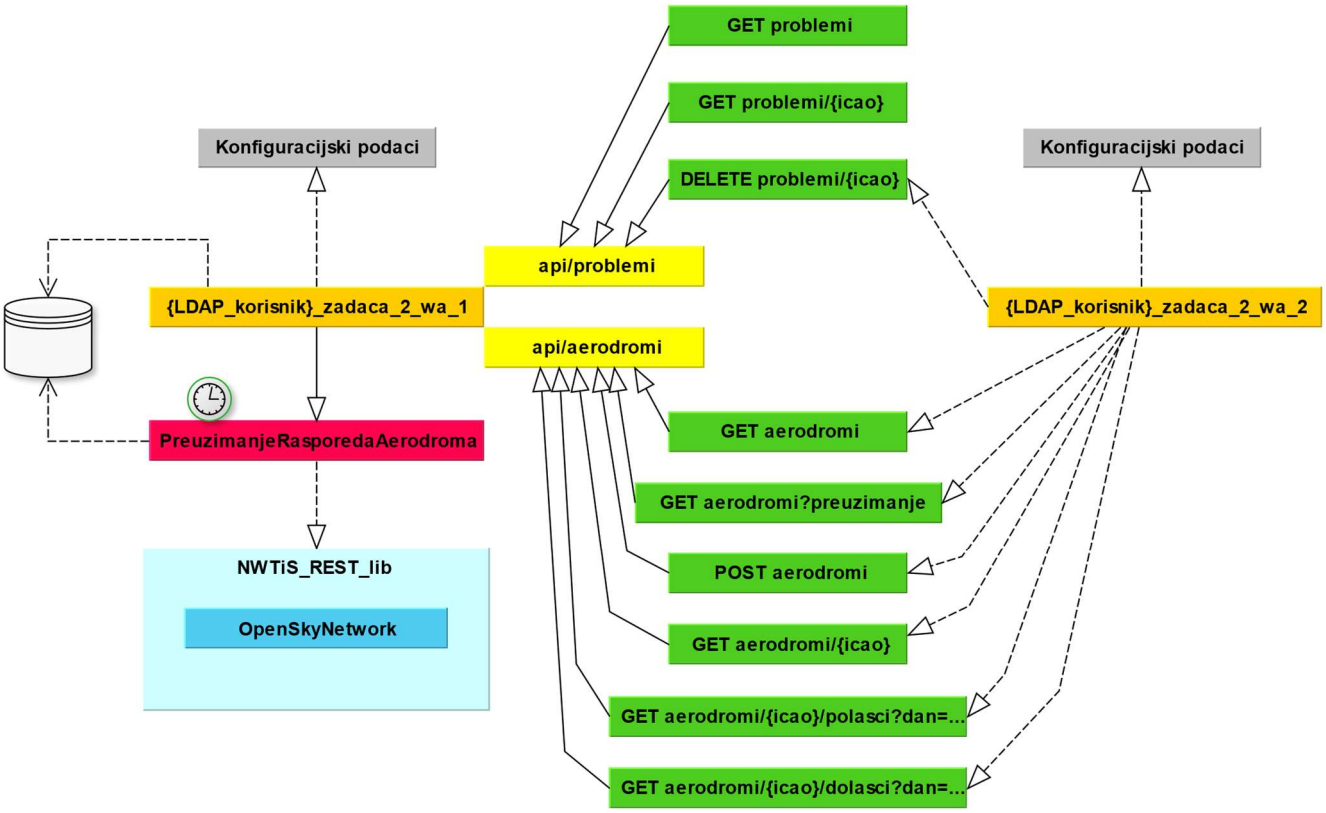
U projektu **ne smiju se koristiti klase i/ili metode koje su zastarjele** tj. anotirane su kao @Deprecated. Na Properties projekta u Java Compiler/Errors/Warnings kod Deprecated and restriced API **obavezno treba postaviti/označiti** *Signal use of depreated API inside deprecated code* i *Signal overrriding 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 Payara Web Server) prikuplja i sprema podatke od vanjskih servisa te pruža RESTful servis. Druga web aplikacija (instalirana na Payara Web Server) usmjerena je na korisničko sučelje čije podatke puni tako da šalje REST pozive servisu iz prve web aplikacije. Shema sustava prikazana je na slici 1.



Slika 1. Shema sustava

Prvo je potrebno pokrenuti poslužiteljsku web aplikaciju `{LDAP_korisnik}-zadaca_2_wa_1`. Nakon toga može se pokrenuti korisnička web aplikacija `{LDAP_korisnik}-zadaca_2_wa_1`

Web aplikacija `{LDAP_korisnik}-zadaca_2_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 PreuzimanjeRasporedaAerodroma. Kod zaustavljanja aplikacije potrebno je upisati u novi redak na kraju datoteke (naziv je određen postavkom „dnevnik.datoteka") „Rad aplikacije: od – do" gdje se od i do zapisuju u formatu dd.mm.gggg hh:mm:ss. Aplikacija pruža RESTful servis pod krajnjom točkom „api" (klasa RestServis).

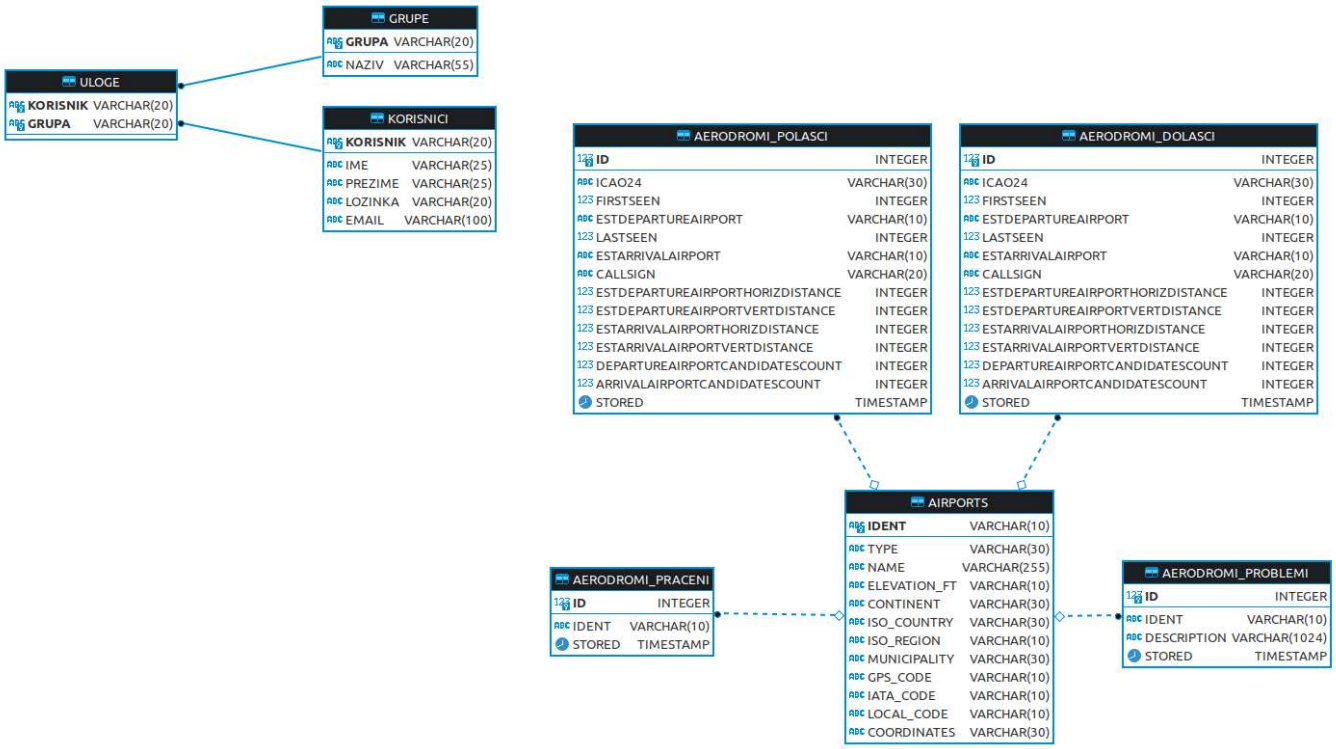
Klasa PreuzimanjeRasporedaAerodroma je dretva koja radi u ciklusima identičnog trajanja. Dretva ima dva brojača ciklusa. Prvi je stvarni brojač ciklusa, a drugi je virtualni brojač ciklusa koji služi kod korekcija vremena spavanja. Oni imaju iste vrijednosti sve dok se ne desi jedna od kasnijih opisanih situacija. Vrijeme početka rada 1. ciklusa označava se kao t_1. Vrijeme t_1 izraženo je u milisekundama i sva kasnija vremena trebaju se pretvarati u milisekunde (sekunde, sati, dani). Zadano je vrijeme ciklusa (postavka „ciklus.vrijeme", u sekundama), koje se označava kao Δtc. Vrijeme početka 2. ciklusa dobije se kao $t_2 = t_1 + \Delta tc$. Općenito može se zapisati da vrijeme početka i+1 tog ciklusa kao $t_{i+1} = t_i + \Delta tc$ ili kao $t_{i+1} = t_1 + (i * \Delta tc)$. To znači da svaki ciklus započinje u vrijeme koje je jednako vremenu početka prethodnog ciklusa

uvećano za zadano vrijeme ciklusa. U vrijeme trajanje ciklusa ulazi vrijeme efektivnog rada koji se obavlja u ciklusu, označava se kao Δte, a preostalo vrijeme do zadanog vremena ciklusa dretva treba spavati, označava se kao Δts. To znači da se vrijeme spavanja izračunava kao Δts = Δtc - Δte. Ako je vrijeme efektivnog rada dretve dulje od zadanog vremena dretve (Δte > Δtc) tada se vrijeme spavanja izračunava kao razlika između prvog umnoška zadanog vremena dretve koji je veći od vremena efektivnog rada dretve i vremena efektivnog rada dretve, Δts = (n * Δtc) - Δte, gdje je n=2,3,... prvi koji ispunjava uvjet (n * Δtc) > Δte. Ovo je prva situacija u kojoj dolazi do razlike u stvarnom i virtualnom brojaču ciklusa. Virtualni brojač ciklusa će sada biti povećan za n, dok će stvarni brojač ciklusa biti povećan za 1. Korekcija vremena spavanja provodi se u određenim situacijama, (postavka „ciklus.korekcija", broj, npr. 10), koja se označava kao ck. U svakom j-tom ciklusu (j - stvarni brojač ciklusa) u kojem je j % ck = 0, potrebno je napraviti korekciju vremena spavanja kako bi se postigla visoka točnost ciklusa (razina 1 milisekunda) u odnosu na početak rada (1. ciklus), broj ciklusa i zadano vrijeme ciklusa. Time će se korigirati odstupanje koje nastaje zbog internog rada sustava prilikom prijelaza dretve u spavanje i buđenja nakon spavanja. Odnosno postignut će se da $t_{k+1} = t_1 + (k * \Delta tc)$ (k - virtualni brojač ciklusa).

Dretva ima zadatak da preuzima podatke o polascima i dolascima letova s određenog skupa aerodoma i upiše ih u tablice u bazi podataka. Trenutno vrijeme na računalu označavaju kao t_r. Zbog točnosti podataka ne želi se preuzimati podaci koji su još u fazi usklađivanja tako da se daje vremenski odmak od stvarnog vremena, koji se označava kao Δto (postavka „preuzimanje.odmak", u danima). U jednom ciklusu dretve preuzimaju se podaci za sve aerodrome iz skupa. Kako ne bi došlo do zagušenja poslužitelja s kojeg se preuzimaju podaci, nakon obrade svakog aerodroma potrebno je napraviti kratku pauzu tj. dretva treba spavati određeno vrijeme (postavka „preuzimanje.pauza", u milisekundama). Podaci se preuzimaju za zadano razdoblje polaska i dolaska aerodroma (postavke „preuzimanje.od" i „preuzimanje.do", zapisane su u formatu dd.mm.gggg hh:mm:ss), koje se označavaju kao tp_1 i tp_n. U jednom ciklusu dretve preuzimaju se podaci za određeno vrijeme (postavka „preuzimanje.vrijeme", u satima), koje se označava kao Δtv. Dretva u 1. ciklusu preuzima podatke za interval vremena koji je definiran početkom tp_1 i krajem $tp_1 + \Delta tv$. Dretva u 2. ciklusu preuzima podatke za interval vremena koji je definiran početkom $tp_2 = tp_1 + \Delta tv$ i krajem $tp_2 + \Delta tv$. Dretva prestaje s radom u i-tom ciklusu kada je $tp_i > tp_n$. Ako je $tp_i > t_r - \Delta to$ tada dretva treba spavati 1 dan. Ovo služi da se ne preuzimaju podaci koji se usklađuju u servisu OpenSky Network. U slučaju spavanja dretve u trajanju od 1 dan dolazi se do druge situacije u kojoj dolazi do razlike u stvarnom i virtualnom brojaču ciklusa. Virtualni brojač ciklusa će biti povećan za m, m = 1 dan/Δtc, dok će stvarni brojač ciklusa biti povećan za 1.

Skup aerodroma koji se prate definiran je u tablici „AERODROMI_PRACENI". Polasci letova upisuju se u tablicu „AERODROMI_POLASCI" ako imaju definiranog aerodroma dolaska (!= null). Dolasci letova upisuju se u tablicu „AERODROMI_DOLASCI" ako imaju definiranog aerodroma polaska (!= null). Ukoliko u pojedinom ciklusu preuzimanje podataka za određeni aerodrom nije uspješno tada se upisuje u tablicu „AERODROMI_PROBLEMI". Dijagram baze podataka prikazan je na slici 2.

Za preuzimanje podataka o polascima i dolascima letova s aerodroma 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 OpenSky Network te predstavlja njegov omotač. JavaDoc biblioteke nalazi se na adresi https://nwtis.foi.hr/NWTiS/apidocs/NWTiS_REST_lib/2.3.0/. Za korištenje OpenSky Network i NWTiS_REST_lib potrebno je obaviti registraciju kako bi se dobili korisnički podaci i spremili u konfiguracijsku datoteku (postavke „OpenSkyNetwork.korisnik" i „OpenSkyNetwork.lozinka"). Detaljniji opis nalazi se u pripremi za vježbu.



Slika 2. Dijagram baze podataka

RESTful servis koji pruža web aplikacija ima aerodrome kao resurs (putanja „aerodromi", klasa RestAerodromi) unutar kojeg se za rad s pojedinim aerodromom (putanja „{icao[1]}") dohvaćaju podaci o polascima letova (putanja „polasci") i dolascima letova (putanja „dolasci"). Mogu se dodati aerodromi za koje se želi preuzimati podatke o polascima i dolascima letova. Drugi resurs se odnosi na spremljene probleme kod preuzimanja podataka (putanja „problemi", klasa RestProblemi) unutar kojeg se za pojedini aerodrom (putanja „{icao}") dohvaćaju informacije o njegovim problemima. Mogu se obrisati svi problemi za pojedini aerodrom. Klase RestAerodromi i RestProblemi svoje usluge/servise temelji na MIME „application/json".

Tablica 1. Postavke za 1. aplikaciju

Ključ	Opis	Vrijednost
dnevnik.datoteka	Naziv datoteke za dnevnik	NWTiS_dnevnik.txt
ciklus.vrijeme	Vrijeme ciklusa u sekundama	300
ciklus.korekcija	Broj ciklusa nakon koje se radi korekcija	10
preuzimanje.odmak	Odmak od trenutnog vremena, u danima	3
preuzimanje.pauza	Pauza između dva aerodroma, u milisekundama	20
preuzimanje.od	Datum od kojeg se počine preuzimanje, u formatu dd.mm.gggg	01.01.2022
preuzimanje.do	Datum do kojeg se provodi preuzimanje, u formatu dd.mm.gggg	31.12.2022
preuzimanje.vrijeme	Vrijieme za koje se preuzimaju podaci u jednom ciklusu, u satima	6
OpenSkyNetwork.korisnik	korisničko ime za OpenSky Network – treba upisati vlastite podatke	xxxxxx
OpenSkyNetwork.lozinka	lozinka za OpenSky Network – treba upisati vlastite podatke	yyyyyy

[1] icao – 4 slovna jednoznačna međunarodna oznaka aerodroma. Sva slova su velika npr. LDZA, LOWW, EDDF.
Više informacija na: https://airportcodes.io/en/icao-codes/ , https://www.world-airport-codes.com/

Web aplikacija `{LDAP_korisnik}-zadaca_2_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_2_wa_1` putem poziva RESTful servisa. Adresa krajnje točke RESTful servisa iz web aplikacija `{LDAP_korisnik}-zadaca_2_wa_1` nalazi se konfiguracijskim podacima (postavka „adresa.wa_1"). Web aplikacija koristi Jakarta MVC za realizaciju korisničkog sučelja, koje ima sljedeće funkcionalnosti/poglede:

1. Pregled svih aerodroma (koristi GET aerodromi)
2. Pregled svih aerodroma za koje se preuzimaju podaci (koristi GET aerodromi?preuzimanje)
3. Pregled jednog aerodroma (koristi GET aerodromi/{icao})
4. Pregled polazaka s jednog aerodroma na određeni dan (koristi GET aerodromi/{icao}/polasci?dan=dd.mm.gggg)
5. Pregled odlazaka s jednog aerodroma na određeni dan (koristi GET aerodromi/{icao}/odlasci?dan=dd.mm.gggg)
6. Dodaj jednog aerodroma za preuzimanje (koristi POST aerodromi)
7. Pregled svih problema (koristi GET problemi)
8. Pregled problema od jednog aerodroma (koristi GET problemi /{icao})
9. Brisanje problema od jednog aerodroma (koristi DELETE problemi /{icao})

Svima funkcinalnostima pristupa se putem poveznica koje se nalaze u početnoj stranici aplikacije (indeks.jsp), a svaka funkcionalnost mora imati poveznicu na početnu stranicu aplikacije. Funkcionalnost 2 je osnova za poziv funkcionalnosti 4 i 5 u obliku poveznica za pojedini aerodrom. Za realizaciju (1, 2, 4, 5, 7, 8) treba se koristiti straničenje kod najmanje 2 pregleda. Straničenje može biti realizirano vlastitim rješenjem (JSP ili JSP + JavaScript) pri čemu se ne smiju učitati svi podaci za pregled pa selektivno prikazivati po stranicama. Treba dohvatiti samo podatke za stranicu koja se prikazuje, a količina koja se dohvaća određena je brojem redova u stranici (postavka „stranica.brojRedova"). Može se koristiti Datatables[2] za koji isto vrijedi da treba dohvatiti samo podatke stranice koja se prikazuje.

Tablica 2. Postavke za 2. aplikaciju

Ključ	Opis	Vrijednost
adresa.wa_1	Adresa krajnje točke RESTful servisa iz 1. aplikacije	http://localhost:8080/api
stranica.brojRedova	Broj redova po stranici	15

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