Projektni zadatak

XML i veb servisi, 2018/19. školska godina

Projektovati, implementirati i testirati zdravstveni informacioni sistem (ZIS). ZIS treba da podrži i interne poslovne procese u zdravstvenoj ustanovi (vođenje zdravstvenih kartona, izdavanje lekarskih izveštaja, uputa i recepta, itd.) i eksterne poslovne procese (izbor lekara, zakazivanje pregleda, pregled zdravstvenih kartona, itd.).

U ZIS postoje korisničke uloge **pacijent**, **medicinski tehničar** i **lekar**.

Pacijent može da:

1. izabere lekara
   - implementirati formu u kojoj pacijent može da se opredeli za izabranog lekara sa spiska lekara
   - voditi računa o opterećenju lekara
   - podatke o izabranom lekaru čuvati u zdravstvenom kartonu pacijenta
2. zakaže pregled kod izabranog lekara
   - implementirati formu u kojoj pacijent može da zakaže pregled kod izabranog lekara u jednom od raspoloživih termina
   - termine modelovati u okviru kalendara svakog lekara
3. pregleda svoj zdravstveni karton (koji sadrži izveštaje, upute, recepte, itd.)
   - implementirati formu za pretragu zdravstvenog kartona pacijenta u kojoj je pacijentu omogućen unos slobodnog teksta (tj. osnovna pretraga) i metapodataka (tj. napredna pretraga)
   - pronalaženje medicinske dokumentacije po više metapodataka realizovati upotrebom logičkih operatora
   - prilikom pregledanja dokumenata omogućiti *link* na referenciranu medicinsku dokumentaciju, a u formi za pretragu omogućiti pronalaženje svih dokumenata koji referenciraju dati dokument
   - prilikom modelovanja medicinske dokumentacije kao XML dokumenata, posebno obratiti pažnju na polja koja predstavljaju metapodatke i reference na druge dokumente, a na osnovu kojih će biti implementirana napredna pretraga

Medicinski tehničar može da:

1. ažurira zakazane preglede
   - implementirati formu za uvidu u zakazane preglede za odabranog lekara, i potvrdu ili promenu termina zakazanog pregleda
   - prilikom promene zakazanog termina omogućiti notifikaciju pacijenata

2. izdaje medicinsku dokumentaciju u agregiranom i anonimizovanom obliku
    - za potrebe sekundarne upotrebe podataka i medicinske dokumentacije omogućiti izvoz zdravstvenih kartona pacijenata u agregiranom i anonimizovanom obliku, primenom XSLT-a nad izvornim dokumentima.

Lekar može da:

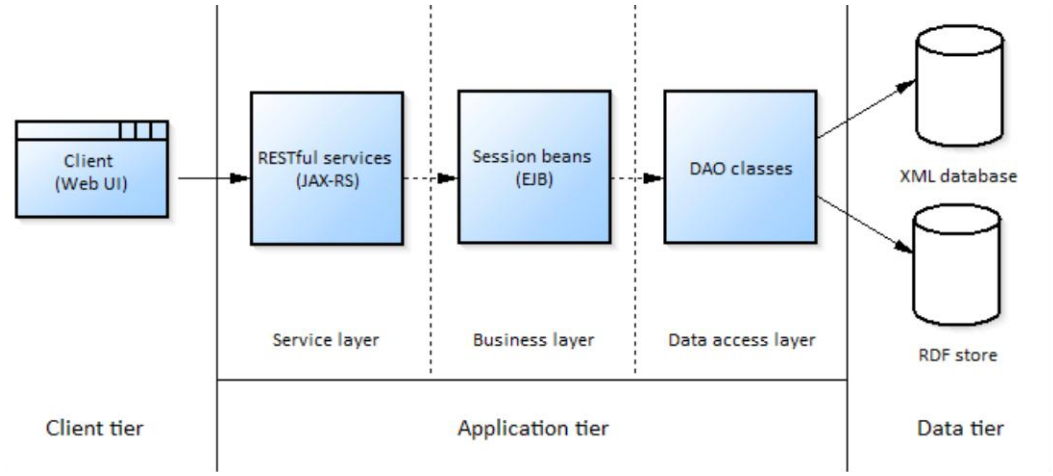
1. ažurira zdravstvene kartone pacijenata
    - implementirati formu za unos i izmenu postojećih podataka pacijenta
    - implementirati pretragu po pacijentima tako da je lekaru omogućen unos slobodnog teksta (tj. osnovna pretraga) i metapodataka (tj. napredna pretraga)
    - pronalaženje medicinske dokumentacije po više metapodataka, kao i u slučaju pretrage sopstvene dokumentacije kod pacijenta, realizovati upotrebom logičkih operatora
    - prilikom modelovanja pacijenata kao XML dokumenata posebno obratiti pažnju na polja koja predstavljaju metapodatke na osnovu kojih će biti implementirana napredna pretraga
    - prilikom pregledanja dokumenata prikazati *link* na referenciranu medicinsku dokumentaciju, a u formi za pretragu omogućiti pronalaženje svih dokumenata koji referenciraju dati dokument
2. izdaje izveštaje
    - omogućiti lekaru da u posebnoj formi u okviru *rich edit* komponente unese svoje mišljenje i postavi dijagnozu u vidu izveštaja (slabo strukturirani XML dokument), na osnovu obavljenog pregleda i pregledane medicinske dokumentacije u zdravstvenom kartonu pacijenta
    - prilikom izdavanja izveštaja omogućiti lekaru referenciranje na dokumente iz zdravstvenog kartona pacijenta
3. izdaje recepte
    - nakon izdatog izveštaja u posebnoj formi omogućiti lekaru izdavanje recepta kao anotiranog teksta u okviru posebne *rich edit* komponente
    - implementirati algoritam preporuke leka na osnovu dijagnoze iz izveštaja lekara i faktora rizika pacijenta (npr. alergija na penicilin) iz zdravstvenog kartona pacijenta
4. izdaje upute
    - nakon izdatog izveštaja u posebnoj formi omogućiti lekaru izdavanje uputa (npr. radi stručnog mišljenja lekara specijaliste) kao anotiranog teksta u okviru posebne *rich edit* komponente
    - uput omogućava pacijentu zakazivanje pregleda kod lekara referenciranog uputom

Potrebno je omogućiti i pojedinačan izvoz dokumenata u XHTML i PDF formatima i metapodataka dokumenata u RDF i JSON formatima i izvoz dokumenata radi sekundarne upotrebe podataka u agregiranom i anonimizovanom obliku za potrebe donošenja odluka, unapređenja prakse i naučna istraživanja.

Poslovne procese modelovati kao UML dijagrame aktivnosti. Dokumente, metapodatke dokumenata i identifikatore dokumenata modelovati po W3C standardima. Dokumente (izveštaje, upute, recepte i

zdravstvene kartone) modelovati kao XML tipove dokumenata u XML Schema jeziku. Metapodatke dokumenata modelovati kao skup RDF iskaze. Identifikatore dokumenata modelovati kao URL.

Sistem projektovati po troslojnoj softverskoj arhitekturi, a komunikaciju između serverske i klijentske strane realizovati pomoću RESTful veb servisa. Izbor programskog jezika i platforme za implementaciju serverske i klijentske strane prepušten je studentima (primeri na vežbama su u Javi). Primer arhitekture sistema u Java/EJB tehnologiji dat je grafičkim prikazom na sledećem dijagramu (slika 1).



Slika 1: Višeslojna arhitektura zdravstvenog informacionog sistema

Backend aplikacija ZIS predstavlja višeslojnu aplikaciju koja se sastoji iz slojeva prikazanih dijagramom. Aplikacioni sloj ZIS realizovati kroz tri podsloja (*data access*, *business* i *service layer*) sa sledeći način:

Data access layer

- generički sloj aplikacije enkapsulira CRUD (*create*, *retrieve*, *update*, *delete*) operacije za pristup podacima u XML I RDF bazama podataka
- DAO klase implementirati ili kao namenske komponente (npr. SLSB) ili kao obične Java klase (POJO)

Business layer

- srednji sloj aplikacije koji implementira fasadu nad slojem za pristup podacima. *Business layer* posreduje između *data access* i *service layer*-a, stavljajući servisnom sloju na raspolaganje neophodan skup funkcionalnosti kroz metode poslovne logike
- *business layer* implementirati kao namenske komponente

Service layer

- krajnji sloj aplikacije koji definiše javno dostupan API implementiranih funkcionalnosti krajnjem korisniku

- servisni sloj implementirati kao RESTful veb servise koje referenciraju komponente iz *business layer*-a upotrebom *dependency injection* mehanizma