Pristup podacima iz programskog koda

Projektni zadatak

Ishodi učenja					UKUPNO
I1	I2	I3	I4	I5	
20	20	20	20	20	100

UPUTE

- Obrana projekta odvija se za vrijeme ispitnih rokova
- Student radi prijavu kao i za pismene ispite
- Rješenja projektnih zadataka predaju se putem GitHub platforme te je obavezno korištenje Git alata prilikom izrade istih

ISHODI UČENJA:

Ishod 1 (20 bodova):

- *minimalni ishod učenja (10 bodova):* Izraditi softversko rješenje uporabom relacijske baze podataka u oblaku kao izvora podataka
- *željeni ishodu učenja (10 bodova)*: Izraditi relacijsku bazu podataka u oblaku i softversko rješenje uporabom relacijske baze podataka u oblaku kao izvora podataka

Ishod 2 (20 bodova):

- *minimalni ishod učenja (10 bodova):* Izraditi softversko rješenje uporabom rješenja za pohranu nestrukturiranih podataka u oblaku kao izvora podataka
- *željeni ishodu učenja (10 bodova)*: Izraditi bazu za pohranu nestrukturiranih podataka u oblaku i softversko rješenje uporabom rješenja za pohranu nestrukturiranih podataka u oblaku kao izvora podataka

Ishod 3 (20 bodova):

- *minimalni ishod učenja (10 bodova):* Izraditi softversko rješenje uporabom nerelacijske baze podataka u oblaku kao izvora podataka
- *željeni ishodu učenja (10 bodova)*: Izraditi nerelacijsku bazu podataka u oblaku i softversko rješenje uporabom nerelacijske baze podataka u oblaku kao izvora podataka

Ishod 4 (20 bodova):

- *minimalni ishod učenja (10 bodova):* Odabrati i implementirati optimalni konceptualni model podataka
- *željeni ishodu učenja (10 bodova)*: Odabrati i implementirati optimalni složeni konceptualni model podataka

Ishod 5 (20 bodova):

- *minimalni ishod učenja (10 bodova):* Implementirati softversko rješenje uporabom odabranih alata ORM
- *željeni ishodu učenja (10 bodova)*: Implementirati složeno softversko rješenje uporabom odabranih alata ORM

Prvi projekt (Ishod 1 – 20 bodova, Ishod 2 – 10 bodova, Ishod 4 – 20 bodova, Ishod 5 – 20 bodova)

Dizajnirajte i implementirajte programsko rješenje temeljeno na relacijskom podatkovnom modelu koji opisuje scenarij medicinskog sustava odgovornog za upravljanje pacijentima, njihovom medicinskom dokumentacijom (povijest bolesti), pregledima i receptima. Sustav bi trebao omogućiti korisnicima (liječnicima) dodavanje i uređivanje pacijenata, upravljanje pacijentovim pregledima i lijekovima te prijenosom i pregledom dodatnih medicinskih datoteka za određene preglede. Medicinska dokumentacija mora sadržavati povijest bolesti pacijenta. Na posljetku, rješenje bi trebalo omogućiti izvoz podataka o pacijentima u CSV formatu.

Pacijenti su definirani imenom, prezimenom, OIB-om, datumom rođenja i spolom. Medicinska dokumentacija prikazuje evidenciju prethodnih bolesti za pacijente koje su opisane nazivom bolesti te vremenom početka i završetka bolesti (ukoliko je pacijent prestao bolovati od iste). Pregledi se provode na određenom pacijentu u određenog datuma i vremena te po tipu, mogu biti jedan od sljedećih postupaka (uz šifre):

- GP – Opći tjelesni pregled
- KRV – Test krvi
- X-RAY – rendgensko skeniranje
- CT – CT sken
- MR – MRI sken
- ULTRA - Ultrazvuk
- EKG - Elektrokardiogram
- ECHO - Ehokardiogram
- EYE – pregled očiju
- DERM – Dermatološki pregled
- DENTA – pregled zuba
- MAMMO - Mamografija
- NEURO – Neurološki pregled

Po izboru, korisnici (liječnici) mogu odabrati prijenos slika (obično za postupke koji uključuju medicinsko snimanje) za određeni pregled pacijenta. Korisnicima je potrebno omogućiti pretraživanje pacijenta po prezimenu i OIB-u te prikaz pacijentovih informacija: povijest bolesti, listu recepata i listu pregleda i njihove pojedinosti.

Programsko rješenje potrebno je razviti kao web aplikaciju, a moguće ga je implementirati u jeziku i radnom okviru (eng. *framework*) po vašem izboru. Međutim, obavezno je koristiti Postgres instancu u oblaku kao bazu podataka te oblikovni obrazac Repository Factory i *Lazy* paradigmu. Na obrani projekta potrebno je detaljno objasniti rad razvijene web (MVC, MVC+MVVM, Web API + SPA, ...) aplikacije. Ako nemate željeni jezik i okvir, preporučuje se korištenje *C#* i ASP.NET (Web API) okvira za poslužiteljsku aplikaciju te *Vue.js* kao radni okvir za klijentsku aplikaciju.

Ishod učenja 1 (Minimalni – 5 bodova) Kreirajte Postgres instancu baze podataka u oblaku sa potrebnim entitetima i relacijama oblaku putem Tembo ili Supabase servisa. Prilikom obrane projekta, obavezno je objasniti korake prilikom kreiranja baze podataka u oblaku te opisati detalje povezivanja na istu. Također, potrebno je demonstrirati izvršavanje različitih DDL i DML SQL skripti korištenjem klijentskog alata po izboru (*DBeaver*, *DataGrip*, *pgAdmin, psql-repl* , vlastito rješenje...).

Ishod učenja 1 (Minimalni – 5 bodova, Željeni – 10 bodova) Kreirajte web aplikaciju koja omogućuje CRUD operacije nad opisanim povezanim entitetima (izbor arhitekture i tehnologije ostavljena je na slobodan izbor). Dodatno, potrebno je implementirati sljedeće:

- validacija podataka
- detaljno objašnjenje komunikacije između aplikacije i baze podataka

Ishod učenja 2 (Željeni – 10 bodova) Implementirajte spremanje i dohvat dokumenata (nalaza) korištenjem nestrukturirane pohrane.

Ishod učenja 4 (Minimalni – 10 bodova) Kreirajte konceptualni podatkovni model koji se sastoji od više entiteta povezanih stranim ključevima (kardinaliteta 1:1, 1:N ili N:M). Napravite odgovarajuću DDL skriptu koju je moguće izvršiti. Konceptualni model potrebno je predati kao odgovarajući ER-dijagram.

Ishod učenja 4 (Željeni – 10 bodova) Nadogradite konceptualni model podataka uvođenjem više entiteta povezanih različitim odnosima poštujući pravila 3. normalne forme. Nadogradnju je potrebno demonstrirati na obrani projekta.

Ishod učenja 5 (Minimalni – 10 bodova) Nadogradite bazu podataka i aplikaciju tako da je moguće prenijeti neograničen broj slika. Baza podataka mora biti izgrađena pomoću odabranog ORM (npr. *Entity Framework*), prema željenom pristupu (*model first, code first, datbase first*). Prilikom obrane, obavezno je objašnjenje rada korištenog ORM alata.

Ishod učenja 5 (Željeni – 10 bodova) Istražite i implementirajte migracije, korištenjem odabranog ORM alatai demonstrirajte njihov rad na primjeru dodavanja novog stupca postojećoj tablici u bazi podataka. Shodno tome, nadogradite web aplikaciju na način koji omogućuje dodavanje novog atributa entitetu (npr. Broj pacijenta). Tijekom obrane projekta potrebno je pokazati teoretsko razumijevanje i praktičnu primjenu migracija.

Drugi projekt (Ishod 2 – 10 bodova, Ishod 3 – 20 bodova)

Dizajnirajte i implementirajte programsko rješenje koje učinkovito prikuplja i pohranjuje podatke o onkološkim pacijentima i njihovoj **genskoj ekspresiji** u nestrukturiranoj pohrani u oblaku (**MiniO bucket storage**). Podaci su dio **TCGA** (The Cancer Genome Atlas) projekta te su prikupljeni u obliku TSV datoteka, kojima se može pristupiti putem portala Xena Browser. Rješenje bi trebalo učitavanje datoteka s poveznice *(web scrapping)* koje sadrže ekspresije gena te vizualizaciju podataka ekspresije gena za pacijente pomoću nerelacijske baze podataka u oblaku (**MongoDB**) i nestrukturirane pohrane **(MiniO).**

Ishod učenja 2 (Minimalni – 10 bodova) Kreirajte programsko rješenje (skripta, *pipeline*) koje dohvaća i učitava **TSV** datoteke koji sadrže podatke o ekspresije gena onkoloških pacijenata, putem portala **Xena** preglednika i pohranjuje ih u MiniO nestrukturiranu pohranu spremnika u oblaku. Datoteke se mogu pronaći ako posjetite poveznice za svaku pojedinost o određenom raku te odaberete **IlluminaHiSeq pancan normalized** datoteku u odjeljku *gene expression* Ako kohorta raka ne sadrži takvu datoteku, slobodno ju možete zanemariti.

Ishod učenja 3 (Minimalni – 10 bodova) Navedeno rješenje mora se spojiti na nestrukturiranu pohranu podataka i čitati vrijednosti ekspresije gena iz kolekcije TSV datoteka. Podatke o pacijentu treba transformirati tako da je lako pristupiti vrijednostima ekspresije gena koji čine **cGAS-STING** genski put: **C6orf150** (cGAS), **CCL5**, **CXCL10**, **TMEM173** (STING), **CXCL9**, **CXCL11**, **NFKB1**, **IKBKE**, **IRF3**, **TREX1** , **ATM**, **IL6** i **IL8** (CXCL8). Ti bi podaci trebali uključivati i vrijednosti **patient_id** i **cancer_cohort** (atribut čiju vrijednost vi morate popuniti, uzimajući u obzir kohortu kojoj pripadaju ekspresije koje čitate). Također, potrebno je omogućiti vizualizaciju podataka o ekspresiji gena odabranih pacijenata dohvaćajući ih iz **MongoDB nerelacijske baze** podataka u oblaku.

Ishod učenja 3 (Željeni – 10 bodova) Implementirajte spajanje dohvaćenih podataka o ekspresiji gena iz kolekcije TSV datoteka s povezanim podacima o kliničkim podacima pacijenata koji se nalaze u datoteci **TCGA_clinical_survival_data.tsv**. Ciljani stupci iz kliničkih podataka trebaju uključivati:

- **bcr_patient_barcode** – TCGA identifikator pacijanta, stupac koji se koristi prilikom spajanja podataka o ekspresiji i kliničkim vrijednostima pojedinog pacijenta
- **DSS** (Disease Specific Survival) – vrijednost koja označava pacijentovo preživljavanje tumorske bolesti (1 – preživio, 0 – nije preživio)
- **OS** (Overall Survival) – vrijednost koja označava pacijentovo preživljavanje (1 - preživio, 0 – nije preživio)
- **clinical_stage** – opisuje stadij tumora zabilježenog za danog pacijenta