Pitanja i odgovori sa LAMS-a iz predmeta CS230

Sadržaj

LEKCIJA 1 - Uvod u distribuirane sisteme i Java EE

1. Šta je distribuirani sistem?
   Distribuirani sistem je kolekcija procesora koji komuniciraju slanjem poruka preko mreže za komunikaciju.
2. Navesti i objasniti 6 osnovnih karakteristika DS?
   * Deljenje resursa - ovo je jedan od osnovnih zahteva kod distribuiranih sistema. Potrebno je razmenjivati podatke i deliti uređaje između sistema,
   * Otvorenost - ova karakteristika podrazumeva mogućnost proširivanja DS-a na različite načine,
   * Istovremenost - s obzirom da se u DS-u nalazi više računara, neophodno je da se istovremeno izvršava više različitih procesa,
   * Skalabilnost - mogućnost dodavanja novih resursa,
   * Otpornost na greške - sposobnost da ukoliko se desi neka greška sistem se sam oporavi, ili da postoji rezervno rešenje koje će u tom slučaju nastaviti sa radom,
   * Transparentnost - stepen skrivanja pojedinih komponenti DS-a od krajnjih korisnika i programera aplikacija.
3. Koji su osnovni ciljevi oblikovanja DS?
   Performanse, pouzdanost, skalabilnost, konzistenstnost, sigurnost.
4. Šta je naziv (ime), a šta identifikator?
   Naziv (ime) predstavlja naziv koji ima značenje za korisnika ili sistem, dok identifikator predstavlja naziv koji ima smisla samo za sistem.
5. Objasnite dva osnovna načina komuniciranja u DS.
   Klijent-server - komunikacija između dva procesa, i grupno slanje - komukacija između grupe procesa koji međusobno sarađuju.
6. Šta je multicast?
   Multicast je vrsta komuniciranja, odnosno slanja, gde se svaka poruka šalje svim članovima jedne grupe, a ne samo jednom članu.
7. Uporedite karakteristike centralizovanih i distribuiranih sistema.
   Kod centralizovanih sistema, softverski elementi ne mogu da funkcionišu samostalno, dok kod distribuiranih mogu. Kod centralizovanih sistema su svi resursi dostupni, dok kod distribuiranih resursi mogu biti ograničeni. Centralizovani sistemi se izvršavaju kroz jedan proces, dok se distribuirani izvršavaju u više paralelnih procesa.
8. Navedite prednosti i nedostatke distribuiranih sistema.
   Prednosti: ekonomičnost, brzina, podela posla, pouzdanost, proširivost.
   Nedostaci: složeniji softver, obavezna upotreba mreže, mogućnost da više komponenti bude izvan funkcije, sigurnost.
9. Navedite neke primere primene distribuiranih sistema.
   Internet, intranet, distribuirani operativni sistemi.
10. Šta je Java EE?
    Java EE (Enterprise Edition) platforma je moćna, razvijena i široko korišćena platforma za razvoj distribuiranih aplikacija.
11. Objasnite slojeve Java EE aplikacija.
    Klijentski sloj se može implementirati na računaru, mobilom telefonu ili bilo kom drugom uređaju koji ima ugrađen JRE. Web sloj je veoma važan za kreiranje veb aplikacija. Sloj poslovne obrade je sloj gde se vrši kreiranje poslovne logike aplikacije.
12. Šta je Java zrno?
    Java zrna pojednostavljuju razvoj većih, distribuiranih aplikacija, sadrže poslovnu logiku aplikacije, koja je jasno odvojena od prezentacionog sloja. Java zrna su prenosiva, tj. mouguće ih je koristiti u razvoju novih aplikacija.

13. Objasnite tipove Java zrna.
    Zrna sesije (session beans), izvršavaju zadatke koje im klijent zada. Zrna vođena porukom (message-driven beans) služe za osluškivanje i reagovanje na određeni tip poruka.

LEKCIJA 2 - Servleti:

1. Šta je servlet?
   Servlet predstavlja server-side komponentu koja se izvršava isključivo u okviru Java virtualne mašine. Servlet se izvršava na serverskoj strani.
2. Koje su prednosti primene servleta?
   Mogu pristupiti čitavoj familiji Java API-a, uključujući i JDBC za pristup bazama podataka. Takođe, mogu pristupiti bilioteci HTTP specifičnih poziva, poprimiti sve povoljnosti Java jezika, kao što su prenosivost, performanse, ponovno korišćenje i zaštitu.
3. Za šta se obično koriste HTTP servleti?
   HTTP servleti se obično koriste za:
   * obezbeđenje dinamičkog sadržaja kao što je uzimanje rezultata upita i vraćanje istih do klijenta,
   * obradu i čuvanje podataka koji se nalaze na HTML strani,
   * upravljanje informacijama koje se odnose na stanje HTTP-a.
4. Objasnite tokove podataka između klijenta i servera.
   Klijent (veb pretraživač) šalje Http zahtev veb serveru, koji zatim šalje Http zahtev servletu za nekom informacijom npr. Servlet poziva određeni Java kod i izvšrava ga, nakon čega dobijene podatke vraća veb serveru putem Http servlet odgovora, a nakon toga veb server dostavlja te podatke veb pregledaču u vidu Http odgovora.
5. Navedite pravila koja bi morala da zadovolji servlet aplikacija.
   * Klijent računar koji se obraća servletu mora da ima instaliran veb pregledač.
   * Na računaru koji hostuje stranicu kreiranu pomoću servleta mora da bude pokrenut neki veb server, koji će da prima zahteve korisnika.
   * Servlet izvšava neki Java kod, koji ima određeni zadatak.
   * Veb server izoluje sadržaj iz HttpServletResponse objekta, pakuje gau HttpResponse objekat i šalje ga veb pregledaču.
   * Veb pregledač prikazuje korisniku primljenu stranicu.
6. Kako se kreira servlet?
   Servlet se kreira kao i bilo koja druga Java klasa, u okviru unapred kreirane veb aplikacije. Desnim klikom na folder "Source packages" bira se opcija "New -> Servlet...".
7. Objasnite životni ciklus servleta.
   Životni ciklus servleta počinje učitavanjem servleta. Nakon učitavanja, kreira se instanca servleta. Kada se kreira instanca, servrletski kontejner poziva init() metodu i prosleđuje joj inicijalne parametre. Nakon uspešno završenog procesa inicijalizacije, servlet je spreman za rad. Ukoliko se servlet ne koristi duže vreme, tj. nije potreban za obradu zahteva, kontejner poziva metodu destroy().
8. Objasnite HTTP GET i POST zahteve u radu sa servletima.
   U html strani se navodi tip metode koja se koristi. Ukoliko se ništa ne navede, poziva se get metoda. Get metoda služi za prikaz podataka sa servera, dok se post metoda koristi za slanje podataka sa stranice na server.
9. Šta je sesija?
   Sesija predstavlja logički zadatak koji korisnik pokušava da izvrši pristupanjem veb prezentaciji.

10. Objasnite najpoznatije načine praćenja sesija.
    Čuvanje sesija se može obaviti na strani klijenta ili na strani servera. Na strani klijenta korisnički podaci se čuvaju u vidu "kolačića" ili modifikovanjem URL adrese.
11. Kako objekat HttpSession čuva podatke o sesiji?
    Pozivom metode getSession(true) pronalazi se objekat sesije klijenta ili se kreira novi objekat ukoliko ne postoji. Rezultat poziva ove metode se čuva u objektu klase HttpSession.
12. Šta je GlassFish server i koja mu je namena?
    GlassFish server predstavlja aplikacioni server koji je baziran na Java EE implementaciji i u potpunosti podržava Java EE platformu, primenu Java EE veb profila i napisan je i dizajniran kao podrška razvoju Java veb aplikacija.
13. Navesti neke prednosti GlassFish servera.
    Pouzdanost, skalabilnost, kvalitetno upravljanje greškama, visok stepen performansi koje pružaju veb aplikacije.

LEKCIJA 3 - Java Server Pages (JSP)

1. Objasnite kako veb server kreira veb stranicu koristeći JSP.
   Veb pregledač šalje HTTP zahtev serveru. Server prepoznaje da je HTTP zahtev za veb stranicu i prosleđuje ga JSP kontejneru (ekstenzija fajla je .jsp). JSP mehanizam učitava JSP stranicu i konvertuje je u servlet sadržaj, prevodi servlet u izvršnu klasu i prosleđuje zahtev servlet kontejneru. Veb server učitava servlet klasu i izvršava je. Dobija se izlaz u HTML formi, koji se prosleđuje veb serveru u okviru HTTP odgovora. Veb server prosleđuje odgovor veb pregledaču, koji obrađuje HTML stranicu i prikazuje je.
2. Koji zadatak obavlja JSP kontejner?
   JSP kontejner proverava da li JSP datoteka već postoji i ukoliko postoji, proverava da li je vreme izmene JSP stranice starije od servleta. Ukoliko jeste, JSP kontejner smatra da JSP stranica nije izmenjena i da se podudara sa servletom.
3. Navesti i objasniti faze životnog ciklusa JSP stranice.
   Kompajliranje - kada veb pregledač zatraži JSP stranicu, JSP kontejner proverava da li je neophodno kompajlirati istu, i ukoliko je potrebno, on izvšrava prevođenje te stranice.
   JSP inicijalizacija - kada kontejner učita JSP stranicu, on poziva jspInit() metodu, kojom se vrši inicijalizacija te JSP stranice.
   JSP izvršavanje - u ovoj fazi se dešavaju sve interakcije za zahtevima. U ovoj fazi kontejner poziva metodu jspService(), koja preuzima dva parametra, HttpServletRequest i HttpServletResponse.
   JSP čišćenje - u ovoj fazi se vrši uklanjanje JSP stranice iz upotrebe od strane kontejnera.
4. Šta je skriptlet?
   Skriptlet je osnovni element JSP sintakse. Pomoću skriptleta se java kod može ubaciti u neku jsp stranicu. Označava se karakterima <% za početak i %> za kraj. Njime može biti obuhvaćen bilo koji broj Java iskaza, promenljivih ili deklaracija metoda.
5. Šta su JSP direktive?
   JSP direktive su instrukcije koje utiču na celokupnu strukturu servlet klase. Opšti kod direktive je <%@ direktiva atribut="vrednost" %>.
6. Koji zadatak imaju JSP akcije?
   Zadatak JSP akcija je kontrola ponašanja servlet kontejnera.

7. Šta su imlicitni objekti? Navedite ih i objasnite.
   Implicitni objekti su Java objekti koji su dostupni na svakoj stranici, uz pomoć JSP kontejnera. Mogu direktno da se koriste bez potrebe za eksplicitnom deklaracijom. Postoje sledeći implicitni objekti:
   * request - objekat klase HttpServletRequest;
   * response - objekat klase HttpServletResponse;
   * out - objekat klase PrintWriter koji služi za slanje izlaza ka klijentu;
   * session - objekat klase HttpSession;
   * application - objekat klase ServletContext;
   * config - objekat klase ServletConfig;
   * pageContext - služi za enkapsulaciju primene specifičnih elemenata servera;
   * page - isto što i this, koristi se za pozivanje metoda servlet klase;
   * exception - dozvoljava pristup podacima izuzetaka.
8. Objasnite razliku između GET i POST metoda.
   Get metoda se koristi za slanje parametara kroz url kako bi se dobile neke informacije od servera, dok se Post metoda koristi za prosleđivanje podataka serveru sa neke forme.
9. Kada zahtevom šaljete informacije poput korisničkog imena i lozinke za koju biste se metodu opredelili? Zašto?
   Kada se šalju poverljive informacije, poput korisničkog imena i lozinke, uvek se treba koristiti metoda Post, zato što se na taj način ne prikazuju podaci kroz url. Ukoliko bi se podaci slali pomoću Get metode, podaci bi se videli u url-u.

LEKCIJA 4 - Java Server Pages - napredni koncepti

1. Zašto se koriste servleti u kombinaciji sa JSP filterima?
   Servleti se koriste u kombinaciji sa JSP filterima da bi filteri presreli zahtev klijenta pre nego što dođe do servera ili zahtev servera pre nego što dođe do klijenta, u cilju vršenja određene provere ili akcije.
2. Koje tipove JSP filtera poznajete?
   Filteri autentifikacije, kompresovanja podataka, enkripcije, okidači za pristupanje događajima, konverzije slike, logovanja i provere, lanca mime-type, tokenizacije, XSL/T - za tranformisanje XML sadržaja.
3. Navesti i objasniti korake koji se izvode kada se koriste kolačići za identfikovanje korisnika.
   * Server skript šalje kolačiće ka veb pregledaču,
   * Veb pregledač snima kolačiće na lokalnom računaru kako bi mogao da ih koristi u budućnosti,
   * Sledeći put kada pregledač pošalje bilo kakav zahtev ka serveru, on zajedno sa zahtevom šalje i podatke kolačića serveru.
4. Objasniti kako se podešavaju i koriste kolačići u JSP stranicama?
   Prvo je potrebno kreirati objekat kolačića (Cookie cookie = new Cookie("ključ", "vrednost");). Nakon toga je potrebno podesiti parametre kolačića primenom odgovarajućih metoda nad objektom cookie. Jedan od parametara je dužina trajanja kolačića, a podešava se na sledeći način: cookie.setMaxAge(vremeUSekundama);.
   Nakon kreiranja objekta kolačića i podešavanja parametara, taj kolačić se šaljeu zaglavlje HTTP odgovora. Ovo se radi na sledeći način: response.addCookie(cookie);.
5. Ako klase za rad na veb aplikaciji nisu direktno dostupne, kako mogu da se učine dostupnim?
   Mogu se preuzeti potrebne biblioteke sa interneta i dodati u projekat. Nakon toga se klase iz tih biblioteka mogu koristiti u projektu.

6. Objasniti i demonstrirati različite načine primene Java klase Date u JSP stranicama.
   Java klasu Date je moguće koristiti u JSP stranicama na isti način kao i u bilo kojoj Java klasi. Ona vraća trenutni datum i vreme kreiranjem novog objekta (Date datum = new Date();).
7. Kako se formatiraju datum i vreme u JSP stranicama?
   Formatiranje datuma i vremena se vrši pomoću klase SimpleDateFormat, u čiji se konstruktor pirlikom kreiranja objekta definiše format datuma koji želimo da dobijemo. Nakon toga se nad tim objektom poziva metoda format, kojoj se prosleđuje objekat klase Date.
8. Objasniti kako je moguće na dva načina realizovati preusmeravanje JSP stranica?
   Pozivom metode sendRedirect() kojoj se prosleđuje lokacija na koju se preusmerava.
   Drugi način je korišćenje metoda setStatus() i setHeader().
9. Kako su klasifikovani JSTL tagovi?
   Na osnovne tagove, tagove za formatiranje, SQL tagove, XML tagove i JSTL funkcije.
10. Objasnite pojedinačno kategorije JSTL tagova.
    Osnovni tagovi se mogu koristiti uključivanjem Core JSTL biblioteke i podržavaju osnovne operacije za rad sa izrazima na stranici.
    Tagovi za formatiranje služe za formatiranje brojeva, datuma, vremenske zone, lokalizacije, itd.
    SQL tagovi služe za interagovanje sa relacionim bazama podataka u JSP stranicama.
    XML tagovi služe za kreiranje i manipulisanje XML dokumentima iz JSP stranica.
    JSTL funkcije se uglavnom koriste za manipulisanje stringovima na JSP stranicama.
11. Kako se koristi JSTL SQL kolekcija tagova, u JSP stranicama, za realizovanje CRUD operacija?
    Pomoću sql:setDataSource taga se podešava baza podataka sa kojom se radi. Da bi se izvršio neki upit (upis, čitanje, update, delete) koristi se tag sql:query.
12. Šta je JavaBEan i koje ima specifičnosti?
    JavaBean je klasa u Javi koja je specijalno kreirana u skladu sa JavaBeans API specifikacijama. Specifičnosti po kojima se JavaBeans klase razlikuju od ostalih Java klasa su sledeće:
    * Obezbeđuju podrazumevane argumente, bez konstruktora,
    * Implementiraju interfejs Seriazable,
    * Sadrže brojne atribute koje je moguće učitavati i upisivati,
    * Za atribute sadrže brojne getere i setere.
13. Objasnite primenu akcije useBean u JSP stranicama?
    Akcija useBean se koristi kako bi se iz JSP stranice pristupilo JavaBeans klasi. Pomoću taga <jsp:useBean...> se kreira objekat JavaBeans klase, a zatim se pomoću tagova jsp:getProperty i jsp:setProperty pristupa, odnosno setuje vrednost atributima kreiranog objekta.
14. U kojim slučajevima, u JSP stranicama, je moguće koristiti JSP EL izraz?
    Moguće je koristiti JSP EL izraze kako bi se pristupilo aplikativnim podacima koji se čuvaju u komponentama koje odgovaraju Java zrnima. Izrazi mogu biti aritmetički i logički.
15. Koja JAva klasa je zadužena za korišćenje lokalnog jezika, pisma i podešavanja u JSP stranicama?
    Klasa Locale iz paketa java.util.

LEKCIJA 5 - JavaServer Faces

1. Šta čini jednu JSF aplikaciju?
   JSF aplikaciju čine jedna ili više XHTML stranica koje sadrže JSF tagove, jedno ili više CDI zrna i konfiguracione fajlove "faces-config.xml" (opciono).
2. Navedite i objasnite faze razvoja JSF aplikacije.
   Faze razvoja JSF aplikacije mogu biti: Development, Production, SystemTest i UnitTest. U Development fazi su dostupne dopunske informacije koje pomažu prilikom debagovanja programa. Faza Production se bavi performansama, dok se preostale dve faze koriste kod testiranja.
3. Šta predstavlja facelet?
   Facelet je XHTML fajl sa dodatim specifičnim tagovima.
4. Objasnite JSF menadžer raspoeda.
   Menadžer rasporeda podrazumeva korišćenje taga <h:panelGrid> koji označava da će se svi elementi prikazati u formi matrice. Navodi se broj kolona, kao i stilovi (poravnanje i sl.) koji će biti primenjeni na kolone. Nakon toga se svaki element dodaje u odvojenu ćeliju.
5. Kako omogućava primenu CSS stilova?
   Pomoću taga <h:outputStylesheet name="nazivCssFajla.css"/>. CSS fajl mora biti smešten u okviru foldera resources -> CSS kako bi ga program prepoznao. U suprotnom, potrebno je navesti putanju fajla u atributu library u prethodno objašnjenom tagu.
6. Objasnite vezu između for i id atributa.
   Id atribut jednoznačno označava neki element na formi (npr. polje za unos teksta). S obzirom da se labela koja stoji pored polja za unos teksta unosi kao poseban element (pomoću taga h:outputLabel), potrebno je povezati za koje tekstualno polje je ta labela, a to se radi pomoću atributa "for", čija vrednost mora biti ista kao i vrednost "id" atributa tekstualnog polja.
7. Šta je CDI zrno?
   CDI zrno je klasična Java klasa koja je obeležena određenim anotacijama i služi za prihvatanje podataka koje korisnik unosi na JSF stranici.
8. Kako se obeleava klasa CDI zrna?
   Anotacijom @Named.
9. Koje oblasti može da pokrije CDI zrno? Kojim anotacijama su određene oblasti?
   Request - @RequestScoped,
   Session - @SesionScoped,
   Conversation - @ConversationScoped,
   Application - @ApplicationScoped,
   Dependent - @Dependent,
   Flow - @FlowScoped.
10. Šta je klasa validator?
    Klasa validator je klasa kojom definišem neki šablon za unos podataka na formi koji kasnije možemo da primenimo nad poljima u formi. Neopohodno je da ova klasa implementira interfejs Validator.
11. Koji zadatak obavlja klasa validator?
    Služi za kontrolu unosa podataka od strane korisnika na formi. Ukoliko je, npr. potrebno da neko polje sadrži samo brojeve, ili da sadrži neki obavezan karakter, to se proverava u klasi validator i ukoliko se unos ne poklapa sa šablonom, vraća se poruka korisniku da unos nije ispravan.

12. Objasnite primenu atributa reqiured.
    Atribut required označava da je unos tog polja obavezan. Ukoliko se polje označeno ovim atributom ne popuni, prikazaće se greška korisniku sa odgovarajućom porukom Atribut required može imati vrednosti true ili false. Po default-u je false.
13. Navedite prednosti primene JSF šablona.
    Primenom šablona postiže se veća održivost veb aplikacije, zato što se sve izmene u rasporedu komponenata stranice dešavaju na jednom mestu.

14. Kako se kreira datoteka šablona?
    File -> New File -> JavaServerFaces -> Facelets Templates -> Next -> "odabir šablona i definisanje naziva" -> Finish
15. Šta je klijent šablona?
    Klijent šablona mora postojati kako bi šablon mogao biti upotrebljen. On nasleđuje sve što postoji u šablonu, a dalje se može modifikovati u skladu sa potrebama.
16. Šta predstavlja Resource Library Contract?
    Biblioteka ugovora resursa (Resource library contracts) predstavlja nov JSF 2.2 alat. Služi za kreiranje Facelet šablona za kreiranje tematskih veb aplikacija. Moguće je definisati različite teme za aplikaciju pomoću ove biblioteke.

LEKCIJA 6 - JSF biblioteke komponenata

1. Objasnite kako se uključuju PrmeFaces komponente u projekat veb aplikacije?
   Potrebno je prilikom kreiranja veb aplikacije odabrati framework "JavaServer Faces" i u tabu components označiti "PrimerFaces".
2. Objasnite primenu i organizaciju komponente PrimeFaces.
   PrimeFaces komponenta je klasična html stranica, sa uključenim p namespece-om, koji omogućava korišćenje specifičnih tagova PrimeFaces-a. Stranica se organizuje iz delova (north, west, east, south, center), koji se navode u atributu position.
3. Objasnite kako se uključuju ICEFaces komponente u projekat veb aplikacije?
   Potrebno je preuzeti ICEFaces biblioteku sa interneta. Zatim je potrebno kreirati novu veb aplikaciju i prilikom kreiranja odabrati framework "JavaServer Faces", a kao komponentu "ICEFaces". U delu More... potrebno je selektovati dva preuzeta fajla "icefaces-version.jar" i "icefaces-ace-version.jar", nakon čega će biblioteka biti uključena u projekat.
4. Navedite i objasnite dva skupa ICEFaces komponenata?
   ICE komponente - poseduju funkcionalnost koja je primarno implementirana kao server-side kod sa ograničenom primenom javascript-a. Koristi se kada je potrebno obezbediti kompatibilnost sa starijim verzijama veb pregledača.
   ACE komponente - novija implementacija u formi kombinacija server-side i client-side koda. Koristi se kod savremenih veb pregledača.
5. Koji skup ICEFaces komponenata smo koristili i zašto?
   ACE, zato što je noviji i kompatibilan sa modernim veb pregledačima.
6. Objasnite kako se uključuju RichFaces komponente u projekat veb aplikacije?
   Prilikom kreiranja veb aplikacije potrebno je odabrati framework "JavaServer Faces" i komponentu "RichFaces", koju je neophodno prvo podesiti. To se radi preko dugmeta "More..." koje se nalazi pored naziva komponente. Da bi se podesila komponenta, potrebno je prvo preuzeti biblioteku sa interneta, a zatim pristupiti podešavanjima.
7. Objasnite i demonstrirajte upotrebu komponente.
   Komponenta je slična prethodno obrađenim komponentama PrimeFaces i IceFaces. Na xhtml stranici gde želimo da koristimo ovu komponentu potrebno je uključiti namespace ove

komponente (xmlns:rich), a nakon toga možemo na stranici koristiti određene tagove koji su specifični za ovu komponentu.

LEKCIJA 7 - Java Persistence API

1. Šta su JPA entiteti?
   JPA entiteti su Java klase čija se polja čuvaju u bazama podataka pomoću JPA API.
2. Kojom anotacijom se obeležavaju JPA entiteti?
   @Entity
3. Šta je conection pool i koje su mu prednosti?
   Connection pool sadrži informacije za konektovanje na bazu podataka, poput naziva servera, porta, pristupnih podataka i sl. Prednost korišćenja connection pool-a je što konekcije u njemu nikad nisu zatvorene i mogu se jednostavno koristiti kada za njima postoji potreba.
4. Šta su DAO objekti?
   DAO (Data Access Objects) šablon dizajna obezbeđuje sve funkcionalnosti pristupa bazi podataka.
5. Zašto primena DAO šablona predstavlja dobru praksu za pisanje koda koji se obraća bazi podataka?
   Odvaja se modul programa pomoću kojeg ostali slojevi aplikacije, kao npr. logika korisničkog interfejsa ili poslovna logika, ne zavise od logike za pristup bazi podataka.
6. Objasnite proceduru kreiranja JPA entiteta iz tabela baze podataka.
   Potrebno je prilikom kreiranja klase odabrati opciju "Entity Classes from Database". U prozoru koji će se otvoriti potrebno je odabrati tabele za koje želimo kreirati entitete. Klikom na dugme "Finish" kreira se entitet.
7. Objasnite anotacije koje koriste JPA entiteti.
   @Entity - ukazuje na to da se radi od entitetu.
   @Table - označava se naziv tabele sa kojom je entitet povezan.
   @Id - označava polje koje je u bazi podataka označeno kao primarni ključ.
   @Column - označava naziv kolone sa kojom je polje sa ovom anotacijom povezano.
8. Šta je obeleženi upit?
   Anotacija @NamedQuery prihvata dva argumenta: name i query. Koristi se za kreiranje upita koji će kasnije moći da se koristi za izvlačenje nekih podataka iz baze.
9. Šta je JPQL?
   JPQL je specifični jezik upita čija je sintaksa slična SQL jeziku. Generiše se iz čarobnjaka New Entity Classes from Database za svako polje iz entiteta.
10. Navedite i objasnite anotacije za proveru podataka.
    @NotNull - označava da polje ne sme biti prazno (null).
    @Size - definiše veličinu polja (min i/ili max vrednost).
    @Pattern - definiše šablon po kojem se mogu unositi vrednosti polja.
11. Navedite i objasnite kardinalnosti relacija JPA entiteta.
    @OneToMany, @ManyToOne, @ManyToMany, @OneToOne.
12. Pokazati na primeru kardinalnosti JPA entiteta.
    @OneToMany i @ManyToOne - jedan student može imati više upisa, jedan upis može biti vezan samo za jednog studenta.
    @ManyToMany - jedan proizvod se može naći u više porudžbina, jedna porudžbina može imati više proizvoda.
    @OneToOne - jedan student može imati registrovan samo jedan nalog na sistemu, jedan nalog na sistemu može biti vezan samo za jednog studenta.

13. Objasnite proceduru automatskog generisanja JSF stranica iz JPA entiteta.
    Prilikom kreiranja novog fajla, potrebno je odabrati opciju Preferences - JSF Pages from Entity Classes. Nakon toga treba odabrati entitete za koje želimo da se generišu JSF stranice i klikom na dugme "Finish" se navedene stranice kreiraju. Za svaki entitet se kreiraju stranice za dodavanje, izmenu i brisanje entieta, kao i za prikaz liste svih entiteta i prikaz detalja o odabranom entitetu.

LEKCIJA 8 - Implementacija poslovnog nivoa pomoću zrna sesija

1. Koji zadatak vrše zrna sesije?
   Zrna sesije enkapsuliraju poslovnu logiku Java EE aplikacije.
2. Navedite i objasnite tipove zrna sesije.
   * Zrno sesije bez stanja (stateless) - ne održava stanje konverzacije sa klijentima između poziva metoda.
   * Zrno sesije sa stanjem (stateful) - održava stanje konverzacije sa klijentima između poziva metoda.
   * Singleton zrno sesije - u aplikaciji može da postoji samo jedna instanca ovog zrna.
3. Koji zadatak obavlja klijent EJB zrna?
   EJB klijent koristi projekat Java biblioteke klasa koji sadrži udaljeni interfejs. Kod klijenta mora da sadrži referencu na instancu klase koja implementira udaljeni interfejs zrna.
4. Objasnite kako je kreiran kod za povezivanje klijenta sa EJB zrnom.
   Desnim klikom na kod u klasi Main bira se opcija "Call Enterprise Bean...". Otvara se prozor u kojem je neophodno izabrati EJB zrno kojem će se klijent obraćati. Klikom na dugme "OK", u kod Main klase se dodaje varijabla Remote tipa koja je obeležena anotacijom @EJB.
5. Šta su transakcije i koji im je zadatak?
   Transakcije omogućavaju izvođenje svih koraka u nekoj metodi i u slučaju pada nekog koraka, ove vraćaju stanje na početak, pre promena koje su nastale u toku izvršavanja metode.
6. Kojom anotacijom je olakšano upravljanje transakcijama u Java EE aplikacijama?
   @TransactionAttribute
7. Objasnite razliku kada je anotacojim obleležena klasa u odnosu na obeležavanje pojedinačnih metoda.
   Ukoliko je samo metoda obeležena, onda se ta anotacija odnosi samo na tu metodu. Ukoliko je cela klasa obeležena, onda se anotacija odnosi na sve metode u klasi. Ukoliko su klasa i neka metoda u njoj obeležene ovom anotacijom, ali sa različitim vrednostima u njoj, prednost će imati anotacija kojom je obeležena metoda.
8. Kako je implementiran AOP primenom presretača?
   Kreiranjem posebne klase presretača i obeležavanjem EJB klase anotacijom @Interceptors.
9. Šta je presretač?
   Presretač je Java klasa koja sadrži jednu metodu koja je obeležena anotacijom @AroundInvoke i kao parametar prima objekat klase InvocationContext. Ova metoda presreće metodu nad kojom se poziva i izvršava svoj deo koda pre izvršenja glavne metode.
10. Kako funkcioniše EJB Timer servis?
    EJB Timer servis funkcioniše tako što se na određeno vreme izvršava određena metoda.
11. Koju ulogu obavlja anotacija @Schedule?
    Anotacijom @Schedule se obeležava metoda koja se treba izvršavati na određeno vreme. Period izvršavanja se navodi kao parametar metode (minuti, sati, sekunde).

12. Objasnite procedure generisanja zrna sesije iz JPA entiteta.
    Nakon kreiranog EJB Module projekta, kreira se novi fajl pomoću opcije "Session Beans Fro Entity Classes". Tu je potrebno odabrati željene entitete za koje želimo da kreiramo zrna sesije. Klikom na dugme "Finish" kreiraju se željena zrna sesije.
13. Koju ulogu vrši apstraktna klasa AbstractFacade?
    Klasa AbstractFacade obezbeđuje metode za izvođenje CRUD operacija nad entitetima.

LEKCIJA 9 - Umetanje konteksta i zavisnosti

1. Koja je razlika između standardnih JSF i CDI aplikacija?
   Razlika je u primeni CDI obeleženih zrna umesto JSF upravljanih zrna za modele i kontrolere.
2. Objasnite ulogu anotacije @Named?
   Anotacijom @Named se označava da je u pitanju klasa CDI zrna. Moguće je u zagradama navesti atribut "value" i postaviti ga na neko drugo ime koje će biti korišćeno za zrno u .xhtml stranicama.
3. Objasnite razliku između JSF i CDI oblasti zrna.
   Slični su. CDI uvodi dve nove oblasti: Coversation i Dependent.
4. Objasnite primenu anotacije @Inject.
   Anotacija @Inject se koristi da se neka klasa umetne tokom vremena izvršavanja (runtime). Na ovaj način se vrši umetanje zavisnoti u CDI aplikacijama.
5. Šta je kvalifikator?
   Kvalifikator je specifična anotacija predstavljena anotacijom @Qualifier.
6. Kako se kreira kvalifikator?
   Kvalifikator se kreira izborom sldećih opcija: File - New File - Contexts and Dependency Injection - Qualifier Type. Ovde je potrebno uneti naziv kvalifikatora i paket i klikom na dugme "Finish" kvalifikator će biti kreiran.
7. Kojom anotacijom se obeležava kvalifikator?
   @Qualifier.
8. Opišite kako se primenjuje kvalifikator na potklasu ili implementaciju interfejsa.
   Tako što se ta klasa obeleži anotacijom koja ima naziv kvalifikatora (@Premium npr.).
9. Šta je stereotip u Java EE aplikacijama?
   Stereotip je nova anotacija kojom se menja grupa od nekoliko CDI anotacija.
10. Kako se kreiraju stereotipovi?
    Kreira se tako što se iz File menija bira opcija New File, zatim kategorija Contexts and Dependency Injection i tip Stereotype. Navodi se naziv fajla, paket i klikom na dugme "Finish" kreira se stereotip.
11. Koje datoteke su neophodne da bi tipovi povezivanja presretača imali punu funkcionalnost?
    Interceptor Binding Type, klasa presretača, CDI konfiguraciona datoteka, kontroler.
12. Šta su tipovi povezivanja presretača?
    Tipovi povezivanja presretača omogućavaju povezivanje presretača sa zrnom.
13. Objasnite kako se kreiraju tipovi pozivanja presretača.
    Iz file menija bira se opcija NewFile, a zatim kategorija Contexts and Dependency Injection i tip fajla Interceptor Binding Type. Unošenjem imena i paketa i klikom na dugme "Finish" kreira se tip povezivanja presretača.
14. Kako se kreiraju vlastite CDI oblasti?
    Prilikom kreiranja novog fajla bira se kategorija Contexts and Dependency Injection i tip Scope Type. Definisanjem imena i paketa i klikom na dugme "Finish" završava se kreiranje vlastite CDI oblasti.

15. Koja vrsta programera će koristiti pristup kreiranju vlastith CDI oblasti?
    Programeri koji se bave krerianjem framework-ova.

LEKCIJA 10 - JMS i zrna vođena porukama

1. Šta predstavlja JMS?
   JMS (Java Message Service) je standardni Java EE API za upravljanje porukama koji omogućava asinhronu komunikaciju između Java EE komponenata.
2. Koje destinacije JMS poruka poznajete?
   Redovi sa porukama (JMS queues) i teme (topics).
3. Koje domene JMS poruka poznajete?
   Point-to-point (PTP) messaging i Publish/Subscribe (pub/sub) messaging.
4. Koje JMS destinacije su karakteristične za različite domene JMS poruka?
   Ukoliko se koristi PTP domen poruka kao JMS destinacija javljaju se redovi za poruke (message queues). Ukoliko se koristi pub/sub domen poruka kao JMS destinacija javlja se tema poruka (message topic).
5. Objasnite kako se dodaje red za poruke na aplikativni server za kreirani Java EE projekat.
   Iz file menija bira se opcija New File... Zatim se bira kategorija GlassFish i tip fajla JMS Resource. Unosi se JNDI naziv i tip resursa Queue. Klikom na dugme Next otvara se prozor gde je potrebno odabrati naziv reda i konačno, klikom na dugme Finish kreira se novi konfiguracioni fajl.
6. Kako se koristi GlassFish konzola za proveru kreiranih JMS resursa?
   U NetBeans IDE-u potrebno je odabrati tab Services i kliknuti desnim klikom na GlassFish server. Iz menija je potrebno odabrati opciju "View Domain Admin Console" nakon čega će se u veb pregledaču otvoriti admin konzola. Iz menija sa leve strane potrebno je odabrati opciju "JMS Resources" i proširiti je. U delu "Destination Resources" moguće je videti kreirane JMS resurse.
7. Kako se naziva XML datoteka u kojoj je upisan kreirani red za poruke?
   glassfish-resources.xml
8. Šta je red za poruke?
   JMS red za poruke je resurs u koji se smeštaju poruke poslate od strane proizvođača JMS poruka i odakle se preuzimaju od strane JMS potrošača.
9. Objasnite kako JMS proizvođač postavlja poruke na red za poruke?
   Pomoću metode za slanje poruke koja se implementira u kontroleru.
10. Objasnite proces automaskog generisanja JMS koda. Objasnite detalje koda koji je generisan i koji je kasnije dodat automatski generisanom kodu.
    Kreira se Java klasa kontrolera. Pomoću kombinacije tastera Alt+Insert otvara se kontekstni meni odakle je potrebno izabrati opciju "Send JMS Message...". Klikom na ovu opciju otvara se novi prozor gde je potrebno odabrati radio dugme "Server Destinations" i tu selektovati prethodno kreirani red. Klikom na dugme "OK" generisaće se kod za slanje JMS poruka u prethodno kreiranoj klasi kontrolera.
11. Šta je Message Driven Bean?
    Message Driven Bean predstavlja zrno vođeno porukama, odnosno JMS potrošač čija funkcija je preuzimanje i obrada poruka iz reda za poruke.
12. Koje osobine može da poseduje zrno vođeno porukama?
    acknowledgeMode, clientId, connectionFactoryLookup, destinationType, destinationLookup, messageSelector, subscriptionDurability, substriptionName.
13. Koji interfejs implementira zrno vođeno porukama?
    MessageListener.

14. Koji zadatak ima metoda onMessage() zrna vođenog porukama?
    Metoda onMessage uzima kao parametar instancu klase javax.jms.Message, a void je tipa. U okviru ove metode moguće je implementirati funkcionalnost po želji, odnosno ono šta je potrebno uraditi sa primljenom porukom.

LEKCIJA 11 - Java API za JSON procesiranje

1. U kojoj formi JSON - P objektni model dozvoljava programerima pisanje JSON objekata?
   U formi strukturiranog stabla.
2. Koji zadatak obavlja add() metoda? Dati detaljno objašnjenje.
   Metoda add() vraća instancu implementacije JsonObjectBuilder interfejsa. Vezivanjem više poziva add() metode dodaje se više elemenata u JSON objekat.
3. Objasniti ulogu metode generateJson().
   Metoda generateJson() služi za generisanje Json objekta i njegovo vraćanje u vidu stringa.
4. Objasniti ulogu metode parseJson().
   Metoda parseJson() ima zadatak da iz Json objekta izvdoji elemente i setuje ih kao atribute određenog objekta.
5. Koje su prednosti primene JSON - P API tokova?
   Bolje performanse kod rukovanja ogromnim količinama podataka.
6. Koje su prednosti primene JSON - P API objektnog modela?
   Jednostavnija primena.
7. Objasniti proces generisanja JSON Stringa iz Java objekta primenom JSON - P API tokova.
   Generisanje JSON stringa se vrši pomoću klase JsonGenerator, za koju se vrši pozivanje jedne ili više redefinisanih metoda write() za dodavanje JSON osobina i odgovarajućih vrednosti u JSON podatke.
8. Objasniti proces popunjavanja Java objekta iz JSON Stringa primenom JSON - P API tokova.
   Popunjavanje Java objekta iz JSON stringa se vrši pomoću metode parseJson(). While petljom se prolazi kroz JSON objekat i pomoću odgovarajućih metoda se proverava tip podataka konkretnog elementa i u zavisnosti od toga se popunjavaju atributi željenog Java objekta.

LEKCIJA 12 - Java API za WebSocket

1. Kako se preuzima demo aplikacija iz NetBeans IDE okruženja?
   Kreira se novi projekat, odabirom opcija File -> New Project -> Samples -> Java EE -> Naziv Projekta.
2. Koja je namena JS koda u priloženom primeru?
   Kreiranje novog objekta WebSocket-a i pozivanje funkcija onopen, onmessage i onerror. Pomoću ovih funkcija ažurira se tekst na stranici.
3. Šta je WebSocket server Endpoint?
   WebSocket server Endpoint je klasa koja obrađuje WebSocket zahtev na serverskoj strani.
4. Kako se kreira klasa krajnje tačke WebSocket servera u NetBeans IDE razvonom okruženju?
   Odabirom opcije File -> New File -> Web -> WebSocket Endpoint. Ovde se unosi naziv klase, paket, kao i putanja WebSocket-a. Klikom na dugme "Finish", završava se kreiranje krajnje tačke WebSocket servera.
5. Koji je zadatak anotacije @OnMessage?
   Prijem poruke od strane klijenta.

6. Kojom anotacijom i na koji način se obeležava krajnja tačka WebSocket servera?
   @ServerEndpoint. Kao parametar anotacije se navodi putanja do te krajnje tačke.
7. Objasnite primenu anotacija: @OnOpen. @OnMessage, @OnClose i @OnError.
   @OnOpen anotacijom se obeležava metoda koja prihvata objekat sesije i otvara sesiju.
   @OnMessage anotacijom se obeležava metoda koja čita akcije i atribute uređaja poslate sa klijenta.
   @OnClose anotacijom se obeležava metoda koja zatvara sesiju.
   @OnError anotacijom se obeležava metoda koja hvata grešku u slučaju da se desi i radi nešto sa tom greškom (čuva u log i sl.).

LEKCIJA 13 - Distribuirani servisi – RESTFul i SOAP veb servisi sa JAX - RS

1. Šta je REST?
   REST predstavlja arhitekturni stil u kojem su veb servisi reprezentovani kao resursi i mogu biti identifikovani preko jedinstvenih identifikatora resursa (URI).
2. Koja je namena RESTFul servisa?
   RESTFul servisi se ponašaju kao frontend za bazu podataka. Klijent koristi RESTFul veb servise za izvođenje CRUD operacija nad bazom podataka.
3. Kako se kreira datoteka RESTFul servisa nad tabelom baze podataka u razvojnom okruženju NetBeans IDE?
   Nakon kreiranja veb aplikacije i uspostavljanja konekcije sa bazom podataka, potrebno je iz File menija odabrati opciju New, a zatim iz kategorije "Web Services" odabrati opciju "RESTful Web Services from Database". Nakon ovog koraka potrebno je odabrati konekciju sa bazom i tabele za koje želimo da kreiramo servise. Praćenjem čarobnjaka i unosom potrebnih podataka, biće kreirani servisi za rad sa tabelama iz baze podataka.
4. Šta predstavlja fasadna klasa?
   Fasadna klasa je omotač klase JPA entiteta. Svaka fasadna klasa nasleđuje apstraktnu klasu AbstractFacade.
5. Šta se testira opcijom Test Resource URI?
   Opcijom Test Resource Uri se testira da li je određeni RESTful veb servis kreiran na adekvatan način.
6. Objasnite postupak testiranja RESTFul servisa opcijom Test RESTFul Services?
   Pomoću opcije Test RESTful Web Services testiraju se svi servisi u aplikaciji. Pokreće se tako što se desnim klikom na projekat otvori kontekstni meni, a iz njega ova opcija.
7. Koji zadatak ima klasa ApplicationConfig.java?
   Zadatak klase ApplicationConfig je da konfiguriše JAX-RS.
8. Kako se generišu datoteke Java klijenata RESTFul veb servisa?
   Potrebno je odabrati opciju File -> New File, a zatim iz kategorije Web Services odabrati opciju RESTful Java Client. Popunjavanjem zahtevanih polja u čarobnjaku i klikom na dugme Finish biće kreirane datoteke Java klijent koda RESTful servisa.
9. Kako se generišu datoteke JS klijenata RESTFul veb servisa?
   Iz fajl menija bira se opcija New File, a zatim iz kategorije Web Services opcija RESTful JavaScript Client. Daljim praćenjem čarobnjaka i unošenjem potrebnih podataka, i na kraju klikom na dugme Finish biće kreriane datoteke JavaScript klijenata RESTful veb servisa.
10. Kako se angažuje generisani JS kod?
    JavaScript kod se poziva sa takođe iz automatski generisanih HTML stranica.
11. Šta karakteriše primenu SOAP protokola?
    Operacije veb servisa se definišu u XML dokumentu pod nazivom WSDL (Web Services Definition Language).

12. Šta je WSDL?
    WSDL je fajl u kojem se definišu operacije SOAP veb servisa.
13. Objasnite upotrebu anotacija @WebService, @WebMethod i @WebParam.
    Sa @WebService se označava klasa koja predstavlja veb servis.
    Anotacijom @WebMethod označava se metoda servisa.
    Anotacijom @WebParam označava se parametar metode koji će biti prosleđivan kroz servis.
14. Objasnite primenu dizajnera veb servisa za kreiranje operacija veb servisa u NetBeans IDE razvojnom okruženju.
    Pomoću dizajnera veb servisa mogu se kreirati metode veb servisa, umesto tradicionalnog kucanja koda. Prilikom kreiranja treba definisati naziv metode, povratni tip i parametre koje će servis primati.
15. Objasnite primenu NetBeans IDE alata Test Web Service.
    Alat Test Web Service služi za testiranje kreiranog veb servisa. Desnim klikom na klasu servisa i odabirom opcije Test Web Service otvoriće se stranica u veb pretraživaču, gde je moguće testirati metode odabranog servisa.
16. Objasnite kako se kreirana Java klasa može registrovati kao klijent postojećeg veb servisa?
    Prevlačenjem određenih metoda iz servisa u željenu klasu.
17. Navedite prednosti primene EJB zrna kao veb servisa.
    Upravljanje transakcijama i aspektno-orijentisano programiranje.
18. Na koje načine je moguće zrna sesije bez stanja definisati kao veb servise?
    Prilikom krerianja veb servisa u projektu EJB modula, taj servis će automatski biti implemetiran kao zrno sesije bez stanja. Pored toga, i postojeća zrna sesije u projektu EJB modula mogu biti deklarisana kao veb servisi.
19. Kako se kreira veb servis iz WSDL?
    U NetBeans IDE razvojnom okruženju se kreira novi fajl na sledeći način: File - New File - Web Services - Web Service from WSDL. Praćenjem čarobnjaka i unošenjem potrebnih podataka biće kreiran veb servis.

LEKCIJA 14 - Oracle Cloud

1. Navedite i ukratko opišite nivoe Oralce Cloud platforme.
   * Nivo pristupa - objedinjuje alate za pristup Oracle Cloud servisima.
   * Nivo servisa - sastavljen je od sledećih tipova servisa: infrastruktura u formi servisa, platforma kao servis, softver kao servis i podaci kao servis.
   * Nivo upravljanja - čine ga mehanizmi za upravljanje operacijama na oblaku, poslovnim procesima na oblaku, bezbednošću itd.
   * Nivo resursa - čine ga apstraktni i fizički resursi: skladišta, diskovi, mehanizmi za obradu itd.
2. Šta čini servisni nivo Oracle Cloud platforme?
   Infrastruktura u formi servisa, platforma kao servis, softver kao servis i podaci kao servis.
3. Od koji servisa je sačinjen Oracle Cloud Iaas?
   Obrada, skladište, umrežavanje, rukovođenje, baze podataka, balansirano učitavanje, krajnji servisi, Ravello, brza konekcija.
4. Od koji servisa je sačinjen Oracle Cloud Paas?
   Upravljanje podacima, razvoj aplikacija, integracija, poslovna analitika, bezbednost, upravljanje, sadržaj i iskustvo.
5. Od koji servisa je sačinjen Oracle Cloud Saas?
   * Customer Experience (CX)
   * Human Capital Management (HCM)
   * Enterprise Resource Planning (ERP)

* Supply Chain Management (SCM)
* Enterprise Performance Management (EPM)
* Internet of Things Applications (IoT)
* SaaS Analytics
* Data
* Industry Solutions
* Deployment .

6. Koje korake je neophodno izvesti za podešavanje razvojnog okruženja NetBeans IDE za rad sa Cloud aplikacijama?
   Preuzeti Oracle Java Clud Services SDK i raspakovati ga na nekoj lokaciji u računaru. Preuzeti i instalirati NetBeans IDE plugin za Oracle Cloud. Kreiranje naloga za Oracle Cloud (potrebni su podaci o platnoj kartici!). Kreiranje novog cloud-a u NetBeans-u.
7. Nabedite i objasnite preporuke razvoja distribuiranih aplikacija za oblak?
   Preporuka je da se aplikacije razvijaju na lokalnom računaru. Prednosti ovog načina razvoja su:
   * aplikacije se na lokalnom serveru pokreću velikom brzinom,
   * inkrementalni razvoj Java EE aplikacija je moguć samo lokalno,
   * upravljanje greškama (debugging) je moguć samo lokalno.
8. Objasnite suštinske razlike prikazanog načina kreiranja JSF veb aplikacija u odnosu na standardni pristup?
   Aplikacija se izvršava na udaljenom (distribuiranom) serveru, dok se kod standardnog pristupa aplikacija izvršava na lokalnom serveru.
9. Šta je whitelist?
   Whitelist je lista podržanih metoda koje su dozvoljene za korišćenje od strane Oracle Cloud-a.
10. Kakvu podršu daje razvojno okruženje NetBeans IDE u radu sa listama dozvoljenih metoda?
    Razvojno okruženje NetBeans IDE poseduje mehanizme za informisanje programera u vezi sa implementacijom nepodržane metode tokom procesa kodiranja. Ukoliko se pokuša korišćenje metode koja nije dozvoljena, ona će biti precrtana.
11. Objasnite kreiranje i registrovanje lokalnih izmena na veb aplikaciji, a zatim i njihovo prosleđivanje na udaljeni repozitorijum.
    U NetBeans IDE-u je potrebno preuzeti i instalirati plugin za Oracle Cloud. Nakon toga je potrebno dodati novi Team server (Team - Team Server - Add Team Server...). Dodaje se ime servera i URL do servera. Klikom na dugme "OK" Team server je kreiran. Sada je potrebno kliknuti na opciju "Login..." kod kreiranog servera i uneti pristupne podatke. Naredni korak je kreianje projekta na serveru. To se radi odabirom opcije "New Project..." kod servera, nakon čega je potrebno uneti naziv projekta, kratak opis, odabrati security opciju (Private) i opciju Wiki Markup (Textile). U narednom prozoru odabrati opciju "Browse" i odabrati lokaciju gde želimo da se na našem računaru smesti lokalni repozitorijum. Nakon ovog koraka biće kreiran lokalni repozitorijum projekta, u koji možemo smestiti kod neke veb aplikacije, koja će nakon toga biti dostupna na cloud serveru.

LEKCIJA 15 - Rekapitulacija gradiva

1. Šta je projektni zadatak?
   Projektni zadatak je rezultat intervjua sa naručiocem softverskog rešenja ili ličnih analiza i preferencija, ukoliko se softversko rešenje razvija za lične potrebe.
2. Koje informacije o projektu sadrži projektni zadatak?
   Naziv projekta, razvojni tim/programer student, predmetni nastavnik/asistent/rukovodilac projekta, polje projekta, mesto izvođenja projekta, firma/organizacija za koju se projekat radi,

cilj projekta, izlaz koji se dobija projektom, aktivnosti tokom realizacije projekta, tehnologije i alati, hardverski resursi, vremenski plan realizacije projekta, mesto i datum izrade projektnog zadatka.

3. Opišite sadržaj glavnog projekta.
   Glavni projekat mora sadržati sledeće elemente:
   * funkcionalni zahtevi,
   * nefunkcionalni zahtevi,
   * dijagrami (klasni i šema baze podataka),
   * hijerarhija projekta,
   * implementacija baze podataka (skripta),
   * listinzi svih JPA entiteta,
   * listinzi svih JPA kontrolera,
   * listinzi svih JSF stranica.
4. Koji zahtevi moraju da budu opisani i prikazani u glavnom projektu?
   Funkcionalni i nefunkcionalni.
5. Objasnite način dokumnetovanja funkcionalnosti aplikacije?
   Prvo što je potrebno dokumentovati, to je početna stranica aplikacije. Potrebno je prikazati izgled ekranai opisati funkcionalnost. Nakon toga, potrebno je objasniti i ostale stranice i sve to propratiti odgovarajućim snimcima ekrana.
6. Koje kategorije pitanja su zastupljene prilokom razgovora za posao Java programera?
   Java osnove, konstruktori, ključna reč static, nasleđivanje, agregacija i kompozicija, overloading, overriding, ključna reč final, polimorfizam, apstrakcija, paket, rukovanje izuzecima, rukovanje stringovima, ugnježdene klase i interfejsi, garbage collection, ulazno/izlazne vrednosti, serijalizacija, umrežavanje, refleksija, AWT i Swing, internacionalizacija, java zrna, RMI, višenitnost, java kolekcije, JDBC.