Paradigmele organizării proiectului

1. Care-i diferența dintre free-lancing și lucrul în echipă din punct de vedere al tehnologiilor și instrumentelor utilizate?

Freelancer este persoana care activează – lucrează pentru sine însuși , mai degrabă decât pentru o companie , pe când echipele și lucru acestora este specific companiilor . Din punct de vedere a tehnologilor și instrumentelor utilizate diferența între freelancer și echipă sunt anume metodele software implimentate. Freelancer-ul utilizează tehnologii fără a raporta altor persoane la ce etapă de implimentare au ajuns , pe când echipă utilizează metologii ca Agile, Waterfall , Interativă , Spirală .

2. Ce paradigm de dezvoltare a produsului program cunoașteți ? Prin ce ele sunt comune și prin ce se deosebesc una de alta ?

Cele mai cunoscute paradigme sunt : Waterfall , Agile , Dezvoltarea Spirală , Dezvoltarea Iterativă , Dezvoltarea Incrementală . Toate aceste sunt metodolii de dezvoltare a produsului program , utilizare pentru organizarea mai ușoară a task-urilor . Diferența lor fiind însăși implimentare . De ex: Wterfall se implimentează liniar , Agile – ciclic ș.a.m.d.

3. Care sunt fazele de dezvoltare a unui produs program în paradigma dezvoltării 'pas cu pas' (waterfall) ? Ce acțiuni de bază sunt prevăute ?

Fazele de dezvoltare e metodologiei Waterfall sunt : Cerințele , Proiectarea , Implimentarea , Testarea și Întreținerea . Rezultatul fiecărei faze este unul sau mai multe documente care trebuie aprobate, iar fiecare următoare fază trebuie să înceapă decât la finalizarea etapei anterioare. Aceată metodă se implimentează când cerințele sunt bine înțelese și este puțin probabilă schimbarea în timpul dezvoltării.

4. Care sunt fazele de dezvoltare a unui produs program în paradigma dezvoltării incrementale (Rational Unified Process - RUP) ? Ce acțiuni de bază sunt prevăute ?

Dezvoltarea incrimentală se bazează pe ieea dezvoltării unei implimentări inițiale , epunând acest lucru la feedback-ul utilizatorilor și evoluând-o prin mai multe versiuni până la dezvoltarea unui system acceptabil .Activitățile unui proces nu sunt separate , ci se împletesc cu feedback-ul implicat în aceste activități. Fiecare increment de system reflect o component a funcționalității care este nevoie de client.

5. Care sunt fazele de dezvoltare a unui produs program în paradigma dezvoltării 'agile' ? Ce acțiuni de bază sunt prevăute ?

Metodologia Agile se referă la un grup de modele de dezvoltare software bazate pe abordarea incrementală și iterative .Aceasta implică clienții în procesul de dezvoltare pentru a propune modificări de cerințe , ce duc la reducerea de comunicări informale , documentații . Acest tip de metodologie este cel mai potrivit pentru aplicațiile în care cerințele se schimbă rapid în timpul procesului de dezvoltare . Există o serie de metode diferite agile : Scrum , Crystal , AM (Agile Modaling) ș.a.

6. Comparați paradigmea dezvoltării a unui produs program 'pas cu pas' cu cea iterativă. Care-s avantajele și dezavantajele fiecărei ?

Paradigma pas cu pas sau așa numita paradigm Waterfall este bazată pe implimentare liniară , pe când cea iterativă se bazează pe dezvoltarea unui sistem prin construirea unor porții mici cu toate caracteristicele pe toate componentele .

Printre cele mai mari beneficii ale tehnicii Waterfall se numără:
- structura clară și simplă a procesului de dezvoltare - aceasta reduce pragul de intrare pentru echipe
-reportare convenabilă - puteți urmări cu ușurință resursele, riscurile, timpul petrecut și finanțele datorită etapelor stricte ale procesului de dezvoltare și documentația detaliată a proiectului
-stabilitatea sarcinilor - sarcinile cu care se confruntă produsul sunt clare pentru echipă încă de la începutul dezvoltării și rămân neschimbate pe parcursul întregului proces
-evaluarea costului și a termenelor pentru proiect - calendarul lansării produsului finit, precum și costul total al acestuia pot fi calculate înainte de începerea dezvoltării.
Printre dezavantajele metodei cascadei se numără:
- proces fără flexibilitate - deci dacă proiectul necesită mai mult timp și resurse financiare decât este posibil, atunci faza de testare va trece sub cuțit.
- „rezistență" la schimbări - cadrul rigid din etapele de dezvoltare și condiția de a furniza numai produsul finit determină incapacitatea de a face modificări în timpul dezvoltării
- inerție - în primele etape, prognoza timpului și a cheltuielilor financiare se pot schimba în direcția creșterii, dar este imposibil să schimbați proiectul în direcția optimizării costurilor, schimbând funcționalitatea sau conceptul înainte de lansarea produsului finit.

Avantajele unei abordări iterative:
• reducerea impactului riscurilor grave în primele etape ale proiectului, ceea ce duce la reducerea costurilor eliminării acestora;
• organizarea feedback-ului efectiv al echipei de proiect cu consumatorul (precum și clienții, părțile interesate) și crearea unui produs care să răspundă cu adevărat nevoilor sale;
• concentrarea pe cele mai importante și critice domenii ale proiectului;
• testare iterativă continuă pentru a evalua succesul întregului proiect în ansamblu;
• detectarea timpurie a conflictelor dintre cerințe, modele și implementarea proiectului;
• încărcare mai uniformă a participanților la proiect;

Dezavantaje: - O înțelegere holistică a capacităților și limitărilor proiectului lipsește de foarte mult timp. -Iteratia trebuie sa arunce o parte din munca depusa mai devreme. -conștiența specialiștilor atunci când efectuează munca este încă redusă, ceea ce este explicabil din punct de vedere psihologic, deoarece sunt dominați în mod constant de sentimentul că „oricum, totul poate fi refăcut și îmbunătățit ulterior

7. Comparați paradigmea dezvoltării a unui produs program incrementală cu cea 'agile'. Care-s avantajele și dezavantajele fiecărei ?

Paradigma agile se bazează pe implimentarea ciclică pe când cea incrementală se bazează pe idee dezvoltării unei inclimentări inițiale .
Model de proces incremental
Beneficiile
-De obicei este mai ușor de testat și depanat decât alte metode de dezvoltare software, deoarece în fiecare iterație se fac modificări relativ mici.
- Livrarea inițială a produsului este mai rapidă și costă mai puțin.
-Model adaugă flexibilitate modelului de bază al cascadei, adăugând procese iterative incrementale.
-Fiecare funcție a sistemului poate fi considerată ca un sistem separat și dezvoltată în conformitate cu modelul unei cascade (proiectare modulară)
- Livrare anticipată a pieselor funcționale de utilizat.
-Clientul are posibilitatea de a-și oferi feedback-ul și evaluarea sistemului în diferite etape de timp, pe măsură ce sistemul se dezvoltă treptat.

-Le caracteristici și funcționalitatea celei mai mari priorități ale sistemului sunt incluse în livrările anticipate.
-În cursul proiectului, este posibilă re-prioritizarea

deficiențe
-Costurile de dezvoltare totală sunt mai mari decât alte modele
- Pe măsură ce se adaugă funcționalitate suplimentară produsului, pot apărea probleme cu arhitectura sistemului care nu era evidentă în prototipurile anterioare.
-Timpul de livrare total al întregului sistem este mai lung.
- Succesul dezvoltării întregului sistem depinde de planificarea corespunzătoare a priorităților între componentele sistemului. Planificarea necorespunzătoare poate duce la dezastru.

agile
avantaje:
-Există multe metode de dezvoltare flexibile; majoritatea reduc la minimum riscul dezvoltând software într-un timp scurt.
-Una iterație într-un model de dezvoltare flexibil este software-ul dezvoltat într-o unitate de timp.
-Fiecare iterație este un întreg proiect software care include toate etapele care există în model: planificare, analiza cerințelor, proiectare, codare, testare și documentare.
-Iterația nu adaugă neapărat suficientă funcționalitate pentru a garanta lansarea produsului pe piață. Dar, la sfârșitul iterației, ar trebui să existe o eliberare accesibilă (nicio eroare).
-La sfârșitul fiecărei iterații, echipa de dezvoltare reevaluează prioritățile proiectului.
- Metodele flexibile pun accent pe comunicarea personală pe documente scrise.
-utilizarea unor abordări flexibile poate îmbunătăți calitatea rezultatelor;
-obiectivele dorite sunt atinse mult mai rapid și cu un efect mai mare, economisește timp și resurse;
-Compania dezvoltă capacitatea de a se adapta bine la schimbări (inclusiv pe cele neașteptate) și la condiții competitive;
- proiectele create sunt planificate și controlate mai atent;
- compania este capabilă să livreze produse care sunt cu adevărat așteptate și căutate de clienți.
dezavantaje:
-stimularea schimbărilor constante ale proiectului: flexibilitatea dezvoltării produsului poate duce la faptul că nu va ajunge niciodată la versiunea finală.
- cerințe sporite pentru calificarea și experiența echipei:
- dificultatea calculării cantității totale de muncă: stimularea modificărilor și îmbunătățirea produsului final duce la o valoare flotantă a costului proiectului.

8. Ce roluri din cadrul unui proiect de dezvoltare soft cunoașteți ? Ce funcții de bază ele îndeplinesc ?

Cele mai des definite roluri în cadrul unui proiect sunt: Project manager se ocupă cu coordonarea oamenilor, proceselor și însăși a produsului și poiectului ; Project analyst – persoana care se procupă de documente , aprobă cerințele ; Project architect se ocupă de coordonarea tehnică a activităților și a artefactelor ; Project developer implimentează , integrează , design și analiză ; Project tester are funcția de a descoperi defectele , precepe calitatea .

9. Ce-I comun și ce-i diferit între rol de proiect și angajat ? Cum este rezolvată diferența dintre rol și angajat ?

În rol de proiect persoana în cauză impreună cu alte persoane urmăresc acelaș scop și îndeplinesc un rol,pe când un angajat deține mai multe roluri . Însăși rolul în proiect este ce fel de îndatoriri are de fapt angajatul și ce îndeplinește exact. Rolul nu coincide întotdeauna cu poziția. Poziție - Obligație oficială oficială, poziție oficială legată de îndeplinirea anumitor îndatoriri. De exemplu, dacă proiectul și echipa sunt mici, fiecare membru al echipei poate îndeplini mai multe roluri simultan.

Cerințele către produsul program și gestionarea lor

1. Care-i esența termenului ‘Stakeholder’ ? Care-I rolul lui în proiect ? Numiți careva personae, aferente proiectului, care pot fi numite ‘Stakeholder’.

Stakeholder - o persoană interesată de eficiența proiectului. Rolul părților interesate: calitate, managementul riscului, procesul de definire a cerințelor. O organizație trebuie să efectueze audituri periodice ale planurilor de asigurare a calității proiectului. Pentru fiecare proiect sunt stabilite diferite obiective de calitate, care la rândul lor se bazează pe cerințele părților interesate.
În standardul privind procesele ciclului de viață a sistemelor software, sarcina de a determina cerințele unui Stakeholder este formulată astfel: determinarea cerințelor sistemului, a căror îndeplinire poate furniza serviciile necesare utilizatorilor și altor stakeholderi într-un mediu de aplicație dat.
Stakeholder poate fi: proprietarul produsului (clientul), dezvoltatorul (membru al echipei care dezvoltă proiectul).

31 Ce reprezintă modelele de dezign (design patterns) și ce ele oferă la implementarea unui produs program ? Ce categorii de modele de design cunoașteți ?

În inginerie software, un model de dezign este o soluție generală repetabilă la o problemă care apare frecvent în proiectarea software. Un model de design nu este un design final care poate fi transformat direct în cod. Este o descriere sau un șablon pentru a rezolva o problemă care poate fi utilizată în multe situații diferite.
Tipuri de modele de design: Creationale , Strucurale, Behavioral(comportamental)

32 Care-I rolul produselor terțe în implementarea unui produs program ? Care-I modul de utilizare a lor ?

Integrarea terților vă permite să dezvoltați aplicații personalizate care se conectează direct cu software-ul dvs. terț, astfel încât să puteți adăuga funcții sau să vă îmbunătățiți procesele de afaceri.

33 Ce impact are infrastructura proiectului asupra procesului de elaborare ?

Modul în care organizația dvs. este structurată influențează modul în care gestionați și conduceți proiecte. De asemenea, poate influența câtă autoritate și abilitate trebuie să-ți faci treaba ca manager de proiect. Există trei structuri organizaționale comune, iar managerii de proiect lucrează în toate: funcțional, proiect și matrice.

34 Care-I destinația versionării componentelor proectului ?

Controlul versional este gestionarea modificărilor la documente, programe de calculator. Modificările sunt de obicei identificate printr-un număr sau cod de literă, denumite „număr de revizuire”, „nivel de revizuire” sau pur și simplu „revizuire”. De exemplu, un set inițial de fișiere este „revizuirea 1”. Când se face prima modificare, setul rezultat este "revizuirea 2" și așa mai departe. Fiecare revizuire este asociată cu o oră de timp și persoana care efectuează schimbarea. Revizuirile pot fi comparate, restaurate și cu unele tipuri de fișiere, combinate.
Controlul versiunilor este important pentru echipele de dezvoltare. Trebuie să facă mai mult decât să gestioneze și urmarirea fișierelor. Acest lucru este important în special pentru echipele care practică DevOps.

35 Care-I rolul prototipării unui produs program și cînd se recurge la ea ?

Prototipul software este activitatea de creare a prototipurilor de aplicații software, adică versiuni incomplete ale programului software în curs de dezvoltare. Este o activitate care poate apărea în dezvoltarea de software Un prototip este o versiune de proiect a unui produs care permite să exploreze ideile și vizualizare spatele unei caracteristici sau a conceptului general de proiectare înainte de a investi timp și bani în dezvoltare.

36 Prin ce diferă un build al produsului program de un release ?

Un release este lansarea oficială a produsului către clienții săi. Build este atunci când este testată insasi produsul și certificată de echipa .Un build poate fi respins de către echipa de testare dacă vreunul dintre teste nu reușește sau nu îndeplinește anumite cerințe.

37 Ce este comun și prin ce diferă integrarea pe faze, integrarea incrementală și integrarea continuuă a produsului program ?

Scopul lor este același: să faca procesul de dezvoltare și lansare a software-ului mai rapid și mai automatizate. Diferența cheie între cele trei este în domeniul de aplicare al automatizării applicate.
Integrare pe faza (CI): ramuri de caracteristici de scurtă durată, echipa se îmbină cu filiala principală de mai multe ori pe zi, proces automatizat de construire și testare care oferă feedback în 10 minute; desfășurarea este manuală.
Integrarea incrementata(CD): CI + întregul proces de lansare a software-ului este automatizat, iar implementarea către producție este manuală.
Implementare continuă: implementare CI + CD + complet automatizată la producție.

38 Care-I perioada de integrare a produsului program, pe care o preferați și de ce ?

As allege incremental integration
În testarea de integrare incrementală, dezvoltatorii integrează modulele unul câte unul folosind butucuri sau drivere pentru a descoperi defectele. Această abordare este cunoscută sub numele de testare de integrare incrementală.

39 Ce instrumente separate și integrate propuneți spre utilizare la implementarea unui produs program ?

IDE: MS VisualStudio **Visual Studio** include un set complet de instrumente de dezvoltare pentru generarea de aplicații ASP.NET, Servicii Web XML, aplicații desktop și aplicații mobile.
Alte instrumente necesare sunt:
Editors • Compilers • Linkers • Debuggers

40 În ce constă problema mediilor 'host-target' la implementare produsului program ?

Host target development este atunci , cind dezvoltam aplicatea in IDE pe o masina de executie, dar lansarea se efectueaza pe alta masina (Windows-Linux)
Probleme apar la necesitatea compilării încrucișate și a bibliotecilor țintă asociate. Sistemul de fișiere trebuie să fie organizat cu atenție, astfel încât să se păstreze bine separarea compilatoarelor native și încrucișate. Necesita interconectarea între cele două sisteme, pentru descărcarea fișierelor executabile și controlul sistemului țintă în timpul depanării

Medii virtuale de soft

9. Ce reprezintă serviciile virtuale în cloud ?

Un serviciu cloud este orice serviciu pus la dispoziția utilizatorilor la cerere prin Internet de pe serverele unui furnizor de calculatoare cloud, spre deosebire de furnizarea de la serverele proprii ale companiei. Serviciile cloud sunt concepute pentru a oferi acces ușor, scalabil la aplicații, resurse și servicii și sunt gestionate complet de un furnizor de servicii cloud.

10.Ce provideri și servicii virtuale în cloud cunoașteți ?

Există mulți provideri de servicii virtuale în Cloud pe piață:

1. **Amazon Web Service (AWS)**
2. Microsoft Azure
3. Google Cloud Platform
4. IBM Cloud Services
5. Adobe Creative Cloud
6. Kamatera
7. VMware
8. Rackspace
9. Red Hat
10. **Salesforce**
11. Oracle Cloud
12. SAP
13. Verizon Cloud
14. Navisite
15. Dropbox

Aceștia utilizează servicii virtuale în cloud cum ar fi :

a) **Software as a service (SaaS)**

SaaS este un serviciu de cloud furnizat de compania cloud. În SaaS, un client poate să ofere un software care poate fi pentru o anumită perioadă de timp, fie pentru întreaga viață. SaaS utilizează internetul și livrează cererea clientului. Cea mai mare parte a aplicației SaaS nu necesită descărcări, deoarece acestea pot fi folosite direct prin browser-ul web.

b) Platform as a Service (PaaS)

Paas este un framework pentru dezvoltatori unde ei pot crea o aplicație pentru particularizarea unei precedente aplicație deja construită. Acest serviciu, de asemenea, este furnizat prin intermediul Internetului și aici toată gestiunea se face de către întreprindere sau orice furnizor terț.

c) Infrastructure as a Service (IaaS)

Iaas ajută clientul să acceseze și să monitorizeze lucruri cum ar fi computer, rețele și alte servicii. În IaaS, Clientul poate achiziționa resurse la cerere, mai degrabă decât să cumpere hardware, care este costisitor și greu de întreținut.

Depanarea si testarea produsului soft

1. Care este scopul testării produsului soft ? Ce plus valoare ea oferă ?

Testarea software reprezintă o investigație empirică realizată cu scopul de a oferi părților interesate informații referitoare la calitatea produsului sau serviciului supus testării[1], luând în considerație contextul operațional în care acesta din urma va fi folosit. Testarea software pune la dispoziție o viziune obiectivă și independentă asupra produsului în dezvoltare, oferind astfel businessului posibilitatea de a înțelege și evalua riscurile asociate cu implementarea produsului soft. Tehnicile de testare includ, dar nu sunt limitate la, procesul de execuție a programului sau aplicației în scopul identificării defectelor/erorilor de software. Testarea software mai poate fi definită ca un proces de validare și verificare a faptului că un program/aplicație/produs software (1) corespunde business cerințelor și cerințelor tehnice care au ghidat proiectarea și implementarea lui; și (2) rulează și se comportă corespunzător așteptărilor.

2. Care sunt pașii de bază a procesului de testare a softului ?

    1. Test Planning
    2. Test Analysis and Design
    3. Test Execution
    4. Test Evaluating

3. Ce metodologii ale testării funcționale conoașteți ? Care-i ordinea aplicării lor ?

    1. Unit Testing
    2. Integration Testing
    3. System Testing
    4. Acceptance Testing

4. Ce metodologii ale testării nefuncționale conoașteți ?

    1. Performance Testing
    2. Security Testing
    3. Usability Testing
    4. Compatibility Testing

5. Ce este 'Unit Testing' și ce avantaje și dezavantaje el aduce cu sine ?

    Unit Testing este o metodologie a testării funcționale, aceasta :

    - Verifică - dacă modulele/ componentele funcționează corect
    - Deobicei UT este scris de dezvoltatori în aceiași prog.language cu aceleași dev.tools.
    - În TDD (Agile, Scrum, XP) UT este scris înainte de module.
    - Include sprijin pentru gestionarea și execuția testului unităților automate

Avantaje : Scopul metodei Unit Testing este de a izola fiecare parte a programului și de a arăta că părțile individuale sunt corecte.

- Unit Testing găsește problema la începutul ciclului de dezvoltare.
- Procesul de redactare a unui set complet de teste îl obligă pe autor să gândească prin intrări, ieșiri și condiții de eroare, definind astfel mai clar comportamentul dorit al unității.
- Costul găsirii unei erori înainte de începerea codificării sau când codul este prima dată scris este considerabil mai mic decât costul de detectare, identificare și corectare a erorii ulterior

Dezavantaje:

- Bug-urile din codul lansat pot cauza, de asemenea, probleme costisitoare pentru utilizatorii finali ai software-ului
- Nu va prinde erori de integrare sau erori mai largi la nivel de sistem (cum ar fi funcții efectuate pe mai multe unități sau zone de test nefuncționale.

6. Care-i scopul la 'Integration Testing', cine și cînd îl rulează ?

Principala funcție sau obiectiv al acestei testări este testarea interfețelor dintre unități / module.
În mod normal, facem testarea de integrare după „Testarea unității” Odată ce toate unitățile individuale sunt create și testate, începem să combinăm acele module „Tested Unit” și începem să facem testarea integrată.
Principala funcție sau obiectiv al acestei testări este testarea interfețelor dintre unități / module.
Modulele individuale sunt testate pentru prima dată în mod izolat. Odată testate modulele, acestea sunt integrate unul câte unul, până când toate modulele sunt integrate, pentru a verifica comportamentul combinațional și a valida dacă cerințele sunt implementate corect sau nu.
Aici ar trebui să înțelegem că testarea integrării nu se întâmplă la sfârșitul ciclului, ci se realizează simultan cu dezvoltarea. Deci, de cele mai multe ori, toate modulele nu sunt de fapt disponibile pentru a fi testate

7. Care-i scopul la 'System Testing', cine și cînd îl rulează ?

System Testing este o testare în care sistemul în ansamblu este testat, adică toate modulele / componentele sunt integrate împreună pentru a verifica dacă sistemul funcționează așa cum este de așteptat și nu apar probleme din cauza modulelor integrate.Aceasta este rulată la sfârșitul tuturor procedurilor de testare , fiind ca un punct de decizie.

Bazele aplicatiilor Web

1. Ce este o aplicatie Web ?

O aplicație web este un program care rulează într-o arhitectură client-server folosind tehnologiile deschise World Wide Web (PHP, ASP, PEARL, PYTHON,HTML, CSS, JAVASCRIPT, etc.). Ele înlocuiesc modelele în care atât serverul cât și clientul rulează tehnologii proprietar, mentenanța aplicațiilor de pe partea de client fiind prea complexă, costisitoare și susceptibilă la erori.

2. Clasificati va rog aplicatiile Web.

   Statice - conținutul pe server este static și oferă un set de fișiere deja definite (imagini, video, audio, applet) încorporate în HTML.
   Dinamice - conținutul este generat de server în funcție de unele variabile definite în cererea HTTP.

3. Ce este browser? Ce standarde un browser tipic poate prelucra ?

   Un browser este o aplicație software (program) ce permite utilizatorilor să afișeze text, grafică, video, muzică și alte informații situate pe o pagină din World Wide Web, dar și să comunice cu furnizorul de informații și chiar și ei între ei.
   Navigatoarele web funcționează pe baza anumitor protocoale HTTP,, HTTPS și FTP.

4. Ce protocol este folosit la nivel de aplicatie pentru comunicare intre client si server al unei aplicatiei Web ?

   HTTP

5. Ce este server si ce este client in contextul unei aplicatii Web ?

   Un server web este un sistem informatic care procesează solicitări prin HTTP, protocolul de rețea de bază utilizat pentru distribuirea informațiilor pe World Wide Web.

   Client se referă la tot ce se află într-o aplicație web care este afișată, ceea ce utilizatorul vede, cum ar fi textul, imaginile și restul utilizatorului, limbajele precum HTML și CSS sunt interpretate de browser pe partea de client,

Aplicatii Web statice

1. Ce este HTML? Pentru ce se foloseste ?

   HTML, limbajul de modelare folosit pentru crearea de documente pe World Wide Web, care definește structura și aspectul unui document Web folosind o varietate de etichete și atribute.

2. Ce este CSS? Pentru ce se foloseste ?

   CSS stilizează paginile web, interacționând cu elemente HTML, specificând aspectele paginii, culorile și fonturile.

3. Ce este JavaScript? Clasificati acest limbaj.

JavaScript este una dintre tehnologiile de bază ale World Wide Web, activează paginile web interactive și este o parte esențială a aplicațiilor web, este utilizat pentru a îmbunătăți paginile HTML și redă paginile web într-o manieră interactivă și dinamică.

Aplicații Web Dinamice

1. Care este rolul serverului Web în aplicații dinamice?

   Rolul serverului Web în aplicații dinamce este de a grupa și oferi către client date, conținut, etc.
   El lucrează la nivel de protocol HTTP, ceea ce permite comunicarea cu o mulțime de medii care folosesc HTML, JSON,REST, etc. Serverul Web comunică și mesajele de eroare, sau realizează redirecționări către alți clienți sau servere.

2. Ce este autentificare si autorizare? In ce ordine sunt executate aceste doua operatii ?
   Autentificarea – mecanism verifică dacă este posibil ca un utilizator să utilizeze o platformă, re determină rolul acestuia.
   Autorizarea- mecanism care oferă unui utilizator accesul către o variatetate de servicii, în funcție de rolul acestuia în cadrul unui sistem

3. Descrieți un algoritm de bază a lucrului unei aplicații Web dinamice.
   1. Clientul trimite o cerere
   2. Serverul localizează fișierele
   3. Serverul procesează instrucțiunile și crează conținut
   4. Conținutul este livrat
   5. Brower procesează conținutul și îl afișează

Aplicații Web bazate pe Web Servicii

1. Ce este un serviciu Web ?
   Un serviciu Web este un mediu standardizat care permite propagarea comuncației între un client si un server .
   Principalele componente ale unui serviciu web este XML.
   Serviciile web sunt invocate în funcție de cererile utilizatorului, Respectiv cînd aplicațiile comunică prin intermediul XML.

2. Ce standarde pot fi folosite pentru prezentarea obiectelor in servicii Web ?
   JSON – JavaScript Object Notation
   XML - Extensible Markup Language

3. Ce este SOAP ?
   SOAP (Simple Object Access Protocol)
   SOAP este cunoscut ca un protocol de transport independent, care se bazează pe transferul de fișiere XML. XML va implimenta o structură specifica, dar nu și conținut. Este de asemenea parte a transferului HTTP

4. Ce este REST? Descrieti principiile de baza a web serviciului care urmareste stilul de proiectare REST.

   REST este folosit pentru a construi Web servicii ușoare de întreținut, mentenabilie și scalabile.

   Principiile de bază:

   RESTful Client-Server – returnează doar cee ace clientul cere

   Stateless – asigură că toate informațiile cerute sunt furnizate de server, și ca el să poată prelucra în mod corespunzător.

   Cache – Deoarece fiecare cerere cerută este independentă, uneori clientul poate să solicite aceeași solicitare, respectiv se verifică cookies

5. Ce este SOAP? Descrieti principiile de baza a web serviciului care utilizeaza SOAP.

   SOAP este un protocol bazat strict pe conceptul de XML și HTTP.
   Principiile de baza :

   - Transport prin HTTP – folosește ca transport pentru mesaje
   - XML information set – XML este caracteristica principal a unui mesaj.
   - SOAP Message blocks - consta din următoarele cîmpuri : Header Element, Body element, Fault element(option)

6. SOAP vs REST – avantajele, dezavantajele si specificul fiecarei modalitati de implemetare serviciilor Web.

Avantaje : SOAP

Neutralitate : SOAP poate fi folosit în orice limbaj

Simplicitate : Mesajele SOAP sunt mesaje simple XML

Scalabilitate : SOAP folosește HTTP , ceea ce il face unul scalabil

Dezavantaje SOAP:

Lent : XML trebuie să fie deserializat la fiecare mesaj, sau raspuns

WSDL Dependent : El depinde de WSDL și nu are un mechanism standardizat pentru descoperirerea dinamică a serviciilor

Avantaje REST :

Scalabilitate : Este caractiristica principal, datorită separări dintre client si server

Flexibilitate și portabilitate : Indiferent de tipul de date transmis, acestea pot fi transmise către un alt server fără pierderi de date

Dezavantaje :

Securitatea datelor : REST nu impune niciun fel de securitate precum SOAP. Acesta este motivul pentru REST este cel mai rău mecanism pentru a fi utilizat pentru serviciile web.

Dezavantaje REST :

Lipsa starii : Majoritatea aplicațiilor web necesită un mecanism de stat.

Sarcina menținerii acestei stări revine clientului, ceea ce face ca aplicația clientului să fie mai grea și dificil de întreținut.