{iT}Factory

Întâlnirea 1

Introducere în Software Testing

Sfaturi generale

- Să tratezi cu seriozitate și profesionalism acest nou obiectiv.
- Cei care își ating obiectivele nu sunt întotdeauna cei mai smart, dar întotdeauna vor fi cei mai muncitori!
- Alocă-ți timp pentru studiu. Rutina dă consistență. Consistența dă excelență.
- Să faci tot posibilul să participi la toate sesiunile live.
- Să lași comentarii explicative în notițele tale, informații pentru tine din viitor.
- Recomand să vizualizezi înregistrarea. Să îți notezi aspectele importante + întrebări pentru trainer pentru ora următoare.
- Să îți faci temele și unde nu reușești singur, să întrebii pe grup. Trainerul va răspunde și vor beneficia și ceilalți cursanți de răspuns.
- Poți chiar să faci un grup doar de studenți și să vă întâlniți o dată pe săptămână să discutați temele împreună. Fiecare va veni cu o perspectivă nouă și în final toți vor avea de câștigat.
- În timpul orelor, să ai curaj să pui întrebări când ceva nu e clar.

Reguli curs

- Va exista un sheet de prezență.
- În cadrul acestuia îți vei asuma și noțiunile învățate. Nu trecem mai departe până nu își asumă toți cursanții noile concepte.
- Temele se vor adăuga în Folderul grupei, unde vei crea un folder cu numele tău. Vei primi feedback la aceste teme.
- Temele vor fi împărțite în 2 categorii:
  - Obligatorii (se pot face doar cu noțiunile învățate la clasă);
  - Opționale (acestea vor fi mai avansate și necesită poate informare suplimentară). Acest lucru îi va motiva și pe cei care au mai mult timp și le place să se aventureze prin task-uri mai dificile.
- Te rog să mă întrerupi oricând ai întrebări. Doar așa îmi pot da seama unde trebuie să mai insist cu explicații/ exemple.
- Te rog să intri cu 3 minute mai devreme la curs în caz că apar probleme tehnice. Astfel putem profita la maxim de cele 2 ore alocate.
- Dacă nu poți intra, sau dacă întârzii, anunță trainerul pe grup.

{iT}Factory

Obiective principale

Până la final TOȚI* veți avea:

- **Cunoștințe solide despre bazele Testării Software și vei avea cunoștințele necesare pentru pregătirea susținerii examenului ISTQB.**
- **Vei înțelege tehnicile de testare și vei avea capabilitatea de a le explica și pune în aplicare.**
- **Vei cunoaște diferențele dintre testare statică și testare dinamică.**
- **Vei cunoaște diferențele dintre whitebox testing și blackbox testing.**
- **Vei înțelege diferitele tipuri de tehnici de testare.**
- **Vei înțelege cum se testează un website.**
- **Vei înțelege cum se folosește un test management tool.**
- **Vei putea face un proiect final de testare manuală cu ajutorul căruia vei pune în aplicare noțiunile predate de trainer la clasă.**
- **Noțiuni de bază despre API testing: testarea manuală - cum comunică un website cu server-ul.**
- **Cunoștințe ale bazelor de date relaționale - mySQL: Curs baze de date.**

*** toti cei care sunt activi, implicați, își fac temele, dedică timp pentru studiu individual și pun întrebări trainerului vor atinge aceste obiective.**

{iT}Factory

Obiective secundare

Nu fac parte din curricula cursului LIVE, dar îți punem la dispoziție materiale extra ca să ai un avantaj la interviuri. Sfatul meu e să te concentrezi pe ele doar după cursul live. Să nu fii copleșit de noile informații.

- Cunoștințe teoretice despre testarea manuală - acces la o platformă mobilă - link acces https://bit.ly/38vON7b.
- Cunoștințe teoretice despre bazele programării - Primii Pași în Programare (Cursul primit la webinar).
- Capacitatea de a construi un mic brand personal (Curs Portofoliu Wordpress). Trebuie să ai:
    - Website propriu prin care angajatorul să te cunoască pe tine și munca ta;
    - CV european în limba Engleză;
    - Profil LinkedIn;
    - Github public (un loc în cloud unde se pune codul scris de tine);
    - Vei primi feedback dacă ne trimiți un email cu ele la hello@itfactory.ro.

Obiective Întalnire 1

- Să înțelegem **de ce avem nevoie de testare.**
- Să înțelegem **de ce apar defectele.**
- Să înțelegem **rolul testării** în procesul de dezvoltare.
- Să înțelegem **când ne oprim din testat.**
- Să înțelegem care este influența **testării asupra calității.**
- Să cunoaștem **activitățile și obiectivele testării.**
- Să știm care este **procesul de testare.**
- Să înțelegem **principiile testării.**
- Să cunoaștem **nivelurile și psihologia testării.**
- Să știm care sunt **elementele unui bug report.**
- Să înțelegem **conceptele importante din testare.**

De ce avem nevoie de testare?

Contextul sistemelor software

Cauzele defectelor

Rolul testării în procesul de dezvoltare

Când ne oprim din testare?

Influența testării asupra calității

Contextul sistemelor software

Sistemele software sunt prezente peste tot în viața de zi cu zi

Fiecare persoană a întâlnit măcar o dată o situație în care un produs software nu a funcționat

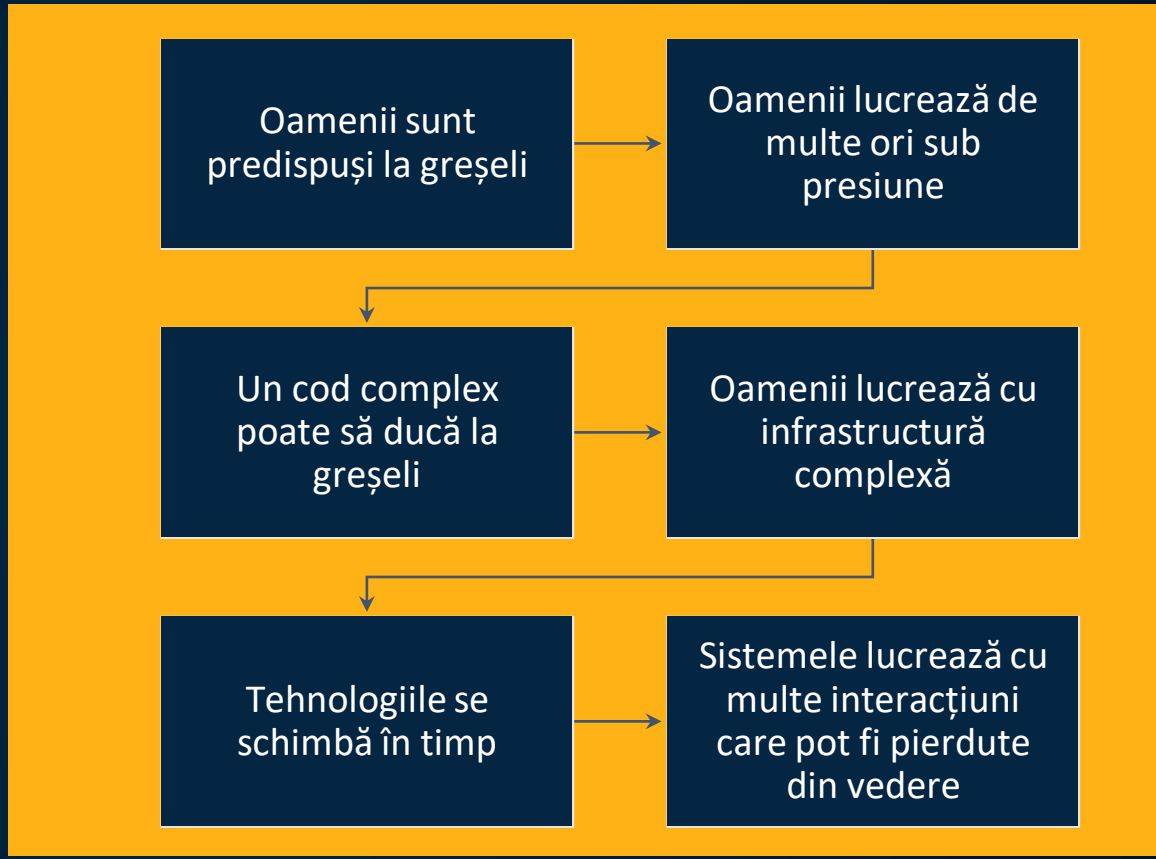
Un produs software care nu funcționează poate cauza pierderi de timp, pierderi financiare sau chiar pierderea sănătății

În cazuri extreme poate duce chiar la pierderea vieții

Cauzele defectelor

- O persoană poate face **o eroare (greșeală)** care produce **un defect** (care mai e numit și **fault sau bug**) în cod sau în documentație.
- Dacă un cod care conține un defect este executat, sistemul poate ajunge în incapacitatea de a face ceea ce ar trebui să facă sau în situația de a face ce nu trebuie.
- *Orice defect POATE să cauzeze un failure, dar NU toate defectele VOR FACE asta.*

De ce apar defectele?



{iT}Factory

Rolul testării în procesul de dezvoltare

O TESTARE RIGUROASĂ A CODULUI ȘI A DOCUMENTAȚIEI POATE REDUCE RISCUL APARIȚIEI PROBLEMELOR ÎNTR-UN MEDIU OPERAȚIONAL

TESTAREA CONTRIBUIE LA CREȘTEREA CALITĂȚII SISTEMELOR SOFTWARE

DACĂ DEFECTELE GĂSITE SUNT CORECTATE ÎNAINTE CA SISTEMUL SĂ FIE LANSAT, ATUNCI COSTUL FIXĂRII DEFECTELOR ESTE MAI MIC

CÂND NE OPRIM DIN TESTAT?

- Atunci când decidem să ne oprim din testare trebuie să luăm în considerare atât nivelul de risc, cât și constrângeri cum ar fi bugetul sau timpul.
- Testarea trebuie să furnizeze informații persoanelor implicate astfel încât acestea să poată lua decizii cu privire la lansarea produsului.
- De obicei criteriile de ieșire din procesul de testare se decid în etapa de planificare a procesului de testare.
- Atunci când testerul consideră că se apropie de finalul testării se analizează situația curentă pe baza criteriilor de ieșire.

{iT}Factory

Influența testării asupra calității

Prin intermediul testării este posibil să măsurăm calitatea produsului în funcție de numărul de defecte găsite și fixate.

Atunci când rulăm un test bi definit și acesta este marcat și trecut, atunci putem să considerăm că nivelul genera de risc s-a diminuat.

În timpul testării trebuie să se tinem cont de concluziile rezultatelor proiectele anterioare.

Testarea ar trebui să fie integrată drept una din activitățile procesului de Quality Assurance.

Activități legate de procesul de testare

Activități efectuate înainte și în timpul procesului de testare

- Planificare și monitorizare;
- Analiza documentației și a codului;
- Crearea condițiilor de test;
- Crearea și executarea cazurilor de testare.

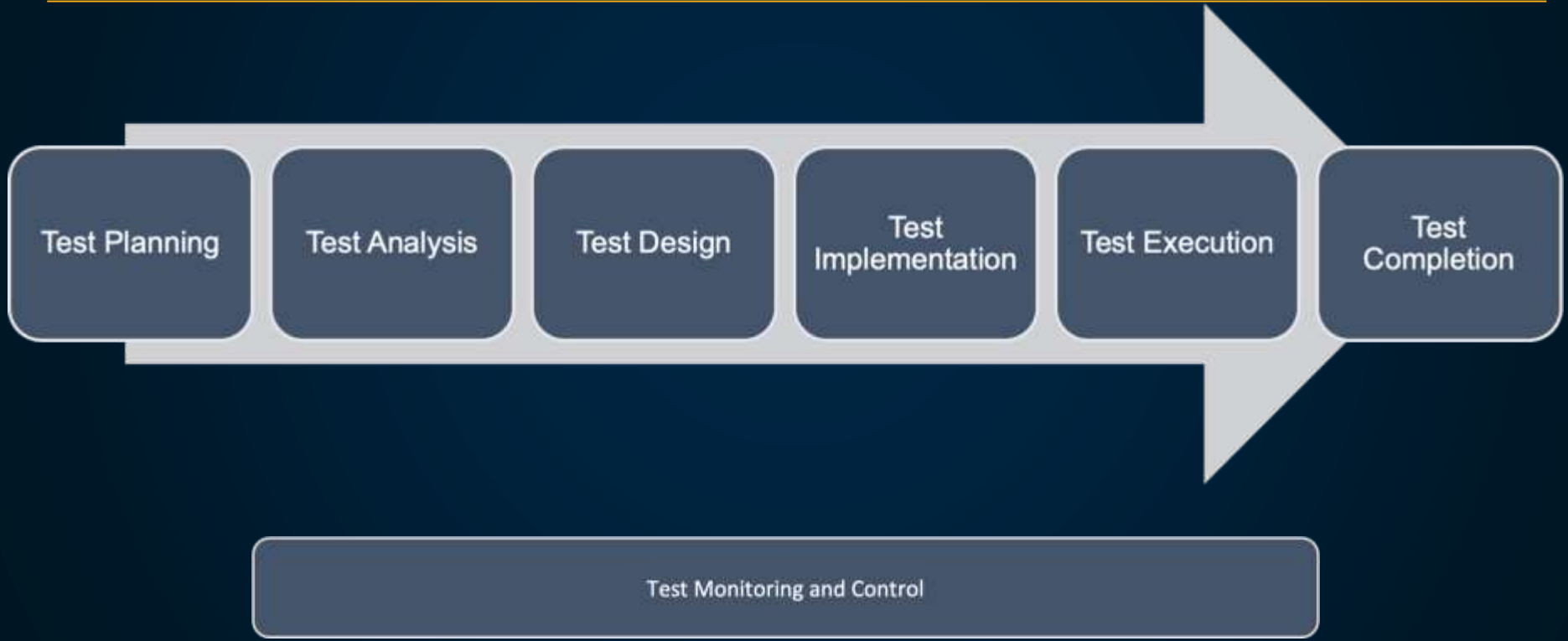
Activități efectuate după încheierea procesului de testare

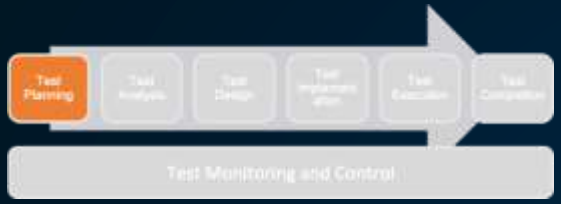
- Verificarea rezultatelor și evaluarea îndeplinirii condițiilor de încheiere a testarii.
- Crearea de rapoarte de încheiere.
- Închiderea procesului de testare.

Obiectivele Testării

1. Găsirea defectelor
2. Creșterea încrederii în calitatea produsului
3. Furnizarea de informații utile în luarea deciziilor
4. Prevenirea defectelor

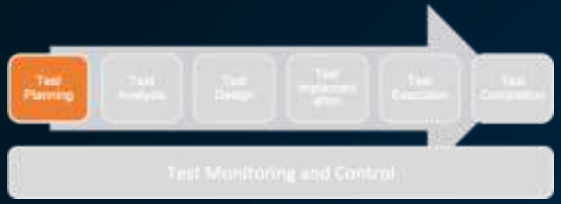
PROCESUL FUNDAMENTAL AL TESTĂRII





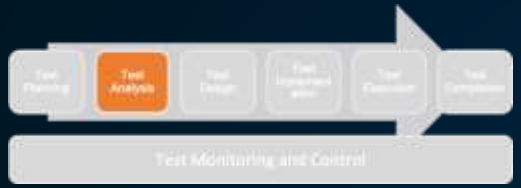
Planificare

- Include activități care definesc obiectivele și abordarea cu privire la testare
- În aceasta etapa se decide care parti ale aplicației se dorește a fi testate
- Se aloca roluri pentru persoanele care vor fi implicate în proiect
- Se definesc criteriile de intrare și criteriile de ieșire
- Se identifica riscurile de proiect inițiale și resursele necesare
- Se programează activitățile și etapele procesului de testare
- Se creeaza un plan de testare care va conține informații generale legate de cum se va desfășura procesul de testare
- Se evaluează criteriile de intrare



Monitorizare si Control

- Este o activitate continuă care se desfășoară cu scopul de a compara progresul actual cu planul de testare (***monitorizare***)
- Începe o dată cu etapa de planificare și se termina o dată cu etapa de inchidere
- Presupune raportare periodica prin rapoarte de status care sa includa și notificarea oricăror devieri de la plan
- În cazul în care se observa riscul de a nu ne îndeplini obiectivele, se iau măsuri de ***control***
- Planificarea și replanificarea testarii ia în considerare feedback-ul obținut atat din etapa de monitorizare și control cât și din celelalte etape ale procesului de testare



Analiză

- Răspunde la întrebarea: “Ce urmează sa testam”?

- In aceasta etapa se analizeaza documentatia primita de la client (cerinte de business, specificatii de design etc) pentru a ne asigura ca le înțelegem, ca nu exista greseli, ambiguitați, inconsistente, neconcordante, contradictii etc. Putem de asemenea sa facem sugestii de îmbunătățire asupra cerințelor, iar clientul va decide dacă sugestiile noastre sunt bune sau nu. Defectele care sunt găsite în cerințe sunt mai ușor de fixat și mai ieftin decat dacă ar fi găsite în cod ***(principiul 3 al testarii - Early Testing)***

- De asemenea, în aceasta etapa se genereaza condițiile de testare (Ce vom testa)

{iT}Factory

Design

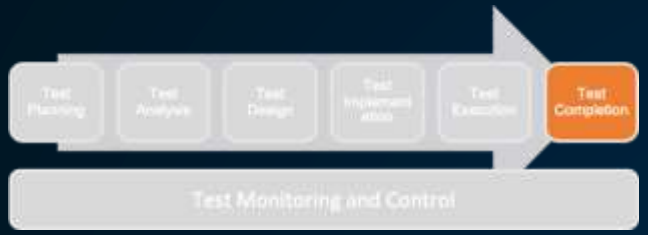
- Răspunde la întrebarea : “Cum vom testa?”
- În aceasta etapa se creeaza cazurile de testare și se prioritizeaza
- De asemenea se identifica datele de testare de care avem nevoie și se face design-ul mediului de testare prin identificarea oricărui tool sau infrastructură de care avem nevoie pentru testare

Implementare

- Răspunde la întrebarea : “Avem tot ce ne trebuie pentru a începe executarea testelor?”
- În aceasta etapa se creeaza datele de testare identificate în etapa anterioara
- Se creeaza mediul de test și tot în aceasta etapa ne asigurăm ca a fost definit in mod corect
- Se prioritizeaza testele (pe baza importanței de business și a riscurilor)
- Se grupează testele pe baza obiectivelor lor (testare functionala, testare de regresie, testare de acceptanta etc)
- Tot în etapa aceasta ne asigurăm ca avem tot ce ne trebuie pentru a începe testarea propriu-zisă (mediu de testare, permisiuni, date de testare, documentatie etc)

Executie

- În aceasta etapa cazurile de testare sunt executate (adică verificam comportamentul produsului software pe baza instrucțiunilor scrise în cazurile de testare)
- Rezultatele sunt raportate în tool-ul în care au fost scrise testele (Passed/Failed/Blocked etc)
- Bug-urile sunt raportate atunci când rezultatele așteptate nu coincid cu rezultatele actuale
- Atunci cand bug-urile sunt fixate, se face retestarea lor pentru a ne asigura că au fost într-adevăr fixate
- Atunci cand codul a fost schimbat (fie pentru fixarea unui bug fie pentru introducerea unei noi functionalitati), se va face si testare de regresie, pentru a ne asigura că schimbările făcute nu au avut un impact negativ asupra functionalitatilor existente



Inchidere

- În aceasta etapa se evaluează criteriile de ieșire pentru a ne asigura că putem sa inchidem procesul de testare în siguranta
- Orice taskuri rămase deschise și buguri sunt re-evaluate si ulterior inchise
- Materialele de testare sunt predate și arhivate ***(handover of testware)***
- Este generat un raport de inchidere a testarii (Care mai e numit si ***test summary report*** sau ***test completion report***) care ulterior este trimis catre stakeholders
- Se analizează lecțiile învățate în urma procesului de testare pentru a evalua schimbările necesare într-un proiect similar viitor

Concepte similare

Fault vs failure

Eroare (greșeală)> defect (fault, bug)> failure

Nu toate defectele cauzează un failure.

Testare vs debugging

Testarea este un process prin care se identifica defectele.

Debuggingul este o activitate specifică programării prin care se analizează cauza defectelor și se găsesc soluții pentru remedierea lor.

Testare vs retestare

Testarea este un process prin care se identifică defectele.

Retestarea este un process prin care se verifică dacă defectele marcate ca și remediate au fost într-adevăr remediate.

Retestare vs Testare de Regresie

Retestarea este un proces prin care se verifică dacă defectele marcate ca și remediate au fost într-adevăr remediate.

Testarea de regresie e un process prin care se verifică programul sau o parte din program pentru a ne asigura că schimbările aduse asupra lui nu au cauzat/descoperit alte defecte.

{iT}Factory

Concepte similare

QA vs QC

Quality Assurance este un proces preventiv care încearcă să se asigure că nu există probleme/nu se introduc erori în timpul procesului de dezvoltare.

Quality Control este un proces reactiv care încearcă să se asigure că produsul finit nu are erori înainte de lansarea la client.

False Positive vs False Negative

False positives înseamnă acel moment când este identificat un bug care de fapt nu e bug. Poate să apară din cauza datelor incorecte, mediului definit incorect etc.

False Negative înseamnă atunci cand nu am găsit un bug pe care trebuia de fapt sa îl găsim.

Testare pozitivă vs Testare Negativă

Testare pozitivă înseamnă testarea sistemului cu valori pe care ar trebui să le poată procesa.

Testare Negativă înseamnă testare cu valori pe care sistemul nu ar trebui să le poată procesa în mod normal pentru a ne asigura ca aceste valori sunt într-adevăr respinse și că nu cauzează un crash al sistemului.

Atentie!

- **false positives** sunt teste care sunt marcate ca și failed (când de fapt ar trebui să fie marcate ca și passed) – **alarmă falsă.**
- **false negatives** sunt teste marcate ca și passed (deși ar trebui să fie failed).

{iT}Factory

Concepte similare

Testare funcțională vs Testare non-funcțională

Testare funcțională (black box) – ***Ce trebuie să facă produsul?*** – Verifică dacă produsul își îndeplinește funcțiile.

Testele funcționale pot fi descrise drept teste scrise pe baza specificațiilor, a use case-urilor și arată ce trebuie să facă produsul.

Testele sunt definite ca acțiuni făcute de către system. Testele funcționale sunt bazate pe bază de funcționalități și comportamente.

Testarea non-funcțională (parameteri) – ***Cum trebuie să se comporte produsul?*** – Verifică atribute care descriu cât de bine își îndeplinește sistemul funcțiile.

e.g. reliability, eficiență, mentenabilitate, transferabilitate, performanță, recuperare, localizare, conformitate etc

Verificare vs validare

Verificare – ***Se creează produsul cum trebuie?*** *Sau* ***Construiesc produsul așa cum trebuie?*** Se concentrează pe corectitudinea materialelor care stau la baza produsului cum ar fi documentație, cod etc.

Validation – Procesul prin care verificăm că produsul finit/o componentă a produsului finit răspunde nevoilor utilizatorului.

Este produsul corect? Construiesc produsul care trebuie?

{iT}Factory

Principiile Testării

- Testarea arată prezența defectelor, nu absența lor
- Testarea exhaustivă este imposibilă
- Testarea timpurie
- Gruparea defectelor
- Paradoxul pesticidelor
- Testarea depinde de context
- Misconcepția absenței erorilor

{iT}Factory

Testarea arată prezența defectelor

TESTAREA VORBEȘTE DESPRE PREZENȚA DEFECTELOR, NU DESPRE ABSENȚA LOR

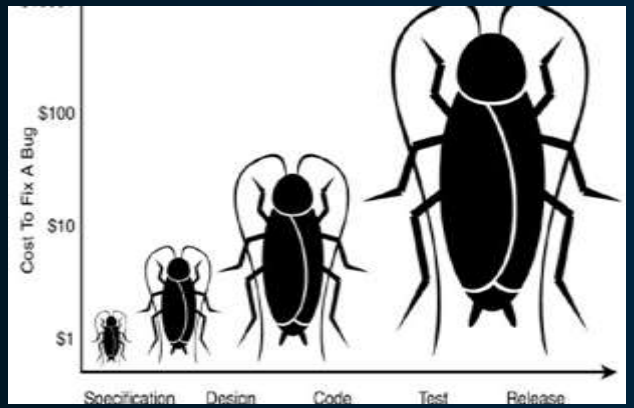
TESTAREA REDUCE PROBABILITATEA DEFECTELOR NEDESCOPERITE RĂMASE ÎN SOFTWARE, DAR CHIAR DACĂ NU SUNT GĂSITE DEFECTE NU ÎNSEAMNĂ CĂ ELE NU EXISTĂ

Testarea exhaustivă este imposibilă

Doar în cazuri triviale este posibil să testăm toate combinațiile posibile dintr-un sistem.

Nu este corect să spunem că un produs software nu are defecte. Ele pot exista în sistem, doar că nu au fost găsite.

Testarea timpurie



Cu cât găsim defectele mai devreme, cu atât ele sunt mai ieftin de fixat.

Un defect găsit în etapa de analiza e cel mai ieftin de analizat, iar unul găsit în producție e cel mai costisitor de fixat.

Gruparea defectelor

Testarea va fi concentrată proporțional în legătură cu densitatea așteptată/ observată a defectelor. În general majoritatea defectelor sunt găsite într-un număr restrâns de componente ale sistemului.

Aceasta așteptare se bazează pe *Principiul Pareto* (care mai este cunoscut drept principiul 80/20) care afirmă că, pentru majoritatea evenimentelor, 80% din efecte derivă în urma a 20% din cauze.

Principiul acesta permite setarea priorităților și facilitează organizarea timpului, și, în consecință permițând maximizarea rezultatelor într-un timp mai scurt.

- 20% of clients bring 80% of profits,
- 20% of drivers cause 80% of accidents,
- we wear 20% of clothes for 80% of the time
- 20% of employees generate 80% of products
- 20% of text allows to understand 80% of content.

Paradoxul Pesticidelor

Dacă vom scrie aceleași teste de mai multe ori, la un moment dat este posibil să nu mai găsim buguri noi.

Pentru a rezolva problema asta, trebuie să ne gândim în permanență dacă testele pe care le avem sunt suficient de acoperitoare sau mai trebuie să adăugăm ceva.

Testarea Depinde de Context

- În funcție de ce vrem să testăm – aplicație de mobile pentru comandă de mâncare sau software pentru gestionarea comunicării cu avioanele – vom alege cea mai potrivită tehnică de testare astfel încât să maximizăm eficiența testării.

- Deși procesul în sine e mereu același, aplicarea lui poate să fie diferită în funcție de ce vrem să obținem (securitatea cibernetică, performanță, funcționalitate, fiabilitate etc).

Absenta Erorilor Este o Misconceptie

Dacă găsim și reparăm buguri/erori nu este de folos dacă într-un final sistemul pe care îl creăm nu răspunde nevoilor utilizatorilor.

Concepte Importante

Mediu de test

Un loc unde echipa de testare poate efectua teste, modifica date în mod liber pentru testare și care stă la baza etapei de test execution.

API - Application Programming Interface

Un API este o interfață de comunicare între un client și un server menit să simplifice crearea unui produs de tip client-server.

Mediu de producție (Operațional)

Un mediu pe care rulează produsul finit și care aduce profit clientului/ utilizatorului final.

Testware

Orice instrumente care pot fi folosite în derularea procesului de testare cum ar fi documentație, scripturi, inputuri, proceduri, rezultate așteptate, fișiere, baze de date, mediu de testare etc.

{iT}Factory

Concepte Importante

Prioritate
Cât de repede avem nevoie de fixarea unui defect
Ex: Atunci când un user nu se poate loga într-un sistem, dacă este singurul user care folosește acel sistem, atunci prioritatea e înaltă. Dar dacă sunt mai mulți useri care folosesc sistemul și care au acces atunci prioritatea e mică spre medie.

Severitate
Impactul pe care îl are un defect asupra funcționării sistemului. Ex: Dacă un defect cauzează un sistem să fie complet nefuncțional, atunci severitatea este critică. Dar dacă e vorba doar despre typos sau greșeli minore, atunci severitatea este mică.

Mascarea Erorilor
Un defect poate ascunde un alt defect, făcându-l invizibil.

Stress Testing
Verificarea comportamentului sistemului atunci cand este încărcat până în apropierea punctului maxim de rezistență.

Load Testing
Verificarea comportamentului la un anumit nivel de încărcare de useri (ex: un eveniment duce la un număr mare de clienți care intră pe site).

Volume Testing
Verificarea comportamentului la un anumit nivel de încărcare de date (ex: un vânzător care are multe produse pe un site de e-commerce va încarca de fiecare dată toate produsele când își va deschide magazinul).

{iT}Factory

Nivelurile de Testare

Testare unitară - Un test unitar reprezintă testarea celei mai mici bucăți funcționale dintr-o aplicație cum ar fi funcții, clase, proceduri, interefețe.

Testarea unitară e o modalitate prin care fiecare bucată individuală de cod este testată pentru a verifica dacă este pregatită pentru utilizare.

Testarea de Componente - Se concentrează pe testarea unui singur modul dintr-o aplicație.

Obiectivele testării de componente:

➢Reducerea riscului;

➢Verificarea comportamentelor funcționale și non-funcționale în raport cu cerințele de business.

Testarea de Integrare - Se concentrează pe interacțiunile dintre componente și sisteme.

Obiectivele testării de integrare:

➢Reducerea riscului;

➢Verificarea comportamentelor funcționale și non-funcționale ale interfețelor și comunicării între sisteme/ componente în raport cu specificațiile.

➢Definirea încrederii în calitatea legăturilor dintre interfețe.

➢Găsirea defectelor în interfețe, între componente sau sisteme.

➢Prevenirea defectelor de la a ajunge în niveluri mai înalte ale testării.

Test Levels

Testarea de sistem - Se concentrează pe comportamentul și capabilitatea sistemului ca un tot unitar, ținând cont de comportamentul end-to-end al funcționalităților pe care sistemul trebuie sa le execute, ținând cont și de comportamentul non-funcțional așteptat al acelor taskuri.

Obiectivele testării de sistem:

- Reducerea riscului;
- Verificarea comportamentelor funcționale și non-funcționale în raport cu cerințele de business;
- Validarea faptului că sistemul este complet și funcționează corect;
- Definirea încrederii în calitatea sistemului complet;
- Găsirea defectelor;
- Prevenirea defectelor de la a ajunge la niveluri mai înalte ale testării.

Test Levels

Testare de acceptanță - Se concentrează pe comportamentul și capabilitățile sistemului ca un tot unitar.

Obiectivele testării de acceptanță:

- ➢ Definirea încrederii în calitatea sistemului ca un tot unitar;
- ➢ Validarea faptului că sistemul este complet și funcționează așa cum ne așteptam;
- ➢ Verificarea comportamentelor funcționale și non-funcționale în raport cu cerințele de business.

Alpha Testing

Ultima sesiune de testare înainte ca produsul să fie lansat publicului larg.
Reprezintă testarea unei aplicații atunci când dezvoltarea este completă sau aproape completă.
În urma testării alpha se pot face câteva schimbări minore dacă e necesar.

Testarea are loc la site-ul dezvoltatorului și se realizează în două faze:
- Faza întâi în care software-ul este testat de către dezvoltatori;
- Faza doi în care se face testarea de către echipa de QA într-un mediu similar cu cel al clientului.

Beta Testing

Are loc la site-ul clientului.
Se va trimite sistemul/ softul la utilizatori, aceștia vor instala aplicația și vor începe sa o folosească în condiții reale.
Scopul testării beta este să pună aplicația în mâinile unor utilizatori reali, oameni ce nu fac parte din echipa de dezvoltatori, pentru a descoperi defecte din perspectiva utilizatorului.

Psihologia testării
- Testarea Independentă-

Testerii iau parte la procesul de testare pentru a furniza o părere obiectivă asupra produsului, pe care dezvoltatorul nu o poate oferi. Testarea independentă poate fi făcută la oricare nivel al testării.

Există patru niveluri de independență:

- ❑ Teste create de către persoana care a scris codul (nivel foarte scăzut de independență);
- ❑ Teste create de către o alta persoana din aceeași echipa cu cea care a scris codul (nivel scăzut de independență);
- ❑ Teste create de către o altă persoană din aceeași companie (echipă independentă de testare);
- ❑ Teste create de o persoană din o organizație diferită (ex: companii de consultanță).

Psihologia testării
- Aspecte Importante -

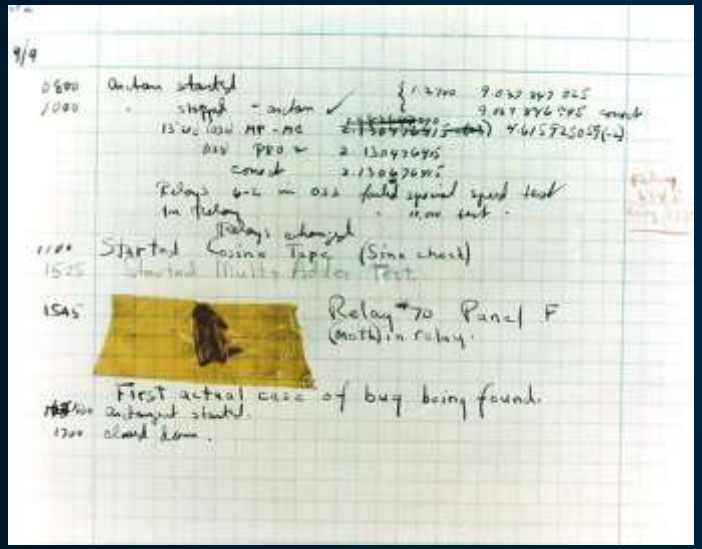
- Oamenii au tendința să își gestioneze planurile pe baza unor obiective definite de către management sau de către stakeholders – de aceea es important să definim obiective clare în testare.
- Găsirea erorilor în timpul testării poate fi văzută ca o modalitate de critică a produsului sau a autorului aplicației. De aceea e nevoie de pregatire prealabilă înainte de raportarea lor.
- Căutarea problemelor într-un sistem necesită: curiozitate, pesimism profesional, atenție la detalii, comunicare eficientă și cunoștințe despr produsul pe care îl testăm.

Psihologia Testării
- Raportarea constructivă a defectelor -

- Începem cu cooperare (obiectivul mutual este să obținem o calitate înaltă a produsului).
- Comunicăm într-un mod neutru.
- Testăm produsul, nu autorul produsului, deci nu criticăm autorul.
- Trebuie să înțelegem și perspectiva celuilalt.

Ce este un bug?

- Un bug este o neconcordanță între rezultatele așteptate și cele reale



Grace Murray Hopper

{iT}Factory

Elementele unui Bug Report

Titlu (Summary)
- O scurtă descriere care permite localizarea rapidă a problemei

Precondiții
- Ex: permisiune, starea necesară a sistemului pentru a putea reproduce eroarea (ex: trebuie să fie un client specific în sistem, clientul trebuie să aibă fonduri în bancă etc).

Pași de Reproducere
- Descriere pas cu pas a pașilor pe care trebuie să îi facem pentru a reproduce eroarea. Este recomandat să furnizăm și datele de testare.

Rezultate Așteptate
- Ce ar trebui să facă sistemul după completarea pașilor anteriori.

Rezultate Actuale
- Ce a făcut de fapt sistemul și de ce e incorect (putem face referință de exemplu către documentație)

Screenshot
- O poză în care se reflectă problema, care poate da o perspectivă adițională echipei de dezvoltare.

Prioritate
- Cât de repede aș avea nevoie de o soluție?

Severitate
- Cât de afectat este sistemul de acel bug?

Alte informații
- Orice alte informații care ar putea ajuta echipa de dezvoltare să găsească o soluție mai repede.

Poți să fii **un tester foarte bun** dacă ai cunoștințe de programare. Poți să fii de asemenea un tester foarte bun dacă nu ai cunoștințe de programare. Și poți să fii un tester groaznic cu sau fără cunoștinte de programare. **Un tester bun va învăța de ce skilluri are nevoie pentru a continua să fie grozav în propriul stil.**

{iT}Factory

Întrebări de interviu:

- Cum testăm un caiet?
- Ce este un bug?
- Care sunt nivelurile de Testare?

Întrebări & curiozități?