Testarea software este o componentă critică a dezvoltării software, având ca scop asigurarea unui software de înaltă calitate și fără erori. Automatizarea procesului de testare este esențială pentru succesul său datorită naturii laborioase și costisitoare a testării manuale. Aproximativ jumătate din resursele de dezvoltare software sunt dedicate testării, ceea ce subliniază importanța acesteia. Erorile pot fi introduse involuntar în timpul proiectării și dezvoltării software, ceea ce face ca testarea amănunțită să fie imperativă. Cu software-ul din ce în ce mai complex de astăzi, caracterizat printr-un volum mai mare de cod, testarea cuprinzătoare devine și mai crucială. Obiectivele principale ale testării includ asigurarea calității, verificarea, validarea și estimarea fiabilității. Testarea implică un compromis între buget, timp și calitate, cu scopul de a atinge un echilibru care să îndeplinească cerințele proiectului. Preocuparea centrală a ingineriei software este asigurarea calității codului computerizat. Testarea este abordarea cea mai utilizată pentru a atinge calitatea software-ului. Testarea poate fi clasificată mai departe în testare funcțională și testare de performanță. Testarea funcțională verifică dacă funcționalitățile oferite îndeplinesc cerințele de afaceri și funcționează conform intenției, în timp ce testarea de performanță analizează parametri precum timpul de răspuns, fiabilitatea și capacitatea de încărcare în comparație cu așteptările clienților. Este esențial să se analizeze critic dacă investiția în testare ar produce randamente acceptabile. În ciuda importanței testării, termenele limită ale proiectelor și bugetele adesea conduc echipele de proiect să sari peste testare sau să minimizeze eforturile de testare. Obiectivul principal al lucrării de cercetare este de a oferi o descriere concisă a ceea ce cercetarea își propune să atingă, rezumând realizările pe care cercetătorul dorește să le obțină prin proiect și oferind direcție pentru studiu.

**2. Strategii de Testare Software**

Testarea software implică executarea codului folosind diverse combinații de intrări și stări pentru a descoperi bug-uri. Testarea manuală și testarea automatizată sunt două abordări comune ale acestui proces. Testarea manuală implică testatori umani care execută manual cazuri de test pentru a identifica bug-uri, în timp ce testarea automatizată implică utilizarea uneltelor și scripturilor pentru a automatiza executarea cazurilor de test.

Cercetătorii lucrează activ pentru a crește automatizarea testării software pentru a reduce costurile asociate cu testarea manuală. Automatizând sarcinile repetitive și consumatoare de timp ale testării, organizațiile pot îmbunătăți eficiența și pot accelera procesul de testare. Testarea automatizată permite, de asemenea, o acoperire de test mai cuprinzătoare și poate ajuta la identificarea defectelor mai devreme în ciclul de dezvoltare, conducând în cele din urmă la produse software de înaltă calitate.

**2.1. Testarea Manuală**

Testarea manuală este o formă de testare software unde cazurile de test sunt executate manual de către un tester fără utilizarea unor unelte automatizate. Scopul testării manuale este de a identifica bug-uri, probleme și defecte în cadrul aplicației software. Este considerată metoda cea mai de bază și primitivă de testare, dar esențială pentru găsirea bug-urilor critice în aplicație. Înainte ca testarea automatizată să poată fi implementată, testarea manuală este necesară pentru a asigura funcționalitatea și stabilitatea aplicației.

Testarea manuală necesită mai mult efort comparativ cu testarea automatizată, dar este crucială pentru a face automatizarea fezabilă. Conceptele de testare manuală nu necesită cunoștințe despre vreun unelte de testare. Un principiu fundamental în testarea software este că "Automatizarea 100% nu este posibilă", subliniind importanța testării manuale. Această recunoaștere subliniază necesitatea testării manuale în procesul de dezvoltare software.

**2.2. Testarea Automatizată**

Testarea automatizată, cunoscută și ca automatizarea testelor, este o tehnică de testare software care implică utilizarea uneltelor specializate de testare automatizată pentru a executa o suită de teste. În contrast, testarea manuală este realizată de un tester uman care execută manual pașii de test pe un computer. Uneltele de testare automatizată pot introduce date de test în sistemul testat, pot compara rezultatele așteptate și cele reale și pot genera rapoarte detaliate de test.

Testarea automatizată necesită investiții semnificative de bani și resurse. Cu toate acestea, odată ce o suită de teste este automatizată, aceasta poate fi executată repetat fără necesitatea intervenției umane. Acest lucru îmbunătățește randamentul investiției (ROI) al automatizării testelor prin economisirea timpului și efortului în executarea cazurilor de test repetitive.

Scopul testării automatizate nu este de a elimina complet testarea manuală, ci de a reduce numărul de cazuri de test care trebuie rulate manual. Automatizând cazurile de test repetitive și consumatoare de timp, testatorii se pot concentra pe sarcini de testare mai complexe și exploratorii, conducând la produse software de calitate superioară.

**Unelte de Testare:**

**Quick Test Professional (QTP):**

Furnizor - Divizia de Software HP: Recunoscut ca soluția de top a industriei pentru testarea funcțională și automatizarea verificării regresiei. Utilizează o abordare bazată pe cuvinte cheie, oferind inginerului de testare automatizată acces complet la proprietățile obiectului de test prin intermediul unei interfețe integrate de scriptare și depanare.

Format XML Deschis pentru Rezultatele Testelor: Stochează rezultatele testelor într-un format XML deschis, facilitând personalizarea ușoară a rapoartelor conform cerințelor individuale și integrarea fără probleme a datelor rezultatelor testelor cu alte aplicații. Rezultatele testelor pot fi acum exportate în HTML.

Mediul IDE Nou: Introduce un mediu de dezvoltare a testelor foarte personalizabil.

Debugger Nou: Permite testerilor să identifice cu precizie erorile de test în timpul creării și întreținerii cazurilor de test.

Managementul Cuvintelor Cheie: Permite gestionarea cuvintelor cheie, inclusiv capacitatea de a activa sau dezactiva metode specifice din vedere cuvintelor cheie.

Suportul Mediului: Quick Test Professional suportă acum testarea funcțională într-o gamă largă de medii, inclusiv Windows, Web, .Net, Visual Basic, ActiveX, Java, SAP, Siebel, Oracle, PeopleSoft și emulatoare de terminal. În plus, există suport pentru medii noi.

Alegerea VBScript pentru scriptare poate să nu fie optimă din cauza limitărilor semnificative. VBScript lipsește suportul pentru module/biblioteci și nu poate fi considerat un limbaj "cu adevărat" orientat obiect deoarece nu suportă moștenirea claselor. Aceste limitări pot împiedica scalabilitatea, reutilizabilitatea și întreținerea scripturilor, afectând potențial eficiența și eficacitatea procesului de testare automatizată. Ca urmare, limbaje alternative de scriptare care oferă caracteristici și capacități mai robuste, cum ar fi Python sau JavaScript, pot fi opțiuni preferabile pentru proiectele de testare automatizată.

**RANOREX:**

a) Furnizor - Ranorex GmbH: RANOREX este recunoscut ca fiind uneltele de automatizare cu cea mai rapidă creștere, încorporând caracteristici de la uneltele de automatizare lider de pe piață și oferind avantaje față de acestea.

b) Depozitul de Obiecte: RANOREX dispune de un depozit de obiecte foarte definit care stochează și prezintă obiectele într-o manieră ierarhică bazată pe prezența lor pe ecran.

c) Codurile Utilizatorului: RANOREX permite utilizatorilor să creeze scenarii personalizate folosind codificare, adăugând la caracteristicile existente ale uneltei.

d) Compilatorul de Cod: Compilatorul de cod al uneltei identifică orice probleme legate de cod în scriptul de test în timpul compilării codului pentru execuție.

e) Vederea Cuvintelor Cheie Îmbunătățită: RANOREX oferă o vedere a cuvintelor cheie elaborată cu funcționalități cuprinzătoare și suplimente.

f) Raport Grafic: Rapoartele de test generate de RANOREX sunt îmbunătățite cu reprezentări grafice, facilitând o comparație mai rapidă și mai bună a defectelor în fiecare rulare de test.

g) Scriptarea Vederei Pas cu Pas: Interfața oferită este bazată pe non-cod, facilitând urmărirea pașilor lipsă în procesul de testare.

h) Suportul Mediului: RANOREX se mândrește cu capacități de procesare bazate pe imagini foarte dezvoltate, permițând suport pentru orice mediu bazat pe grafică.

i) Limbajul de Scriptare: RANOREX suportă limbaje avansate cum ar fi C# și VB.NET, oferindu-i un avantaj față de alte unelte de testare. În plus, permite testerilor să codeze în limbajul preferat pentru un confort și o flexibilitate sporite.

**3.3. LambdaTest**

LambdaTest este o platformă bazată pe cloud care facilitează testarea compatibilității cross-browser, atât manuală cât și automatizată. Utilizatorii pot efectua teste interactive live și teste de automatizare Selenium pe peste 2000 de browsere și sisteme de operare direct din browserul lor. Caracteristici principale:

Grid Selenium Scalabil: Sigur și fiabil pentru teste de automatizare.

Testare Cross-Browser: Suportă testarea pe site-uri web publice sau găzduite local și aplicații web.

Capturi de Ecran Automatizate: Permite testarea aspectului pe 36 de dispozitive diferite cu un singur clic.

Integrări: Oferă integrări cu un singur clic cu unelte populare de management de proiect și enterprise, cum ar fi Jira, Asana, Trello, GitHub, GitLab, BitBucket, Slack și Visual Studio Team Services.

Scop: Asigură că aplicațiile web funcționează eficient pe toate browserele, esențial pentru asigurarea calității.

**3.4. TestComplete**

Dezvoltat de SmartBear Software, TestComplete este un instrument de testare automatizată a interfeței utilizatorului pentru aplicații desktop, web și mobile. Aspectele sale deosebite includ:

Integrare AI: Simplifică crearea, întreținerea și executarea testelor funcționale.

Motor de Recunoaștere a Obiectelor Hibrid: Combină recunoașterea vizuală bazată pe proprietăți și AI.

Flexibilitate: Suportă operațiuni atât cu script, cât și fără script în șapte limbi de programare diferite.

Ușor de Utilizat: Recunoscut pentru interfața sa prietenoasă cu utilizatorul, crescând acoperirea testului și asigurând livrarea de software de înaltă calitate.

**3.5. Katalon Studio**

Katalon Studio este un instrument versatil pentru testarea automatizată a interfețelor web, API-urilor și aplicațiilor mobile. Se remarcă prin:

Bazat pe Selenium: Oferă un motor bazat pe Selenium pentru crearea și executarea scripturilor de test automatizate fără codificare.

Interfețe de Scriptare Duală: Se adresează atât utilizatorilor non-tehnici, cât și testerilor experimentați cu nevoi avansate de scriptare.

Cross-Platform: Suportă testarea pe Windows, macOS și Linux.

Capacități de Testare Cuprinzătoare: Permite testarea elementelor UI, inclusiv pop-up-uri, iFrames și gestionarea timpilor de așteptare.

Integrare și Raportare: Dispune de un set larg de integrări și oferă rapoarte de analiză detaliate.

**3.6. Selenium**

Selenium este un cadru de testare automatizată open-source pentru aplicații web, oferind:

Suport Multi-Limbă: Permite crearea de scripturi de test în limbi precum Java, C#, Python și altele.

Set de Unelte: Include Selenium IDE, RC, WebDriver și Grid, satisfăcând diverse cerințe de testare.

Flexibilitate și Compatibilitate: Suportă testarea cross-platform și cross-browser.

Suport API și Cadru: Oferă suport extensiv de bibliotecă și driver, facilitând integrarea cu diverse cadre de testare.

**3.7. API (Interfața de Programare a Aplicațiilor)**

În contextul Selenium, API-ul facilitează:

Portabilitatea Scripturilor: Permite integrarea scripturilor de testare în diferite limbaje de programare.

Suport Comprehensiv: Include legături client-side de bază și legături terțe pentru suport extins al limbajelor de programare.

Puncte Cheie ale Tabelului de Comparare

Suport Platformă și Aplicație: Majoritatea instrumentelor suportă testarea cross-platform a aplicațiilor web, mobile și serviciilor API/web, unele, cum ar fi TestComplete, acoperind și aplicațiile desktop Windows.

Limbaje de Scriptare și Abilități de Programare: O combinație de limbaje și cerințe de abilități, cu unele instrumente concepute pentru ușurința de utilizare fără nevoia de abilități avansate de programare.

Timp de Creare a Scriptului: Variază între instrumente, cu TestComplete și Katalon Studio remarcate pentru crearea rapidă a scripturilor, în contrast cu ritmul mai lent în Selenium.

**4. Unelte de Cercetare**

Uneltele de cercetare sunt definite ca mijloace care facilitează în general cercetarea și activitățile asociate. Aceste unelte științifice permit cercetătorilor să colecteze, organizeze, analizeze, vizualizeze și să publice rezultatele cercetării.

**4.1. Unelte pentru Testarea Performanței**

Scopul testării performanței este de a atinge viteza înaltă, scalabilitatea și stabilitatea unui sistem. Există mai multe tipuri de teste de performanță, fiecare simulând scenarii posibile diferite, cum ar fi:

Scalabilitatea: Testele de scalabilitate sunt cruciale pentru sisteme care doresc să susțină un număr mare de utilizatori.

Stabilitatea: Se dorește ca aplicația să funcționeze constant, chiar și sub sarcini grele.

Unelte Specifice Include:

**4.1.1. WebLOAD:** Preferat de întreprinderi pentru testări complexe și încărcări grele, WebLOAD facilitează testarea încărcării și stresului aplicațiilor web, generând sarcini din cloud sau mașini locale.

**4.1.2. LoadNinja:** Oferă posibilitatea de a crea teste de încărcare sofisticate fără scripturi, reducând timpul de testare și utilizând browsere reale pentru simulare.

**4.1.3. HP Load Runner:** Standard industrial pentru testarea încărcării aplicațiilor, folosind utilizatori virtuali pentru a simula sarcini reale.

**4.1.4. Apache J Meter:** Instrument open-source pentru testarea comportamentului funcțional și a performanței, cu suport pentru testare distribuită.

**4.1.5. Selenium:** Folosit pentru testarea aplicațiilor web, include patru instrumente care suportă diferite browsere web.

**4.1.6. NeoLoad:** Măsoară performanța aplicațiilor web și oferă soluții pentru optimizarea site-urilor web.

**4.1.7. WebLoad Professional:** Specializat pentru testarea performanței aplicațiilor internet, cu suport pentru tehnologii Ajax, JSON și servicii web SOAP/XML.

4**.1.8. LoadUI:** Instrument open-source pentru testarea performanței, eficient în evaluarea aplicațiilor web.

**4.1.9. WAPT:** Se concentrează pe testarea performanței, încărcării și stresului aplicațiilor web, oferind informații despre sarcinile utilizatorilor virtuali.

**4.1.10. Loadster:** Instrument pentru testarea încărcării aplicațiilor web, manevrează cookies, sesiuni de utilizator și date dinamice de formular.

**4.1.11. LoadImpact:** Simulează traficul în mod realist în medii de testare, generând automat scripturi și măsurând utilizarea resurselor sistemului.

**5. Identificarea Cerințelor de Utilizabilitate ale Uneltelor de Testare Software**

Conform lui Andreas Hollinger, utilizabilitatea se definește ca ușurința de utilizare și adecvarea unui sistem pentru o anumită clasă de utilizatori care efectuează activități specifice într-un mediu precis. S-au specificat criterii precum ușurința de utilizare, de învățare, simplitatea, eficacitatea, informația și interfața utilizatorului ca fiind cerințe de utilizabilitate pentru evaluarea uneltelor de testare software selectate.

**5.1. Selecția Participanților**

Pentru studiile de utilizabilitate, au fost selectați 20 de participanți, studenți ai Institutului Tehnologic Blekinge, care studiază în domeniul informaticii și ingineriei software. Participanții au fost împărțiți în două grupuri pentru a conduce teste de utilizabilitate folosind LoadRunner și respectiv JMeter. Scenariile bazate pe sarcini selectate pentru testul de utilizabilitate acoperă toate funcționalitățile instrumentelor selectate, fiind concepute pentru a atinge obiectivele cercetării.

**5.2. Sarcini Bazate pe Scenarii**

Sarcinile selectate pentru testul de utilizabilitate sunt bazate pe scenarii și scrise în limba engleză. Aceste sarcini acoperă toate funcționalitățile parametrilor selectați ai instrumentelor în cauză. Autorul a proiectat scenariile în așa fel încât să atingă obiectivul acestei cercetări, evidențiind dificultatea de a testa toate funcționalitățile instrumentelor alese.

**5.3. Locația Testului**

Testul de utilizabilitate a fost realizat în sălile de grup ale Institutului Tehnologic Blekinge. Participanților li s-au furnizat sarcinile pe hârtie și au fost lăsați singuri pentru a efectua testul fără întreruperi, asigurând astfel condiții optime pentru concentrarea pe sarcini.

**5.4. Echipamente și Materiale pentru Testul de Utilizabilitate**

Pentru a avea un experiment controlat, fiecărui participant i s-a furnizat un computer identic. După finalizarea testului, s-a verificat ca instrumentul de testare software să fie complet dezinstalat. Toate sarcinile și chestionarele au fost scrise pe hârtie și furnizate individual fiecărui participant.

Tabelul 4. Comparație între Uneltele LoadRunner și JMeter

Costul de Achiziție: LoadRunner este un instrument licențiat cu costuri de întreținere ridicate, în timp ce JMeter este un instrument open-source și gratuit.

Timpul de Instalare: Instalarea LoadRunner este consumatoare de timp și utilizează mai mult spațiu pe disc, pe când JMeter se instalează rapid și necesită mai puțin spațiu pe disc.

Generarea de Sarcină: LoadRunner restricționează numărul de utilizatori chiar și ca instrument licențiat, în timp ce JMeter nu impune restricții asupra numărului de utilizatori, permițând oricâte, în funcție de suportul hardware.

Scripturi: LoadRunner ascunde scripturile în nivele pentru a face scriptul să pară mai organizat, în timp ce JMeter afișează întregul script, descriind fluxul HTTP al scenariului.

Rezultate: LoadRunner oferă o analiză excelentă și generare dinamică de grafice, împreună cu metrici de partea serverului, spre deosebire de JMeter, care generează grafice limitate fără metrici de partea serverului.

Timpul de Răspuns la Tranzacții: Este mai mare pentru LoadRunner, deoarece include timpul de descărcare a imaginilor, spre deosebire de JMeter care exclude acest timp.

Analiza și Monitorizarea: Caracteristicile oferite de LoadRunner sunt lider pe piață, în timp ce JMeter nu este foarte puternic în această categorie.

Funcția de Înregistrare și Redare: LoadRunner oferă o funcție de înregistrare și redare fără egal, comparativ cu orice sursă deschisă, pe când la JMeter această caracteristică nu este foarte puternică.

Utilizare Preferată: LoadRunner este preferat atunci când se lucrează cu diferite tipuri de aplicații IT, care se ocupă de diferite protocoale, pe când JMeter este preferat pentru lucrul extensiv pe protocoale web.

Rezoluții: Soluțiile sunt implementate mai rapid cu LoadRunner, în timp ce cu JMeter, testatorii trebuie să găsească modalități de a rezolva problemele, procesul fiind mai lent.