Titlul lucrării

PROIECT DE DIPLOMĂ

Conducător științific: **Prof.dr.ing. Honoriu VĂLEAN**

|  |  |
| --- | --- |
| DECAN  **Prof.dr.ing. Liviu MICLEA** | Vizat,  DIRECTOR DEPARTAMENT AUTOMATICĂ  **Prof.dr.ing. Honoriu VĂLEAN** |

Autor: **Ruxandra Mara IERIMA**

Cuprins

[1. INTRODUCERE 2](#_Toc10300436)

[2.1 Context general 2](#_Toc10300437)

[2.2 Obiective 2](#_Toc10300438)

[2.3 Specificații 2](#_Toc10300439)

[2. STADIUL ACTUAL 3](#_Toc10300440)

[2.1 Evoluția conceptului de testare 3](#_Toc10300441)

[3. FUNDAMENTARE TEORETICĂ 6](#_Toc10300442)

[4.1 Aspecte generale despre testare 6](#_Toc10300443)

[3.11 Definirea și scopurile testării 6](#_Toc10300444)

[3.12 Nivele de testare 6](#_Toc10300445)

[3.13 Testarea automată (tipuri de teste automate exemple) 7](#_Toc10300446)

[3.14 Certificări( ISTQB) 7](#_Toc10300447)

[4.2 Selenium WebDriver 7](#_Toc10300448)

[3.21 Introducere (cand a aparut, ce face, definitii) 7](#_Toc10300449)

[3.22 Comenzi (descriere operatii si comenzi, implementarea pe un proiect-page object) 7](#_Toc10300450)

[4.3 Gherkin 7](#_Toc10300451)

[3.31 Introducere (benefici utilizare, fondare etc) 7](#_Toc10300452)

[3.32 Limbajul Gherkin (descriere cuvinte cheie, sintaxa, exemple) 7](#_Toc10300453)

[4. IMPLEMENTAREA SOLUȚIEI 7](#_Toc10300454)

[4.1 Descrierea aplicației (framework-ului) 7](#_Toc10300455)

[4.2 Configurarea programelor (setup imagini tooluri folosite) 7](#_Toc10300456)

[4.3 Implementarea soluției (cod) 7](#_Toc10300457)

[5. REZULTATE EXPERIMENTALE 7](#_Toc10300458)

[5.1 Rularea testelor (se ruleaza toate testele deodata) 7](#_Toc10300459)

[5.2 Generarea raportului de validare 7](#_Toc10300460)

[6. CONCLUZII 7](#_Toc10300461)

[7. BIBLIOGRAFIE 7](#_Toc10300462)

[8. ANEXE 8](#_Toc10300463)

# INTRODUCERE

## Context general

In introducere familiarizați cititorul cu motivația lucrării, plasați lucrarea într-un context care să permită cititorului să înțeleagă obiectivele.

Descrieți importanța lucrării, de ce merita să o faceți, plasati ideile într-un context larg.

Sustineți studiul: de ce exact aceasta aplicație/implementare. Comentați asupra aspectelor teoretice sau practice care v-au facut să o alegeți.

In funcție de natura lucrării, ar putea fi necesar să prezentați informații de fond asupra domeniului în care se încadrează aplicația, mai ales dacă aveti o lucrare într-un domeniu multidisciplinar. In acest caz, puteți introduce terminologia pe care o utilizați în continuare.

Descrieti pe scurt lucrarea: ce conține fiecare capitol.

## Obiective

Enumerati și explicati obiectivele lucrării: ce v-ați propus să realizați în contextul prezentat anterior.

Obiectivele pot fi prezentate sub formă de listă care să evidențieze precis orientarea lucrării, să identifice conceptele fundamentale pe care le studiati, să stabilească scopul aplicației pe care o realizați, sau enuntați întrebarile la care intentionați să răspundeți în lucrare.

## Specificații

In specificațiile lucrării detaliați cerințele. Descrieți ce intentionați să obtineți. Vă puteți referi la funcțiile aplicației, interfață, nivele de performanță, structuri de date, elemente, securitate, fiabilitate, calitate, limitări, etc.

# STADIUL ACTUAL

Testarea software are ca scop oferirea de informații cu privire la calitatea unui produs sau a unui serviciu. În investigarea produsului supus testării se va lua în considerare contextul în care acesta va fi utilizat. Pentru a putea înțelege și estima riscurile pe care le reprezintă implementarea produsului în dezvoltare, testarea software oferă o perspectivă obiectivă.

Această testare reprezintă procesul de identificare a defectelor și a erorilor unui produs prin observarea diferențelor dintre un comportament dorit și un comportament observat cu ajutorul diferitelor tehnici de testare. Un alt aspect al testării software reprezintă validarea produsului finit prin verificarea faptului că acesta corespunde așteptărilor.

Testarea este foarte importantă deoarece chiar și unele erori minore dintr-o aplicație pot avea efecte negative precum costuri suplimentare sau desconsiderarea calității acelui produs de către client.

Acest concept nu este unul nou apărut însă dezvoltarea și identificarea de noi tehnologii este in continuă creștere datorită standardelor și așteptările ridicate ale clienților.

## Evoluția conceptului de testare

Clasificarea fazelor și obiectivelor testării realizată de către Gelperin și W.C. Hetzel (1):

**Orientarea spre depanare (până în 1956)**

Această perioadă a fost orientată mai mult spre partea hardware, unde testarea a fost adesea asociată cu depanarea: neexistând o diferență clară între teste și depanare. Pe partea de software, defectele nu erau considerate atât de importante, în general persoanele care se ocupau cu implementarea codului făceau si partea de testare.

Prima referire la termenul de “bug” în domeniul informatic când pe un circuit electronic defectuos s-a descoperit o molie.

**Orientarea spre demonstrație (1957-1918)**

Diferența dintre depanare și testare începea să se distingă în această perioadă. Accentul pe testare și pe remedierea defectelor a crescut, persoanele implicate in dezvoltarea de programe informatice fiind mult mai conștiente de această problemă. Scopul sistemelor software era acela de a îndeplini cerințele.

**Orientarea spre defectare (1979-1982)**

În această perioadă se propune, pe lângă procesul de testare, o serie de activități de analiză și control cu scopul creșterii calității produsului. Această abordare vine ca o rezolvare a testării bazate pe demonstrație unde setul de date introdus pentru validarea procesului putea fi ales involuntar in favoarea produsului.

**Orientarea spre evaluare (1983-1987)**

Principalul scop a fost evaluarea produsului în timpul ciclului de viață al acestuia, cât și alegerea diferitelor metode de testare și validare în funcție de caracteristicele sistemului in ideea creșterii calității. În această perioadă au apărut și primele locuri de muncă dedicate de tester.

**Orientarea spre prevenire (1988-prezent)**

Pentru a reduce costurile de dezvoltare și de întreținere a unui sistem software, testarea a fost introdusă încă din primele faze, în paralel cu implementarea produsului. Încep să fie definite primele activități principale ale testării precum: planificarea, analiza, proiectarea, implementarea, execuția și întreținerea. Prin respectarea acestor metode, defectele apărute în urma implementării aplicației scad considerabil.

**Metodologii utilizate in procesul de testare:**

**Definire metodologii……**

* **Testarea Waterfall –** reprezintă un proces secvențial cu diferite faze prezentate in Figura 2.1, unde faza următoare începe doar în momentul în care toate etapele din faza anterioară au fost finalizate.

1. Prima fază a modelului este faza de „Cerințe” în care toate cerințele proiectului sunt definite în întregime înainte de a începe testarea. În această fază, echipa de testare definește domeniul de aplicare al testării, strategia de testare și elaborează un plan detaliat de testare.
2. În faza de „Proiectare” echipa stabilește detaliile tehnice precum limbajele de programare care urmează să fie folosite.
3. A treia fază este cea de „Implementare” unde se creează testele și se implementează

funcționalitățile care ajută la dezvoltarea acestora.

1. Faza de „Verificare” reprezintă procesul de rulare al testelor, și comunicarea rezultatelor.
2. Ultima fază, cea de „Mentenanță” este etapa de întreținere a produsului software, care poate necesita schimbări ulterioare, în funcție de cerințele clientului.

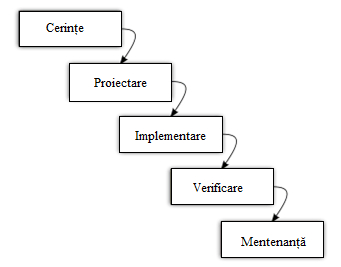
****

Figura 2.1. Etapele testării Waterfall

* **Testarea Agile -** este un proces de testare care urmează principiile Agile ale dezvoltării software, unde cerințele sunt specificate treptat de către client. Procesul este unu continuu în care exista o integrare între implementare și testare, scopul comun fiind obținerea unei calități ridicate.

Testarea Agile este nestructurată în comparație cu abordarea în cascadă și există o planificare minimă. Fiindcă procesul de testare se realizează chiar de la începutul proiectului, erorile pot fi rezolvate mult mai rapid. Avantajele acestor metodologii sunt flexibilitatea, adaptabilitatea la schimbări și o mai bună determinare a problemelor prin întâlniri zilnice între echipa de testare și cea de dezvoltare.

Figura 2.2. Etapele testării Agile

* **Metodologia Scrum –** activitățile de testare din cadrul acestui proces implică estimarea efortului de către tester cu privire la timpul necesar pe care îl dedica pentru testarea unei anumite secțiuni din aplicație. O altă activitate importantă din acest proces este aceea de a prioritiza defectele după gradul de severitate pe care acestea le prezintă.

# FUNDAMENTARE TEORETICĂ

## Aspecte generale despre testare

### Definirea și scopurile testării

Unele dintre definițiile și citatele des întâlnite care descriu conceptul de testarea sunt:

“Testarea este procesul prin care se execută un program cu intenţia de a găsi erori” (Myers)

„Procesul de exercitare sau de evaluare a unui sistem manual sau automatizat înseamnă să se verifice dacă îndeplinește condițiile specificate sau pentru a identifica diferențele dintre rezultatele așteptate și cele reale.” (2)

„Procesul constă în toate activitățile din ciclul de viață al unui produs, atât din punct de vedere static, cât și dinamic, referinduse la planificarea, pregătirea și evaluarea produselor software și a produselor de lucru aferente pentru a determina dacă îndeplinesc cerințele specificate, pentru a demonstra că sunt adecvate scopului și pentru a detecta defectele.” (3)

O concluzie pe care o putem trage în urma acestor definiții enumerate mai sus este aceea că scopul testării constă în asigurarea calității unui produs, la standardele cerute de către client. De aceea, când o organizație investește într-un produs software, acesta are obligația de a evalua calitatea produsului din perspectiva utilizatorilor. Testarea Software este procesul care vine să facă această evaluare posibilă într-un mod cât mai obiectiv.

Unul dintre marile avantaje aduse de testare este diminuarea pierderilor.

NIST(Institutul Național de Standarde și Tehnologie) a realizat un studiu în anul 2002 care arată că defectele și erorile apărute în produsele software au cauzat pierderi anuale estimate la 59.5 miliarde de dolari în economia S.U.A.

În funcție de caracteristicile fiecărui proiect în parte, testarea are diferite obiective (4):

* Să evalueze cerințele, proiectarea si codul.
* Să verifice dacă au fost îndeplinite toate cerințele specificate.
* Să valideze dacă produsul testat cuprinde toate caracteristicile și se ridică la așteptările utilizatorilor sau a altor părți interesate.
* Mărirea încrederii în nivelul de calitate a produsului.
* Prevenirea defectelor.
* Găsirea defectelor.
* Reducerea nivelului de risc pentru calitatea necorespunzătoare a software-ului.

### Nivele de testare

Fiecare nivel de testare reprezintă o instanță a procesului de testare, cuprinzând factori care influențează acest proces din cadrul unei organizații precum: bugetul și resursele, complexitatea, standarde interne și externe cât și cerințele contractuale.

**Testarea componentelor** **–** este știută și sub numele de testarea unității(Unit Testing), unde sunt testate unitățile/componentele individuale ale software-ului. O componentă este definită ca cea mai mică parte testabilă din aplicație. Aceasta poate reprezenta structuri de cod si date, clase, module de baze de date sau chiar componente mai ample, ele depinzând de complexitatea aplicației. Acest nivel de testare este primul efectual asupra codului, imediat după ce acesta a fost scris. De aceea, testarea componentelor este efectuală de obicei de către developeri, necesitând un grad de cunoștințe de programare mai ridicat.

Scopul este de a valida faptul că fiecare unitate a software-ului funcționează așa cum a fost proiectată.

**Testarea integrării** –

**Testarea sistemului** –

**Testarea de acceptare** –

### Testarea automată (tipuri de teste automate exemple)

### Certificări( ISTQB)

## Selenium WebDriver

### Introducere (cand a aparut, ce face, definitii)

### Comenzi (descriere operatii si comenzi, implementarea pe un proiect-page object)

## Gherkin

### Introducere (benefici utilizare, fondare etc)

### Limbajul Gherkin (descriere cuvinte cheie, sintaxa, exemple)

# IMPLEMENTAREA SOLUȚIEI

## Descrierea aplicației (framework-ului)

## Configurarea programelor (setup imagini tooluri folosite)

## Implementarea soluției (cod,diagrama claselor, secventiala etc)

# REZULTATE EXPERIMENTALE

## Rularea testelor (se ruleaza toate testele deodata)

## Generarea raportului de validare

# CONCLUZII

# BIBLIOGRAFIE

1. **Gelperin, D.** *„The Growth of Software Testing”.* 1988.

2. *IEEE Standard Glossary.* 1983.

3. **ISTQB®-Certified.** *Tester,Basics of Software Testing, Foundation Level Curriculum.* 2007.

4. *ISTQB®- Foundation Level.* 2018.

# ANEXE