



PROJECT FINAL

TIMOFTE ALEXANDRU BOGDAN

DATA EXAMEN 10.04.2025

Există două discipline diferite implicate în testarea software: **testarea manuală și testarea automată**. În ciuda faptului că ambele au efectiv aceeași funcție, ele sunt discipline distincte pe care companiile le folosesc pentru a-și examina pachetele software.

- Procesul de testare software este structurat pe **patru niveluri de testare**, fiecare având un scop distinct. Acestea sunt:

1. Testarea unității (Unit Testing)

- Se testează componentele individuale ale codului (funcții, metode, module).
- Scopul este de a verifica dacă fiecare unitate funcționează corect.
- De obicei, este automatizată și realizată de dezvoltatori.

2. Testarea integrării (Integration Testing)

- Verifică interacțiunea dintre module sau componente individuale.
- Scopul este de a detecta erori în comunicarea dintre diferite părți ale aplicației.
- Poate include testarea API-urilor, bazelor de date și serviciilor externe.

3. Testarea sistemului (System Testing)

- Se testează aplicația ca un întreg, verificând dacă respectă cerințele specificate.
- Include testarea funcționalității, performanței, securității și compatibilității.
- Se realizează într-un mediu cât mai apropiat de producție.

4. Testarea de acceptanță (Acceptance Testing)

- Validează dacă sistemul îndeplinește cerințele utilizatorului final.
- Poate fi efectuată de echipa QA sau direct de către client.
- Exemple: User Acceptance Testing (UAT) și Beta Testing.

- **Evaluarea cerințelor de business** are loc în **etapa de revizuire a cerințelor** sau **testarea statică**, care este o activitate inițială în cadrul procesului de testare software. Aceasta face parte din **Testarea de Validare** și se realizează înainte de implementarea efectivă a codului.

- In Situatia de fata, pentru aplicatia “**HapifyMe**” trebuie să luăm în considerare mai multe aspecte esențiale, inclusiv funcționalitățile principale, audiența țintă, obiectivele de afaceri și modelul de monetizare. In detaliu ar fi cerintele urmatoare:

1. Obiectivele aplicației

- Crearea unei platforme sociale care permite utilizatorilor să se conecteze, să partajeze conținut și să interacționeze.
- Creșterea bazei de utilizatori și maximizarea retenției acestora.
- Generarea de venituri prin publicitate, abonamente premium sau alte metode.
- Oferirea unei experiențe sigure și intuitive pentru utilizatori.

2. Funcționalități principale

a. Autentificare și gestionare conturi

- Creare cont prin email, telefon sau conturi externe (Google, Apple, Facebook).
- Logare securizată.
- Recuperare parolă și resetare cont.
- Personalizare profil (poză, biografie, preferințe, setări de confidențialitate).

b. News Feed & Postări

- Creare și partajare postări (text, imagini, videoclipuri, linkuri).
- Algoritm de recomandare bazat pe preferințe și interacțiuni.
- Reacții la postări (like, love, etc.).
- Comentarii și răspunsuri la comentarii.
- Distribuie postări pe profil sau în grupuri.

c. Prietenii și Conectare între Utilizatori

- Cereri de prietenie și sugestii de prieteni.
- Urmărirea altor utilizatori (pentru conținut public).
- Blocare utilizatori și raportare conținut.

d. Mesagerie și Comunicare

- Chat în timp real (mesaje text, GIF-uri, emoji).
- Mesaje vocale și apeluri audio/video.
- Grupuri de chat.
- Notificări pentru mesaje noi.

e. Grupuri și Comunități

- Creare și gestionare grupuri tematice.
- Moderare de conținut și membri.
- Postări exclusive pentru grupuri.

f. Paginile și Business-uri

- Creare de pagini pentru companii, branduri și personalități publice.
- Instrumente de promovare și publicitate.
- Rapoarte despre engagement și audiență.

g. Story-uri și conținut efemer

- Postări temporare care dispar după 24h.
- Suport pentru imagini, video și text.
- Filtre și efecte speciale.

h. Notificări și Interacțiuni

- Notificări push pentru mesaje, reacții și solicitări de prietenie.
- Posibilitatea de a personaliza tipurile de notificări primite.

i. Setări de confidențialitate și securitate

- Gestionarea vizibilității postărilor și profilului.
- Controlul asupra datelor personale și a modului în care sunt utilizate.
- Opțiuni GDPR pentru descărcarea și ștergerea contului.

3. Cerințe Tehnice și Non-Funcționale

- **Scalabilitate:** Capacitate de a susține milioane de utilizatori activi.
- **Securitate:** Protecție împotriva atacurilor cibernetice și criptarea datelor.
- **Performanță:** Timp de răspuns rapid și încărcare optimizată a conținutului.
- **Design Responsiv:** Suport pentru desktop, mobil și tabletă.
- **Accesibilitate:** Adaptabilitate pentru utilizatori cu dizabilități.

4. Model de Monetizare

- Publicitate targetată (anunțuri sponsorizate, reclame video).
- Abonamente premium pentru funcționalități exclusive.
- Marketplace cu taxe de tranzacție.
- Funcții speciale pentru pagini business (promovare, analize avansate).

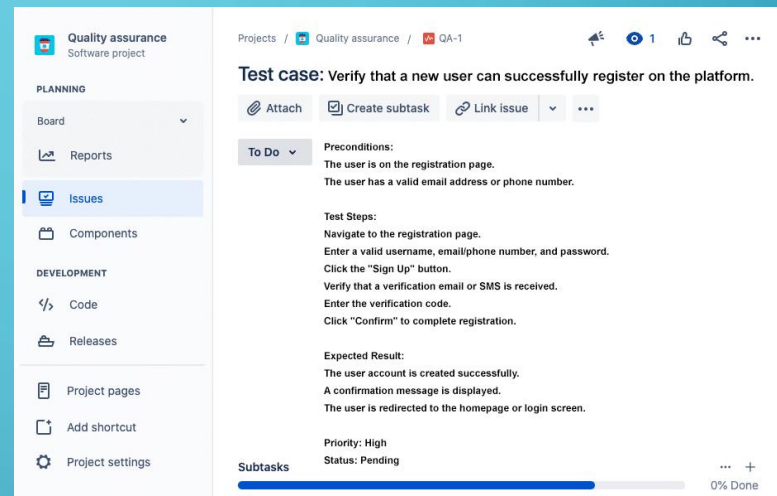
5. KPI-uri pentru succes

- Numărul de utilizatori activi zilnic (DAU) și lunar (MAU).
- Timpul petrecut pe platformă de utilizatori.
- Creșterea bazei de utilizatori.
- Venitul generat din reclame și abonamente.

Acestea ar fi cerințele esențiale pentru aplicați **“HapifyMe”**.

- Un **test case** poate avea mai multe statusuri în funcție de rezultatul rulării și de stadiul său în procesul de testare. Cele mai comune statusuri sunt:

1. **Not Executed (Neexecutat)**
2. **Passed (Trecut/Reușit)**
3. **Failed (Eșuat)**
4. **Blocked (Blocat)**
5. **Skipped (Sărit/Peste)**
6. **In Progress (În Curs de Execuție)**
7. **Retest (Retestat)**
8. **Deferred (Amânat)**



- **Criteriile de intrare și criteriile de ieșire** sunt condiții utilizate în procesul de testare software pentru a determina când poate începe sau când se poate considera finalizată o activitate de testare.

- **Criteriile de intrare** definesc **ce trebuie să fie pregătit** pentru a începe testarea.
- **Criteriile de ieșire** definesc **ce trebuie să fie îndeplinit** pentru a încheia testarea.

- **Precondition (Precondiție)**

O precondiție este o condiție care trebuie să fie îndeplinită înainte de executarea unui test. Practic, reprezintă starea inițială necesară pentru ca testul să fie valid.

- **Exemple de precondiții:**

- Utilizatorul trebuie să fie autentificat înainte de a testa funcționalitatea de schimbare a parolei.
- Baza de date trebuie să conțină un produs înainte de a testa procesul de adăugare în coș.
- Serverul de test trebuie să fie activ înainte de a începe testele de performanță.

- **Test Condition (Condiție de testare)**

O condiție de testare este un aspect specific al sistemului care trebuie verificat în cadrul testării. Acesta este un element derivat din cerințele software și definește **ce trebuie testat**.

- **Exemple de condiții de testare:**

- Verificarea dacă un utilizator poate reseta parola.
- Asigurarea că utilizatorul primește un mesaj de eroare dacă introduce date incorecte.
- Testarea funcționalității de adăugare a unui produs în coșul de cumpărături.

- **Condiția de testare** este ceea ce urmează să fie verificat în cadrul testului.
- În procesul de testare se vor folosi diferite tool-uri pentru a ne face munca mai ușoară și de a simplifica trecerea de informații de la un departament la altul. Un instrument esențial pentru Managementul Proiectelor este **JIRA**.
- **Jira** este o platformă de management al proiectelor dezvoltată de Atlassian, utilizată în principal pentru urmărirea erorilor și gestionarea activităților în cadrul echipelor de dezvoltare software. Datorită flexibilității și adaptabilității sale, Jira este folosită și în alte domenii, precum IT, marketing și gestionarea proceselor de afaceri.
- Un **release** reprezintă versiunea finală a unui produs software sau a unei actualizări, care este livrată utilizatorilor finali. Acesta conține noi funcționalități, îmbunătățiri sau remedieri de bug-uri și poate fi distribuit intern (echipe de testare, clienți beta) sau public.
- **Cycle Summary** este un raport care detaliază rezultatele unui ciclu de testare și ajută la luarea deciziei de release.
- Un **release** este despre **distribuirea produsului**, iar un *cycle summary* este despre **evaluarea testării** înainte de release.

- **Utilitatea Matricei de Trasabilitate:**

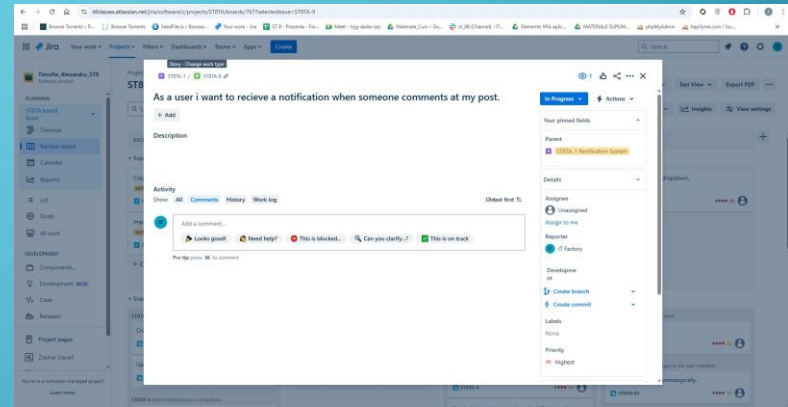
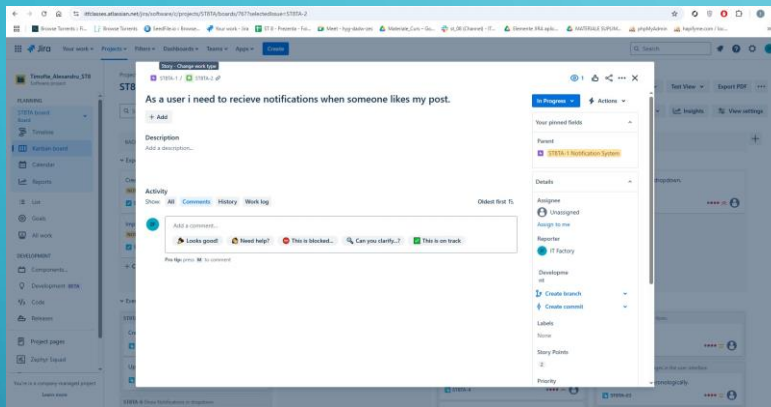
1. Asigură acoperirea completă a cerințelor
2. Ușurează gestionarea schimbărilor
3. Transparență pentru părțile interesate
4. Verificarea conformității cu cerințele de business
5. Reducerea riscurilor

- **Matricea de trasabilitate** este un document sau un instrument utilizat în managementul testării software pentru a urmări și a demonstra legătura dintre cerințele de business și testele efectuate. Ea asigură că toate cerințele au fost testate corespunzător și ajută echipele de testare să verifice că fiecare cerință este acoperită printr-un caz de testare adecvat.

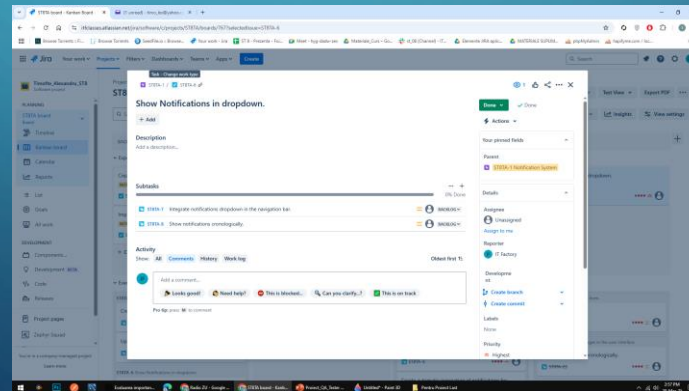
- **Cei doi parametri principali după care se evaluează importanța unui bug sunt:**

1. **Severitatea** reflectă **gravitatea defectului** din punct de vedere tehnic.
2. **Prioritatea** reflectă **urgenta și importanța** rezolvării defectului din perspectiva utilizatorului și business-ului.

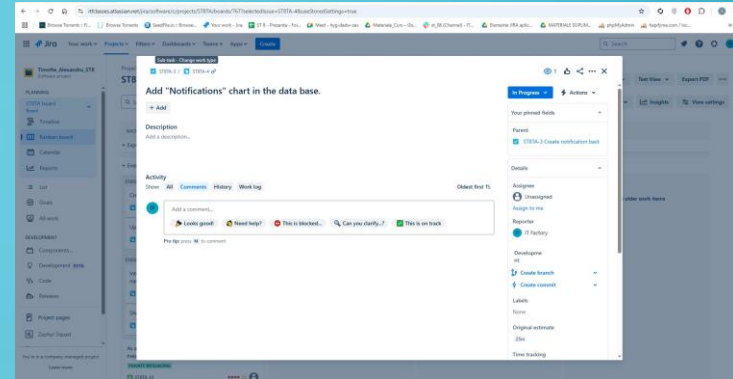
2.Story – O cerință funcțională sau o caracteristică care trebuie implementată. Este o unitate de lucru mai mică dintr-un Epic. In aceasta situatie , din Epic-ul “Notification System” se creeaza doua subnivele pentru a rezolva problema mult mai usor.



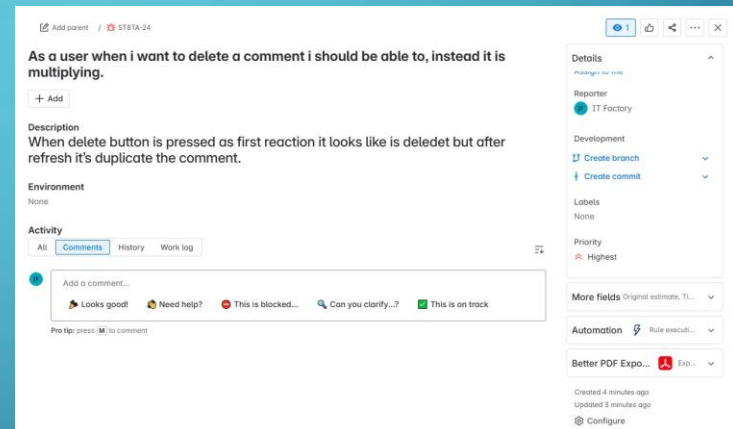
3.Task – O sarcină independentă care trebuie realizată. Poate face parte dintr-un Epic sau poate fi un element de sine stătător.



4.Sub-task – O subdiviziune a unui Task sau Story, utilizată pentru a împărți lucrul în părți mai mici.



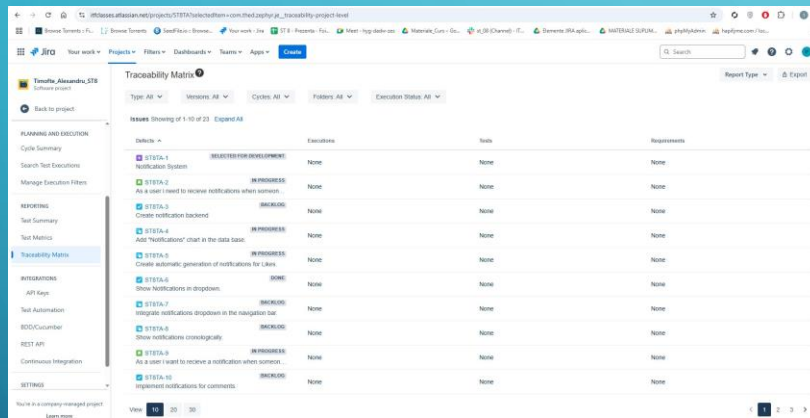
5.Bug – O eroare sau problemă care trebuie remediată. Poate fi asociat cu un Story, Task sau Epic.



6.Spike – Un task special utilizat pentru cercetare sau investigație, folosit adesea pentru a clarifica cerințele înainte de dezvoltare.
Această ierarhie ajută echipele să organizeze eficient proiectele și să urmărească progresul fiecărei componente.

Tipuri de Traceability Matrix:

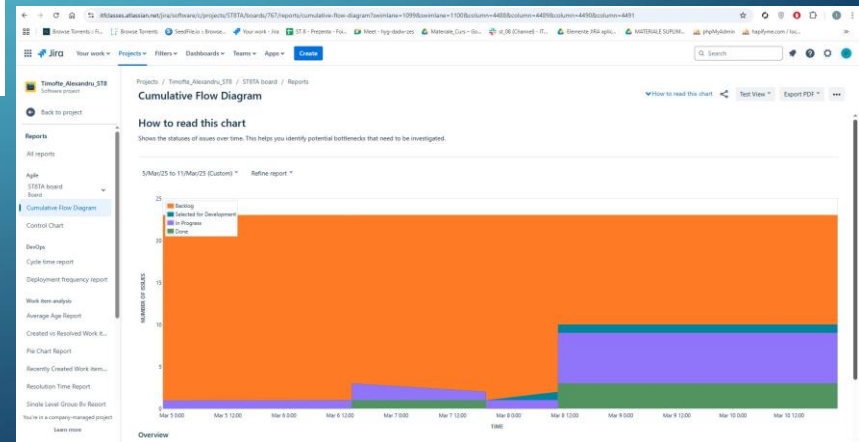
- 1. Forward Traceability** – Urmărește cerințele și asigură că fiecare cerință are un caz de testare asociat.
- 2. Backward Traceability** – Verifică dacă fiecare caz de testare este legat de o cerință, evitând testarea inutilă.
- 3. Bidirectional Traceability** – Oferă trasabilitate în ambele direcții, asigurându-se că toate cerințele sunt testate și că fiecare test corespunde unei cerințe.



The screenshot shows the JIRA Traceability Matrix for a project named 'Timothy_Alexander_STB'. The table displays requirements and their associated test cases, categorized by status (To Do, In Progress, Done).

Requirements	Test Cases	Status
STB1.1	STB1.1.1	To Do
STB1.2	STB1.2.1	To Do
STB1.3	STB1.3.1	To Do
STB1.4	STB1.4.1	To Do
STB1.5	STB1.5.1	To Do
STB1.6	STB1.6.1	To Do
STB1.7	STB1.7.1	To Do
STB1.8	STB1.8.1	To Do
STB1.9	STB1.9.1	To Do
STB1.10	STB1.10.1	To Do

Acces catre GitHub



The background is a blue gradient with faint concentric circles. White circuit-like lines with circular nodes are positioned in the corners: top-left, top-right, bottom-left, and bottom-right.

Mulțumesc pentru atenție!