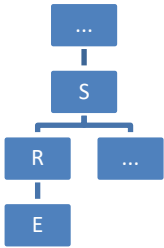


Obiective

- Testarea aplicațiilor la diferite niveluri de testare, e.g., testare unitară, testare de integrare.
- Utilizarea tool-urilor folosite în procesul de testare (TestLink, Jenkins, JUnit, Maven, Git, Mockito).

Cerințe

Să se realizeze următoarele task-uri:

Task, puncte	Descriere task
	Identificați la nivelul arhitecturii clase particulare, codificate aici prin E, R și S care împlinesc următoarele condiții: <i>E reprezintă o entitate din domeniul problemei. R este un repository cu elemente de tip E. S permite gestionarea repository-ului. Pentru E s-a realizat unit testing în Lab02.</i>
[Unit Testing. Mockito] 2 puncte	<p>Realizați testarea în izolare (unit testing) pentru clasele R și S, folosind framework-ul Mockito (vezi Tutorial Mockito). Se vor descrie teste care folosesc mock sau spy ca test doubles. Pentru fiecare clasă testată se va descrie câte o clasă de test separată cu minimum 2 teste.</p> <p>Nu se cere:</p> <ul style="list-style-type: none"> • elaborarea de fișiere similare cu Lab02_BBT_TCs_Form.xls și Lab03_WBT_TCs_Form.xls; • crearea în TestLink a unor suite cu cazuri de testare asociate explicit testelor scrise pentru R și S.
[Integration Testing. Mockito] 2 puncte	<p>Considerăm în aplicația dezvoltată existența următoarei diagrame de dependență dintre module, unde E, R și S corespund claselor deja testate în izolare.</p>  <pre> graph TD S[S] --- R[R] S --- Dots1[...] R --- E[E] </pre> <p>Realizați testarea de integrare utilizând strategia de integrare incrementală top-down. Se vor evidenția următoarele clase cu teste:</p> <p>[rezolvat] Step 1. testare unitară pentru E (Lab02), R (Lab04) și S (Lab04);</p> <p>Step 2. integrare R (se testează S cu R, pentru E se folosesc <i>stubbed mocks</i>);</p> <p>Step 3. integrare E (se testează S cu R și E);</p> <p>Pentru Step 2. și Step 3. fiecare clasă de test va avea minimum 2 teste.</p>
[TestLink] 2 puncte	<p>În cadrul proiectului PrjAAA, corespunzător userului xyir1234 utilizat anterior pentru Lab03, se vor realiza următoarele task-uri:</p> <ol style="list-style-type: none"> 1.1. definiți planul de testare xyir1234_IntT_TP în cadrul proiectului PrjAAA (secțiunea <i>Test Plan</i>); 1.2. creați suita de teste xyir1234_IntT care va conține 6 cazuri de testare, câte două cazuri de testare pentru fiecare pas de realizare a strategiei de integrare Top down (secțiunea <i>Test Specification</i>); 1.3. asociați cazurile de testare create la planul xyir1234_IntT_TP; 1.4. asociați cazurile de testare create la cerințele create anterior, după caz, la xyir1234_F01 sau xyir1234_F02; 1.5. generați documentația aferentă din (secțiunea <i>Test Specification</i>, opțiunea <i>Generate Test Specification Document</i>) în format .docx.
[Jenkins] 2 puncte	<p>Se va crea, configura și executa câte un job (vezi Tutorial Jenkins) pentru fiecare plan de testare creat anterior, i.e., xyir1234_BBT_TP, xyir1234_WBT_TP, xyir1234_IntT_TP. Job-urile create vor fi denumite xyir1234Job_BBT, xyir1234Job_WBT și xyir1234Job_IntT.</p> <p>Observație:</p> <ul style="list-style-type: none"> • pentru fiecare test creat în TestLink se va verifica (1) dacă este un test executat automat și (2) dacă are setat corect numele clasei de test și numele metodei de test (vezi Tutorial TestLink, pagina 7).
[TestLink+ Jenkins] 1 punct	Vizualizarea rezultatului execuției fiecărui job Jenkins în TestLink (vezi Tutorial Jenkins). Pentru fiecare test case executat este necesar să apară statusul passed .
[Git] 1 punct	<p>Se va actualiza conținutul repository-ului Git cu documentele elaborate în cadrul acestei teme:</p> <ul style="list-style-type: none"> • în folderul Docs/Lab04 fișierul cu documentația generată în TestLink; • pachetul/ele cu teste implementate în Java pentru Lab04; • dacă este cazul, codul sursă modificat în urma depanării.

Predarea temei de laborator**[Unit Testing. Mockito]**

- implementarea claselor cu teste pentru clasele R și S, folosind **Mockito**.

[Integration Testing. Mockito]

- implementarea claselor cu teste folosind strategia de integrare **Top Down**.

[TestLink+Jenkins]

- Câte un job pentru fiecare dintre cele trei planuri de testare create în TestLink: **xyir1234Job_BBT**, **xyir1234Job_WBT** și **xyir1234Job_IntT**.
- Fiecare test din TestLink are statusul **passed**.

Termene de predare

Săptămâna	Tema de laborator			Primul termen de predare	Ultimul termen de predare
S07	L04.	Niveluri de testare	JUnit, TestLink, Jenkins, Git, Mockito	S09	S11*°
S08				S10	S12*°

*) Temele restante se vor putea preda în limita timpului disponibil.

°) Se pot preda cel mult două teme de laborator.