Tema de laborator

Data	Versiune	Comentarii
2022-11-13	1.0	Enunt initial

Sa se creeze o aplicatie pentru gestiunea activitatii unei biblioteci de carti.
Despre carti:

- Fac parte din cel putin un domeniu;
- Domeniile se pot imparti in subdomenii, ex. Stiinta -> {Matematica, Fizica, Chimie, Informatica}; Informatica -> {Algoritmi, Programare, Baze de date, Retele de calculatoare, …}; Algoritmi -> {Algoritmica grafurilor, Algoritmi cuantici, …}
- Daca o carte face parte dintr-un subdomeniu, automat va fi regasita si ca fiind parte din domeniile stramos, fara ca acest lucru sa fie declarat explicit in incadrarea initiala a cartii; exemplu: daca o carte face parte din domeniul Baze de date, automat e si din domeniile “Informatica” si “Stiinta”.
- Se va verifica faptul ca o carte nu poate sa se specifice explicit ca fiind din domenii aflate in relatia stramos-descendent. De exemplu, e o eroare sa se incadreze o carte in mod explicit atat in domeniul Algoritmi, cat si in domeniul Stiinta (domeniu stramos);
- O carte nu poate face parte din mai mult de DOMENII domenii, unde DOMENII e setat din aplicatie (valoare in fisier de configurare sau baza de date)
- O carte poate avea oricati autori; o carte poate fi tiparita in mai multe editii (e.g. Editura stiintifica, 1980; Editura All, 2000 editia 1; Editura All, 2010, editia a treia etc). Pentru fiecare editie se vor memora elemente specifice (numarul de pagini, tip carte)
- Din anumite carti, o parte din exemplare (sau toate) sunt doar pentru sala de lectura
- O carte poate fi imprumutata doar daca: nu are toate exemplarele marcate ca fiind doar pentru sala de lectura si numarul de carti ramase (inca neimprumutate, dar nu din cele pentru sala de lectura) este macar 10% din fondul initial din acea carte.

Despre cititori:

- Au date personale, ce trebuie asigurate ca fiind intr-o forma consistente (nume, prenume, adresa etc.); pentru contactare, macar unul din: numar de telefon, adresa de email trebuie sa fie specificate; impuneti alte conditii de validitate
- Pot imprumuta un numar maxim de carti NMC intr-o perioada PER;
- La un imprumut pot prelua cel mult C carti; daca numarul cartilor imprumutate la o cerere de imprumut e cel putin 3, atunci acestea trebui sa faca parte din cel putin 2 categorii distincte
- Nu pot imprumuta mai mult de D carti dintr-un acelasi domeniu – de tip frunza sau de nivel superior - in ultimele L luni
- Pot imprumuta o carte pe o perioada determinata; se permit prelungiri, dar suma acestor prelungiri acordate in ultimele 3 luni nu poate depasi o valoare limita LIM data
- Nu pot imprumuta aceeasi carte de mai multe ori intr-un interval DELTA specificat, unde DELTA se masoara de la ultimul imprumut al cartii
- Pot imprumuta cel mult NCZ carti intr-o zi.

Despre personalul bibliotecii:

- Daca sunt inregistrati drept cititori cu acelasi cont, atunci in cazul lor valorile pragurilor NMC, C, D, LIM se dubleaza, DELTA si PER se injumatatesc.
- Nu pot acorda mai mult de PERSIMP carti intr-o zi; limita NCZ se ignora pentru ei.

Aplicatia trebuie sa implementeze verificarea acestor constrangeri; testele trebuie sa urmareasca diferitele scenarii, conditii complexe in combinatie, validarea starii obiectelor. Notarea aplicatiei se va face prin rularea unitatilor de test scrise. Unitatile de test vor include verificarea coerentei la nivel de service layer (daca parametrii trimisi unor metode de serviciu nu sunt cu valori acceptabile, e de asteptat ca metoda sa arunce exceptie sau sa returneze un rezultat din care se deduce faptul ca nu se poate indeplini cererea); validati fluxul de mesaje, starea + contextul curent.

Cerinte:

1. Unit testing folosind NUnit, xUnit sau Visual Studio 2019+; minim 200 de metode de testare; se vor testa daca valorile proprietatilor au valori adecvate, restrictiile de mai sus si altele de validare a corectitudinii rezultatelor si a fluxului de lucru.

2. Folosire de blocuri din suita Microsoft Enterprise Library: logging (se poate folosi alternativ log4net), validation application block (sau se poate scrie cod de validare manual, sau se poate folosi o biblioteca de tipul FluentValidation), optional: exception handling application block, optional: security application block. Nu se face logging in unit testing (se permite in straturile aplicatiei, insa). Validarea starii obiectelor prin teste este conditie obligatorie pentru luarea in considere a temei. Logging-ul este obligatoriu pentru luarea in considerare a temei.

3. Dezvoltare pe straturi (layers): Domain layer si Data Layer; nu se ia in considerare/nu se cere user interface.

4. Baza de date relationala : SQL Server 2017+, MySQL, Oracle sau PostgreSQL.

5. Domain Layer implementat prin Domain Model; Data Layer implementat prin Data Mapper.

8. Code coverage de minim 90% pentru codul de Domain Model si minim 90% pentru Service Layer.

9. Mocking.

10. Comentarii, minim pentru metode, constructori, proprietati, indexatori, destructori, tipuri imbricate. Puteti folosi GhostDoc.

11. Se va face analiza codului sursa folosind StyleCop (plugin de Visual Studio; versiune recomandata 4.7.x). Toate regulile (tab-ul Rules) trebuie bifate; in tab-ul Company Information la sectiunea „Company Name” sa figureze „Transilvania University of Brasov”, iar la Copyright numele studentului. Maxim 30 de avertismente sunt permise.

12. Pragurile scrise cu litere mari (NCZ, C etc.) se vor specifica de asa maniera incat modificarea valorilor asociate sa nu necesite recompilarea aplicatiei.

Puteti folosi pluginuri auxiliare de forma CodeItRight.

Observatii:

- **Elementele scrise cu rosu sunt obligatorii si eliminatorii.**
- **Urmariti versiunile ulterioare ale documentului.**