Pràctica 2: UBFLIX: Iteració 1 - Disseny i implementació

DS - curs 2019-2020

Objectius:

L'objectiu principal d'aquesta pràctica és aprendre a realitzar el disseny d'una aplicació a partir de la seva especificació utilitzant la metodologia Test Driven Development (TDD): des dels test d'acceptació es dissenyen les classes necessàries (basant-nos en el Model de Domini) aplicant els principis de disseny GRASP, el patró arquitectònic Model-Vista-Controlador i patrons de disseny. Com a objectiu de disseny s'aprendrà a identificar els patrons més adients i la seva aplicació en un problema concret.

Enunciat de la pràctica 2:

A partir del Model de Domini dissenyat en la pràctica anterior i els tests d'acceptació definits en Concordion, en aquesta pràctica es dissenyarà i implementaran les següents funcionalitats:

1. Alta d'un nou client
2. Login d'un nou client (es dóna parcialment dissenyat)
3. Alta d'un nou usuari
4. Login d'un nou usuari
5. Veure Perfil
6. Llistar el catàleg de sèries
7. Mostrar els detalls d'una sèrie
8. Marcar la Sèrie per a que surti en el llistat MyList (NOTA: és la Sèrie la que es marca per a que aparegui a MyList, no l'episodi com el cas UC5)
9. Visualitzar un nou episodi
10. Valoració d'un episodi
11. Visualització del llistat Watched List (sèries ja vistes i acabades)
12. Visualització del llistat Continue Watching (sèries on queden episodis per veure)
13. Visualització del llistat MyList (sèries on s'han marcat com a sèries que es volen veure més endavant. Quan la sèrie ja s'ha vist, no surt en el llistat MyList)
14. [OPCIONAL] Modificar perfil usuari
15. [OPCIONAL] Cerques Per Nom, Per Etiqueta (temàtica), Per Visualitzacions i Per Valoracions.
16. [OPCIONAL] Llistar les sèries recomanades de les visualitzacions anteriors i de les valoracions que n'han fet els altres usuaris amb sèries de temàtiques similars (Top Picks for You).

Es suposa que quan **UBFLIX** ofereix una sèrie en el seu catàleg, la sèrie té com a mínim una temporada amb tots els seus episodis. Com podeu observar, potser hi han històries d'usuari que potser han variat una mica (com la de marcar sèrie i no episodi per a sortir al llistat de MyList).

Per a realitzar aquesta pràctica es seguirà el Model-Vista-Controlador explicat a classe de teoria, desenvolupant la part del Model i del Controlador. La part de la Vista es fa a partir dels tests de Concordion. Així, l'output d'aquesta pràctica no serà mitjançant la consola o una finestra gràfica, sinó que serà mitjançant els tests d'acceptació que vosaltres programareu amb Concordion. Les parts de Controlador i Model s'han d'implementar en dues carpetes separades (controller i model, respectivament) de la carpeta

src del projecte. La part del projecte que té relació amb la capa de recursos es guardarà en una carpeta interior a src, anomenada resources.

Metodologia TDD

La metodologia a seguir es basa en TDD i és la següent:

1. S'escull el test d'acceptació a implementar del fitxer html de la carpeta de test. A partir d'ell, s'identifiquen les classes del Model de Domini que cal tenir en compte.

2. S'escriu el codi del test d'acceptació en Java a la carpeta de test del projecte. En aquesta carpeta de test es crearà una carpeta per a cada història d'usuari, contenint la seva especificació (html) i la fixture (java). Pots trobar un exemple en el projecte base de la pràctica del github.

3. Només has de desenvolupar el codi necessari pel test d'acceptació a la carpeta src la part del controlador i del model que estiguin implicades en aquest test d'acceptació, tot mirant si es poden aplicar algun dels patrons explicats a classe i tot seguint els principis S.O.L.I.D.

4. Es prova el test via Concordion i tots els tests anteriors per a validar que segueixen funcionant.

5. Es redisenya (refactoritza) el codi per evitar duplicats i es tornen a validar els patrons utilitzats. S'hi inclouen de nous, si és necessari.

6. Es comprova que tots els test segueixen funcionant.

7. Es procedeix a implementar el següent test d'acceptació (anar al pas 1).

Així, per cadascuna de les funcionalitats demanades, heu de partir de la història d'usuari i dels tests d'acceptació associats a ella, per a validar el seu bon funcionament. A part, haureu de fer **un add/commit/push a github per cada història d'usuari** que implementeu. Això vol dir, que en finalitzar l'entrega haureu de tenir, com a mínim 13 commits/push a github (un per cada història d'usuari). Normalment, es procedeix a fer un commit/push a cada test d'acceptació per a no acumular molts canvis del projecte local en relació al remot.

Especificacions tècniques (material de suport pel desenvolupament):

- Per la pràctica es proporciona un software inicial. Cal replicar-lo a partir de l'enllaç següent:

  https://classroom.github.com/g/Sh99DKB9

- El software es basa en una arquitectura en tres capes: (1) la capa de vista que és la que correspon als tests, (2) la capa de lògica de negoci (on està el model i el controlador) i (3) la capa de recursos o persistència, que és l'encarregada de guardar les dades. La capa de recursos o persistència es podrà substituir per l'accés a una bases de dades compartides amb tota la classe amb dades sobre les pel·lícules, clients, artistes, temàtiques i valoracions. Quan s'inicïi l'aplicació, s'inicialitzaran totes les dades de la capa del model necessàries per a l'aplicació.
- En el repositori de github es proporciona un codi de la part de persistència (o recursos) que cal utilitzar i ampliar. En aquesta part s'utilitzen els patrons de disseny d'AbstractFactory i DAO per a poder canviar fàcilment la capa de persistència.

- En aquesta pràctica cal que inicialitzeu les dades des d'un MOCK de la Base de Dades (un *mock* és un objecte que simula un altre objecte a efectes de test). En el projecte base que us proporcionem hi han dos Mocks, un per inicialitzar Clients i un altre per inicialitzar Sèries. Cal que feu els que inicialitzen les altres dades de l'aplicació.
- En el projecte base també es proporciona un test de control del login de l'usuari mitjançant Concordion, per entendre l'organització del codi i de les especificacions. Sobre aquest projecte podeu basar l'arquitectura de la vostra pràctica.

Instruccions per al lliurament

Cal lliurar la memòria IMPRESA el dia del lliurament.

Tots els lliuraments es presenten junt amb una MEMÒRIA explicativa que inclogui com a mínim els següents punts:

- Portada amb títol, nom, cognom i NIUB dels dos membres del grup que han realitzat la pràctica
- Índex de la memòria paginat
- Model de classes generat des del codi (usant el plugin de IntelliJ anomenat Sketch it!). Potser caldrà que repasseu el diagrama de classes final, doncs de vegades el plugin de Sketch it! no detecta totes les relacions o les classes.
- Observacions i justificacions dels patrons utilitzats o les decisions de disseny fetes per a seguir els principis de disseny S.O.L.I.D.
- Conclusions de la pràctica detallant el repartiment de feina entre els integrants de l'equip.
- No oblidis d'incloure els canvis que s'han realitzat de la pràctica anterior.

El dia del lliurament es penjarà en el campus virtual un fitxer comprimit en **format ZIP** amb el nom dels tres membres del grup i el numero de lliurament com a nom de fitxer. Per exemple, BartSimpsonLisaSimpsonHommerSimpsonL2.zip, on L2 indica que es el "lliurament 2". El fitxer ZIP inclourà: la memòria en format pdf i el projecte de IntelliJ final corresponent al projecte de la tasca del classroom de github del teu grup.