# Pràctica 2: Grups C i F

## Objectius:

L'objectiu principal d'aquesta pràctica és avaluar el codi desenvolupat a la pràctica 1 usant els principis disseny GRASP, SOLID i el patró arquitectònic per capes integrant l'arquitectura clàssica de Model-Vista-Controlador.
Com a objectiu secundari, s'aprendrà a utilitzar el patró DAO per a codificar la capar de recursos.

## Enunciat de la pràctica 2:

Seguint amb les funcionalitats relatives fer comentaris sobre les i pagar les activitats fetes per cada Soci, es suposa que teniu la següent llista de Històries d'Usuari implementades:

- Històries UC1. Enregistrar-se
- Històries UC2. Login
- Històries UC3. Llistar excursions
- Històries UC4. Cercar Excursió
- **Històries UC9. Pagar Activitat**
- **Històries UC10. Visualitzar Pagaments Soci**
- **Històries UC11. Comentar Activitat**
- **Històries UC13. Visualitzar Comentaris d'una Activitat**
- **Històries UC14. Visualitzar TOP 10 Activitats per Comentaris**

Tingues en compte que l'aplicació podrà estar offline (sense connexió externa) i per tant, totes les dades hauran d'inicialitzar-se a memòria. Només caldrà estar online en el moment que es registre o es fa un login d'un Soci.

Aquesta pràctica consta de dues parts:

1. Analitzar el codi sota la perspectiva de l'acoblament i la cohesió de les classes que has desenvolupat i implementat a la pràctica 1
2. Permetre la inicialització de les dades a partir de diferents recursos: d'unes classes que "simulen" una base de dades (DAO-MOCK) o a partir d'unes classes que guarden a disc (DAO-JSON)

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 heu programat amb Concordion. Les parts de Controlador i Model s'han d'implementar en dues carpetes separades (controller i model, respectivament) dins 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.

## NOTA: Material de suport pel desenvolupament de la pràctica 2:

> Si ets sents confortable amb la teva pràctica 1, segueix amb el teu codi i el teu projecte d'IntelliJ. En el cas que no l'hagis pogut lliurar a temps o saps que el teu lliurament està a mitges, replica el projecte solució a partir del següent enllaç:
>
> https://classroom.github.com/a/ScwCbBr0

## PART 1: Refactoring seguint els criteris GRASP

Segueix els següent passos i contesta els següents punts:

1. Extreu el diagrama de classes del teu codi de la pràctica 1 utilitzant el plugin Sketch it! De IntelliJ. Completa el fitxer plantUML per a detallar totes les dependències entre classes. Inserta aquí la imatge del diagrama de classes del teu projecte:

2. Detecta males olors del teu codi:

    a. Identifica quines classes estan més acoblades mb d'altres

    b. Identifica quines estan menys cohesionades.

3. Segueix la metodologia TDD per anar refactoritzant el codi pas a pas:

    a. Dedica't a UN test d'acceptació per començar a veure com es comporta el codi internament en el teu diagrama de classes anterior.

    b. Intenta seguir les recomanacions GRASP per poder millorar el codi d'aquest test. Recorda que els patrons GRASP els pots trobar a les transparències de classe de teoria: Baix Acoblament i Alta Cohesió, Expert en Informació, Creador, Controlador, Fabricació Pura, Indirecció, Polimorfisme i Variacions Protegides. Els tens explicats també en el vídeo 18 – Patrons GRASP. No vol dir que hagis d'usar tots els patrons a cada test, sinó potser et serveixen de guia per poder modificar el codi.

    c. Refactoritza segons els criteris que pensis i prova el test via Concordion i tots els tests anteriors per a validar que segueixen funcionant. Potser que ara no vagi algun dels tests posteriors però fixa't només en els que vagis refactoritzant

    d. Es puja el github el test fet i validat amb el missatge que identifica el test correctament i el patró o patrons GRASP que has usat.

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

Així, per cadascuna de les funcionalitats demanades, haureu de fer **un add/commit/push a github per cada test d'acceptació** implementat amb el comentari adient del test i quin patró GRASP heu usat. Això vol dir, que en finalitzar l'entrega haureu de tenir, com a mínim tants commits/push a github com tests d'acceptació. 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. Al final del punt 3, llista aquí per cada tests d'acceptació el criteri GRASP que has fet servir i explica breument què has canviat en el teu codi inicial, Afegeix files a la taula si així ho necessites:

| Test d'acceptació | Criteri GRASP | Breu explicació de les classes canviades |
|---|---|---|

| | | |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

4. Detalla el diagrama final de classes que has obtingut al final de la refactorització de tots els tests d'acceptació. Si detectes encara alguna "mala olor" en el codi", explica-la aquí breument:

## PART 2: Codificant la capa de recursos

En aquesta part, aprendrem a utilitzar i codificar el patró DAO per inicialitzar les dades de l'aplicació usant de forma flexible recursos externs com poden ser fitxers o bases de dades. Per això, segueix els següent passos i contesta els punts que ho requereixin:

1. Clona el projecte base que trobaràs al classroom:

   https://classroom.github.com/a/Ys-xd_Cj

2. Explora com el projecte s'estructura en una arquitectura per capes:

   - 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 excursions, activitats, socis, etc.. Quan s'inicii l'aplicació, s'inicialitzaran totes les dades de la capa de lògica de negoci 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 aquest projecte base s'inicialitzen 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 Socis i un altre per inicialitzar Excursions.

   - En el projecte base també es proporciona els mateixos tests de la pràctica 1 base. Trobareu una descripció més detallada en el Manual del Campus Virtual.

3. En relació al codi de la capa de persistència (carpeta recursos i data service), quins patrons GRASPS s'estan identificant?. Llista'ls a continuació, justificant breument la teva resposta (afegeix files a la taula si és necessari):

| Patró GRASP usat a la capa de persistència | Breu Justificació |
|---|---|

| | |
|---|---|
| Per exemple, Cohesió alta a les classes DAO-MOCK | Les classes DAO-MOCK internament només tenen la responsabilitat de gestionar les dades com si fossin una base de dades i cap altra |
| | |
| | |

4. Modifica aquest projecte base per inicialitzar especies en el seu DAO corresponent i no des dels tests com es fa fins ara (mira CercaExcursions a la carpeta de tests). Quines classes caldria que actualitzessis? Per això intenta resseguir el codi des del Controlador per veure com es fa la inicialització de les excursions, per exemple. Intenta modificar el codi per donar d'alta les Espècies.

5. [OPT] Integra el codi que gestiona la capa de persistència a la teva pràctica resultat del pas 1. Vigila d'integrar poc a poc, mètodes del controlador des del projecte DAO al teu codi.

6. [OPT] Afegeix els DAOs corresponents a noves dades que necessites. Com gestionaràs la càrrega d'activitats referents a una excursió? Raona la teva solució a continuació, justificant-la segons els patrons GRASP.

Finalment inclou aquí el repartiment de la feina que heu decidit per realitzar aquesta pràctica:

Estudiant 1:

Estudiant 2:

Observacions finals: (si vols explicar algun altre comentari=

## Instruccions per al lliurament

Tots els lliuraments es presenten junt amb una còpia d'aquest document amb les respostes incloses a cada pregunta.

El dia del lliurament es penjarà en el campus virtual un fitxer comprimit en **format ZIP** amb el nom dels dos membres del grup i el numero de lliurament com a nom de fitxer. Per exemple, A01-BartSimpsonLisaSimpsonL2.zip, on L2 indica que es el "lliurament 2" de l'equip A01. El fitxer ZIP inclourà: aquest document amb les respostes en format pdf i el projecte de IntelliJ final corresponent al projecte de la tasca del classroom de github del teu grup.