

UNIVERSITAT POLITÈCNICA DE CATALUNYA

SISTEMA RECOMANADOR

METODES I ATRIBUTS - SEGONA ENTREGA - v1.0

PROJECTES DE PROGRAMACIÓ

Autors

HECTOR Pueyo Casas
(hector.Pueyo@estudiantat.upc.edu)

AGNÈS FELIP I DÍAZ
(agnes.felip@estudiantat.upc.edu)

MIQUEL FLORENSA
(miquel.florensa@estudiantat.upc.edu)

DAVID LATORRE
(david.latorre.romero@estudiantat.upc.edu)

Supervisor

SERGIO ÁLVAREZ NAPAGAO



UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

| | |
|--------------------------------------|----------|
| 1. DOMINI | 7 |
| 1.1. ALGORISME | 7 |
| 1.2. ATRIBUT | 7 |
| 1.2.1. addItem | 7 |
| 1.2.2. deleteItem | 7 |
| 1.3. ATRBOOL | 8 |
| 1.4. ATRDOUBLE | 8 |
| 1.5. ATRINT | 8 |
| 1.6. ATRSTRING | 8 |
| 1.7. CBFILTERING | 8 |
| 1.7.1. recalculaRecomanacions | 9 |
| 1.7.2. knn | 9 |
| 1.8. COLFILTERING | 9 |
| 1.8.1. recalculaRecomanacions | 9 |
| 1.9. DCG | 10 |
| 1.9.1. calculaDCG | 10 |
| 1.10. DISTANCIA | 10 |
| 1.11. DISTITEM | 10 |
| 1.11.1. afegirItem | 11 |
| 1.11.2. eliminarItem | 11 |
| 1.12. ITEMDISTITEM | 12 |
| 1.13. DISTUSUARI | 12 |
| 1.13.1. afegirUsuari | 13 |
| 1.13.2. eliminarUsuari | 13 |
| 1.13.3. recalcularTotesDistancies | 13 |
| 1.14. EXCEPCIONS | 13 |
| 1.15. HYBRID | 14 |
| 1.15.1. calculRecomanacions | 14 |
| 1.16. ITEM | 14 |
| 1.16.1. addValoracio | 14 |
| 1.16.2. deleteValoracio | 15 |
| 1.17. KMEANS | 15 |
| 1.17.1. calcularClusters | 15 |
| 1.18. SLOPEONE | 16 |
| 1.18.1 prediccio | 16 |
| 1.19. TIPUSATRIBUT | 16 |
| 1.19.1. afegirAtribut/eliminaAtribut | 17 |
| 1.19.2. crearAtribut | 17 |

| | |
|--|----|
| 1.20. UACTIU | 17 |
| 1.20.1. eliminarItemCreat/afegirItemCreat | 18 |
| 1.21. USUARI | 18 |
| 1.21.1. isActiu | 18 |
| 1.21.2. haValorat | 18 |
| 1.21.3. addItemValoracio/eliminarValoracio | 18 |
| 1.22. VALORACIO | 18 |
| 1.24. PairComparator | 18 |
| 1.24.1. compare | 18 |
| 1.25. StrategyTipus | 19 |
| 1.26 StrategyString | 19 |
| 1.26.1. crearAtribut | 19 |
| 1.27. StrategyBool | 19 |
| 1.27.1. crearAtribut | 19 |
| 1.28. StrategyDouble | 19 |
| 1.28.1. crearAtribut | 19 |
| 1.29. StrategyInt | 19 |
| 1.29.1. crearAtribut | 19 |
| 1.30. CONTROLADORS DOMINI | 20 |
| 1.30.1. CtrlDomini | 20 |
| 1.3.0.1.1. inicialitzarCtrlDomini | 20 |
| 1.3.0.1.2. validarPath | 20 |
| 1.3.0.1.3. setTipusAtributs | 20 |
| 1.3.0.1.4. canvisTipusAtributArxiuItems | 20 |
| 1.3.0.1.5. carregarRecomanacions | 21 |
| 1.3.0.1.6. carregarUsuarisActius | 21 |
| 1.3.0.1.7. getPosicioId | 21 |
| 1.3.0.1.8. afegirValoracionsUA | 21 |
| 1.3.0.1.9. registraUsuari | 21 |
| 1.3.0.1.10. iniciarSesio | 21 |
| 1.3.0.1.11. getNomTipusAtributs | 21 |
| 1.3.0.1.12. getTipusAtributs | 21 |
| 1.3.0.1.13. setTipusAtributId | 21 |
| 1.3.0.1.14. almenysUnaValoracioFeta | 22 |
| 1.3.0.1.15. nomsItemsRecomanats | 22 |
| 1.3.0.1.16. setTipusAtributTitol | 22 |
| 1.3.0.1.17. existeixItem | 22 |
| 1.3.0.1.18. getIdItem | 22 |
| 1.3.0.1.19. getValorsTipusAtributDeItem | 22 |
| 1.3.0.1.20. creaValoracio | 22 |

| | |
|--|----|
| 1.3.0.1.21. eliminaValoracio | 22 |
| 1.3.0.1.22. lecturaHeader | 22 |
| 1.3.0.1.23. inicialitzarItems | 22 |
| 1.3.0.1.24. mapeigRatings | 23 |
| 1.3.0.1.25. inicialitzarUsuarisValoracions | 23 |
| 1.3.0.1.26. comptarQuotes | 23 |
| 1.3.0.1.27. consultaTipusTipusAtributs | 23 |
| 1.3.0.1.28. consultaValoracionsUsuari | 23 |
| 1.3.0.1.29. eliminarUsuariActiu | 23 |
| 1.3.0.1.30. eliminaItem | 23 |
| 1.3.0.1.31. creaItem | 23 |
| 1.30.2. CtrlDominiConfiguracio | 24 |
| 1.3.0.2.1. getDadesPerFitxer | 24 |
| 1.3.0.2.2. getTipusTipusAtributs | 24 |
| 1.30.3. CtrlDominiItem | 24 |
| 1.3.0.3.1. getTitolsItems | 24 |
| 1.3.0.3.2. existeixItem | 24 |
| 1.3.0.3.3. getIdItem | 24 |
| 1.3.0.3.4. getValorsTipusAtributDeItem | 24 |
| 1.3.0.3.5. eliminarItem/afegirItem | 24 |
| 1.30.4. CtrlDominiUsuari | 25 |
| 1.3.0.4.1. registraUsuari | 25 |
| 1.3.0.4.2. iniciarSesio | 25 |
| 1.3.0.4.3. afegirValoracio/eliminarValoracio | 25 |
| 1.3.0.4.4. checkCreador | 25 |
| 1.3.0.4.5. eliminarItem/crearItem | 25 |
| 1.30.5. CtrlDominiValoracio | 25 |
| 1.3.0.5.1. afegirValoracio/eliminarValoracio | 25 |
| 1.3.0.5.2. almenysUnaValoracioFeta | 25 |
| 1.3.0.5.3. eliminarUA | 25 |
| 1.3.0.5.4. eliminarItem | 25 |
| 1.30.6. CtrlDominiValoracio | 26 |

| | |
|---------------------------------------|-----------|
| 2. PERSISTÈNCIA | 26 |
| 2.1 CONTROLADORS PERSISTÈNCIA | 26 |
| 2.1.1 CtrlPersistencia | 26 |
| 2.1.1.1. inicialitzarCtrlPersistencia | 26 |
| 2.1.1.2. validaPath | 26 |
| 2.1.1.3. inicialitzarControladors | 26 |
| 2.1.1.4. canvisTipusAtributArxiuItem | 27 |
| 2.1.1.5. llegeixLlíniesArxiu | 27 |

| | |
|--|-----------|
| 2.1.1.6. llegeixArxiu | 27 |
| 2.1.1.7. carregarUsuaris | 27 |
| 2.1.1.12. getNomId | 28 |
| 2.1.1.13. getNomTitol | 28 |
| 2.1.1.14. eliminarUA | 28 |
| 2.1.1.15. setHeaderVal | 28 |
| 2.1.1.16. eliminarItem | 28 |
| 2.1.1.17. afegirItem | 28 |
| 2.1.2. CtrlPersistenciaConfiguracio | 28 |
| 2.1.2.1. comprovarCanvisItems | 28 |
| 2.1.2.2. comprovacioFitxers | 29 |
| 2.1.2.3. getNomId | 29 |
| 2.1.2.4. crearFitxerConfig | 29 |
| 2.1.2.5. afegirTipusAtribut | 29 |
| 2.1.2.6. getTipusAtributs | 29 |
| 2.1.2.7. getNomTitol | 29 |
| 2.1.3. CtrlPersistenciaItem | 29 |
| 2.1.3.1. eliminarItem/afegirItem | 29 |
| 2.1.4. CtrlPersistenciaRecomanacio | 29 |
| 2.1.4.1. crearFitxerRecomanacions | 29 |
| 2.1.4.2. eliminarRecomanacionsUsuari/afegirRecomanacionsUsuari | 29 |
| 2.1.4.3. eliminarRecomanacioItem | 30 |
| 2.1.4.3. getRecomanacionsUsuari | 30 |
| 2.1.5. CtrlPersistenciaUsuari | 30 |
| 2.1.5.1. crearFitxerUsuarisActius | 30 |
| 2.1.5.2. afegirUsuariActiu/eliminarUsuariActiu | 30 |
| 2.1.6. CtrlPersistenciaValoracio | 30 |
| 2.1.6.1. crearFitxerValoracionsUA | 30 |
| 2.1.6.2. detectarHeader | 30 |
| 2.1.6.3. afegirValoracio | 30 |
| 2.1.6.4. eliminarValoracio | 30 |
| 3. PRESENTACIÓ | 31 |
| 3.1 CtrlPresentacio | 31 |
| 3.2 VistaPrincipal | 31 |
| 3.3 VistaSesioIniciada | 33 |
| 3.4 panellRecomanacio | 37 |
| 3.5 panellBusca | 39 |
| 3.6 panellAdministraValoracions | 41 |

1. DOMINI

1.1. ALGORISME

Algorisme és la classe mare de les classes [CBFiltering](#) i [COLFiltering](#). Aquesta es troba buida.

1.2. ATRIBUT

Atribut és la classe mare de totes les classes que comencen per Atr. Aquesta està formada per dos atributs privats; *items*, un Set d'Ítems que guarda tots els ítems que contenen aquest atribut, i *tipusAtribut* de tipus [TipusAtribut](#) que defineix de quin tipus és l'atribut i si és calculable per l'algorisme.

La classe té dos constructores, una crea una instància d'atribut sense items i l'altre crea una instància d'atribut amb un item inicial, i un getter per *tipusAtribut* i un setter per *items*.

Com a mètodes ens trobem amb l'operació *public void addItem(Item item)* i *public void deleteItem(Item item)*.

1.2.1. addItem

Afegeix un item al conjunt d'*items* que contenen l'atribut.

Paràmetre: El paràmetre *item* és el nou ítem que conté l'atribut.

Retorna: -

1.2.2. deleteItem

Esborra un item al conjunt d'*items* que contenen l'atribut.

Paràmetre: El paràmetre *item* és el ítem a eliminar del conjunt.

Retorna: -

1.3. ATRBOOL

AtrBool és una de les subclasses d'[Atribut](#). Aquesta classe té un sol atribut, que és el booleà privat *atributBool*, que guarda el valor d'un atribut de tipus booleà. Degut a la seva poca complexitat la classe tan sols té una funció pública, que seria el seu getter, i dues constructores.

1.4. ATRDOUBLE

AtrDouble és una de les subclasses d'[Atribut](#). Aquesta classe té un sol atribut, que és el double privat *atributDouble*, que guarda el valor d'un atribut de tipus Double. Degut a la seva poca complexitat la classe tan sols té una funció pública, que seria el seu getter, i dues constructores.

1.5. ATRINT

AtrInt és una de les subclasses d'[Atribut](#). Aquesta classe té un sol atribut, que és el int privat *atributInt*, que guarda el valor d'un atribut de tipus Integer. Degut a la seva poca complexitat la classe tan sols té una funció pública, que seria el seu getter, i dues constructores.

1.6. ATRSTRING

AtrString és una de les subclasses d'[Atribut](#). Aquesta classe té un sol atribut, que l'string privat *atributString*, que guarda el valor d'un atribut de tipus String. Degut a la seva poca complexitat la classe tan sols té una funció pública, que seria el seu getter, i dues constructores.

1.7. CBFILTERING

Content Based Filtering és la classe que calcula les recomanacions en funció de l'algorisme knn. Aquesta conté els atributs privats; *usuari*, que és l'Usuari sobre el que funciona l'algorisme, *recomanacions*, la un ArrayList<Item> en la que es guardaran les recomanacions, *totesDistàncies*, el HashMap<Item, HashMap<Item, Double>> amb totes les distàncies entre ítems, *bonesRecomanacions*, un Booleà que diu si les recomanacions són bones i *k*, un Integer amb el número de recomanacions que es volen obtenir.

La classe té un sol constructor i els seus getters.

En quant als modificadors públics ens trobem amb *recalculaRecomanacions*, els dos setters *editaK* i *editaDistancies* i els mètodes restants són tots privats. Per entendre millor el funcionament de la classe també parlarem del mètode privat *knn*.

1.7.1. recalculaRecomanacions

Funció que torna a calcular les recomanacions. Aquesta fa una crida a *knn()*

Paràmetre: El paràmetre *item* és el ítem a eliminar del conjunt.

Retorna: - No retorna re però actualitza l'atribut privat *recomanacions*.

1.7.2. knn

Algorisme de recomanació basat en *knn*, ens retorna els k ítems més semblants als ítems que ens han agradat.

Paràmetre: -

Retorna: *ArrayList<Item>* *recomanacions*, una llista de k recomanacions i a més guarda aquestes recomanacions.

1.8. COLFILTERING

Collaborative Filtering és la classe que calcula les recomanacions en funció dels l'algorismes [SlopeOne](#) i [KMeans](#). Aquesta conté els atributs privats; *usuari*, que és l'[Usuari](#) sobre el que funciona l'algorisme, *usuaris*, un *HashMap<Integer,Usuari>* sobre els que buscarem usuaris semblants, *recomanacions*, un *ArrayList<Item>* en la que es guardaran les recomanacions, *distàncies*, el *HashMap<Item, HashMap<Item, Double>>* amb totes les distàncies entre ítems, i *k*, un *Integer* amb el número de recomanacions que es volen obtenir.

La classe té un sol constructor i els seus getters.

En quant als mètodes públics ens trobem amb *recalculaRecomanacions*.

1.8.1. recalculaRecomanacions

Algorisme de recomanació basat en slopeOne i kMeans, ens retorna els k ítems més semblants als ítems que ens han agradat.

Paràmetre: -

Retorna: *ArrayList<Item>* *recomanacions*, una llista de k recomanacions i a més guarda aquestes recomanacions.

1.9. DCG

Discount Cumulative Gain (DCG) és la classe que avalua la qualitat de la recomanació feta pels algorismes. Aquesta conté els atributs privats; *DCG*, *IDCG*, *NDCG*, tots Integers, *LT*, *HashMap<Item,Double>* i *LR* un *ArrayList<Item>*.

La classe té un sol constructor i els seus getters.

En quant als mètodes públics ens trobem amb *calcula DCG*

1.9.1. calculaDCG

Calcula el DCG, IDCG i NDCG mitjançant la llista de recomanacions calculades pels algorismes i la llista de recomanacions ideals donades als arxius de test.

Paràmetre: -

Retorna: -

1.10. DISTANCIA

Distància és la classe mare de les classes [DistItem](#) i [DistUsuarí](#). Aquesta es troba buida.

1.11. DISTITEM

DistItem és una classe que, donat un conjunt d'items, calcula les distàncies entre tots aquests, és a dir, calcula totes les combinacions possibles de distàncies entre un conjunt d'items.

Per calcular aquestes distàncies, l'estructura que utilitza la classe és:

- *HashMap<Item, HashMap <TipusAtribut, HashSet<Atribut> >> tipusAtributsItems;*
- *HashMap<Item, HashMap<Item, Double> > distàncies;*

tipusAtributsItems, per a cada item te organitzats tots els seus atributs segons el tipus d'atribut que siguin aquests, i *distàncies* col·lecció *HashMap<Item, HashMap<Item, Double> >* que per a cada item te emmagatzemades les distàncies a tots els altres items.

Al document d'estructures de dades i algorismes s'explica amb detall com utilitzem aquestes estructures de dades, i com calculem exactament aquestes distàncies.

La classe té dos constructors i els seus getters.

En quant als mètodes públics ens trobem amb *distItem*, *afegirItem* i *eliminarItem*.

1.11.1. afegirItem

Afegeix un item més a la classe, i per tant es calcularan les distàncies també cap a aquest.

Paràmetre: El paràmetre *item* és l'ítem que s'afegeixà

Retorna: -

1.11.2. eliminarItem

Funció que, és l'encarregada d'eliminar un ítem de la classe, i per tant no es calcularan les distàncies d'aquest cap a altres items

Paràmetre: El paràmetre *item* és l'ítem que s'afegeixà

Retorna: -

A més dels mètodes públics, també cal destacar el mètode privat **calculaDistanciesDeItem** que que calcula totes les **distàncies** d'un ítem cap a tots els altres, i afegeix aquestes a la col·lecció **distancies**. Aquesta funció privada, crida a la funció **calculaDistanciaItemItem**, que calcula la distància entre dos ítems.

Però, com que per a calcular la distància entre dos ítems tenim una classe pròpia anomenada **ItemDistItem**, llavors la funció **calculaDistanciaItemItem** el que farà serà calcular la distància entre dos ítems a partir d'una instància de la classe **ItemDistItem**.

1.12. ITEMDISTITEM

ItemDistItem és la classe encarregada de calcular la distància només entre dos items. Per tant, aquesta classe li proporcionarà la distància entre dos items a la classe DistItem.

Pel que fa als atributs privats d'aquesta funció, podem observar que tenim *Item item1* i *Item item2* els quals son els dos items a partir dels quals una instància d'aquesta classe calcularà la distància. També tenim, però, l'atribut *DistItem distItem*. Aquesta referència a la classe DistItem que ha creat la instància de ItemDistItem, en termes d'eficiència ens és útil, ja que podem consultar l'atribut *tipusAtributsItems* de la classe DistItem per calcular la distància, tal com diu a la documentació d'estructures de dades i algorismes. Finalment, també tenim l'atribut *Double distancia*, el qual conté la distància.

Pel que fa als mètodes públics d'aquesta classe, només disposem de la constructora, i del getter de la distància, ja que la constructora mateixa ja crida al mètode privat que calcula la distància.

És important destacar també que, com que la distància entre dos items es calcula comparant els atributs dels items que siguin d'un mateix TipusAtribut, a part del mètode privat principal que calcula la distància entre els dos items, també tenim 4 mètodes privats: *distTAtributBool*, *distTAtributInt*, *distTAtributString* i *distTAtributDouble*. Els quals, segons el tipus d'atributs que contingui un TipusAtribut, calcularan una porció de la distància d'una manera o d'altre.

1.13. DISTUSUARI

DistUsuari és la classe que calcula les distàncies entre diferents Usuaris. Aquesta conté els atributs privats; *uAnalitzat*, de tipus [Usuari](#) que és l'usuari a partir del qual es calcularan totes les distàncies a altres usuaris en la instancia de la classe (DistUsuari) pertaneixent, i *distàncies* col·lecció `HashMap<Usuari, Double>` que és el Map que conté totes les distàncies que hi ha entre l'usuari *UAnalitzat* i tots els altres usuaris.

La classe té dues constructores, una que crea una instància de la classe sense calcular cap distància cap a cap usuari i l'altra que a partir dels paràmetres calcula les distàncies d'un usuari cap a altres.

Les seves funcions públiques són *afegirUsuari*, *eliminarUsuari* i *recalcularTotesDistancies*:

1.13.1. afegirUsuari

Funció que afegeix un altre usuari amb el qual es calcularà la distància cap a l'usuari *UAnalitzat*.

Paràmetre: *usuari* És l'usuari nou amb el qual també es calcularà la distància cap a l'usuari *UAnalitzat*.

Retorna: -

1.13.2. eliminarUsuari

Funció que elimina un usuari d'entre aquells amb els quals els calculava la distància cap a l'usuari *UAnalitzat*.

Paràmetre: *usuari* És l'usuari que s'eliminarà del càlcul de distàncies cap a l'usuari *UAnalitzat*.

Retorna: -

1.13.3. recalcularTotesDistancies

Funció que és la responsable de recalcular totes les distàncies entre l'usuari *UAnalitzat* i tots els altres. Aquesta funció, per tant, és útil quan es creen/modifiquen/eliminen valoracions.

1.14. EXCEPCIONS

Excepcions és la classe de les excepcions del programa, aquesta no té atributs ni getters, ni constructora perquè tots els mètodes són d'Exception.

1.15. HYBRID

Classe que calcula les recomanacions en funció de les classes [Collaborative Filtering](#) i [Content Based Filtering](#).

Conté un [CBFiltering](#), un [COLFiltering](#), k (el nombre de recomanacions que volem) i un ArrayList de les últimes recomanacions calculades.

Té una única constructora i un getter.

El seu únic mètode públic és *calculRecomanacions*.

1.15.1. calculRecomanacions

És un mètode que calcula les recomanacions en funció de les classes [CBFiltering](#) i [COLFiltering](#), en funció del atribut privat *k* treurà el nombre de recomanacions que es demanen i aquest resultat es guardarà a l'ArrayList recomanacions.

Paràmetre: -

Retorna: un ArrayList<Item> de recomanacions

1.16. ITEM

Aquest és la classe que representa un Item. Conté un *id*, unes *valoracions* amb format HashMap<Usuari, Valoracio> i un set d'Atribut anomenat *Atributs*.

Té tres diferents constructors, un buit, un amb tots els atributs privats inicialitzats i un altre amb tots els atributs privats menys les valoracions. També té els seus corresponents getters.

Com a mètodes públics només hi ha *addValoracio* i *deleteValoracio*.

1.16.1. addValoracio

Mètode que afegeix una valoració a *valoracions* si no existeix una valoració amb aquest mateix usuari.

Paràmetre: Valoració *valoració* i Usuari *autor*.

Retorna: -

1.16.2. deleteValoracio

Mètode que elimina una de les valoracions si l'usuari que l'està eliminant n'és creador.

Paràmetre: Usuari *autor*.

Retorna: -

1.17. KMEANS

La classe kMeans es basa en l'algorisme k-Means, un mètode d'agrupament que té com a objectiu la partició d'un conjunt de n elements en k clusters en el qual cada observació pertany al grup més proper a la mitjana del clúster. En aquest cas volem fer clústers d'usuaris (u_1, u_2, \dots, u_n). Volem fer k particions de tal forma que cada usuari estigui amb un grup d'usuaris semblants. Per tal de determinar la proximitat entre dos usuaris diferents posseïm la distància entre cada parell d'usuaris. La distància és la distància euclídea i es calcula a partir de les valoracions de cada usuari a un ítem.

La classe té tres atributs privats; el primer és un atribut de tipus Double final anomenat *maxValue*, un real que representa el valor infinit que es fa servir per definir una distància màxima a partir de la qual es calcula la distància mínima a un centroide, seguidament tenim *clusters* un ArrayList<HashSet<Integer>> on a cada posició hi ha un HashMap amb el id dels usuaris que conformen cada cluster i finalment *distancies*, que és un HashMap de HashMaps de idUsari, distància. Aquestes son afegides per la constructora del k-Means i les calcula la classe DistUsuari.

Té una sola constructora i els seus getters.

L'únic mètode públic que hi ha és *calcularClusters*.

1.17.1. calcularClusters

Calcula els k clusters i introduceix cada usuari al corresponent cluster. Primerament escull k usuaris random per ser els centroides. Després classifica cada usuari al centroide més proper. La distància d'un usuari X a un centroide es calcularà com a la mitjana de cada totes les distàncies de cada usuari que formi part del centroide a l'usuari X.

Paràmetre: k És l'usuari del qual volem trobar tots els seus veïns (usuaris del mateix cluster).

Retorna: - . Actualitza la variable privada *clusters*.

1.18. SLOPEONE

La classe slopeOne es basa en l'algorisme Slope One, un algorisme que consisteix en predir la valoració que donaria l'usuari actiu a un item donat, a partir de les valoracions fetes per altres usuaris. Els altres usuaris soLEN ser usuaris pròxims, és a dir, amb gustos similars i es treuen del k-means.

Aquesta classe té dos atributs privats, el primer és Usuari *usuari*, que és l'usuari actiu al qual es vol fer una predicción sobre un item j, i el segon és HashMap<Integer,Usuari> *usuarisVeins*, que són tots aquells usuaris que es troben en el mateix cluster i per tant comparteixen gustos amb l'usuari actiu. Aquests usuaris es troben amb l'algorisme k-Means.

A part de la seva constructors, la classe té un sol mètode públic anomenat *prediccio*.

1.18.1 prediccio

Predicció és la predicción d'un Usuari "usuari" sobre un Item. Retorna un real que és la puntuació que l'usuari donaria.

Paràmetre: Item j, l'item del qual volem saber la seva predicción per l'usuari actiu.

Retorna: Double, el real que és la puntuació que l'usuari donaria.

1.19. TIPUSATRIBUT

TipusAtribut és la classe que dicta com serà l'atribut que estem llegint, aquesta classe ens permetrà classificar el si és calculable, el seu tipus i algunes propietats que se li assignen.

La classe té els atributs privats següents:

- *nom* : identifica el TipusAtribut.
- *calculable*: determina si els atributs relacionats amb el TipusAtribut s'utilitzaran a l'hora de calcular les recomanacions per a l'usuari.

- *atributs*: representen totes les instàncies d'Atribut relacionades amb el TipusAtribut.
- *max*: s'utilitza per saber quina dada, dels atributs dels TipusAtribut (Int o Double), és la més gran.
- *min*: s'utilitza per saber quina dada, dels atributs dels TipusAtribut (Int o Double), és la més petita.
- *tipusDada*: determina de quin tipus és la dada de l'atribut del TipusAtribut, és a dir: Integer, Double, Boolean, o String.
- *MaxMinInicalitzats*: indica si els valors dels atributs *max*, *min* estan inicialitzats.
- *strategyTipus*: fa referència a una classe *StrategyTipus* per a la implementació del mètode *crearAtribut*.

1.19.1. afegirAtribut/eliminaAtribut

Aquest dos mètodes s'encarreguen de gestionar el set d'atributs de la classe. Eliminen iafegeixen al set el atribut que passa com a paràmetre, i a més actualitzen els valors de max i min en el cas de que aquell atribut fes referència a aquests valors.

1.19.2. crearAtribut

Aquest mètode crida al mètode *crearAtribut* de la instància atribut *strategyTipus*. El tipus d'aquesta instància es determina en funció del valor de *tipusDada* (l'assignació es fa a la constructora). Bàsicament segons l'estratègia implementada, crearà una subclasse d'atribut diferent.

1.20. UACTIU

UActiu és classe filla d'[Usuari](#), i és l'UsuariActiu, el que està fent ús del programa i sobre el qual es calcularan els algorismes quan entri en funcionament, el que està “logged in” en aquell moment.

Aquesta classe té un atribut privat *nom* de tipus String que representa el nom de l'usuari actiu, un String anomenat *password* que guarda la contrasenya d'accés d'aquest mateix usuari, i un Set d'ítems *itemsCreates* que representen tots els ítems creats per l'usuari actiu.

La classe té una sola constructora, els seus getters i setters.

1.20.1. eliminarItemCreat/afegirItemCreat

Aquests dos mètodes s'encarreguen de la gestió de l'atribut *itemsCreates*.

1.21. USUARI

Usuari és la classe que representa un Usuari qualsevol i la classe mare d'[UActiu](#). Aquesta classe té l'atribut privat Integer *id*, que és el que representa l'usuari, i *itemsValoracions*, un Map que conté com a clau tots aquells ids de ítems que l'usuari amb identificador id a valorat, i com a valor d'aquestes claus, el id de la valoració que correspon.

La classe té una sola constructora, els seus getters i setters.

1.21.1. isActiu

Retorna si l'usuari és de tipus UActiu (true) o no.

1.21.2. haValorat

Retorna si l'usuari ha valorat l'ítem passat per paràmetre (true) o no.

1.21.3. addItemValoracio/eliminarValoracio

Aquests dos mètodes s'encarreguen de la gestió de l'atribut *itemsValoracions*.

1.22. VALORACIO

Classe que representa una valoració. Conté l'atribut privat Usuari *usuari*, el que ha escrit la valoració, un altre atribut privat Double *puntuacio*, que representa la puntuació assignada, un String *descripcio*, i *item*, l'Item al que correspon la valoració.

La classe té una sola constructora, els seus getters i setters. No té mètodes.

1.24. PairComparator

Aquesta classe és una extensió de la classe Comparator. S'utilitza per ordenar maps amb claus de tipus Pair<Integer, String>.

1.24.1. compare

Aquest mètode retorna la comparació entre els valors *first()* dels dos *Pair*.

1.25. StrategyTipus

Representa interfície, amb la qual apliquem el patró estrategia pel mètode *crearAtribut*.

1.26 StrategyString

Classe filla de l'interfície *StrategyTipus*.

1.26.1. crearAtribut

Crea una instància de *AtrString* amb *atributString* = Paraula.

Afegeix el nou *AtrString* al Set d'atributs del *TipusAtribut* corresponent.

Afegeix el nou *AtrString* al Set *atributsItem*.

1.27. StrategyBool

Classe filla de l'interfície *StrategyTipus*.

1.27.1. crearAtribut

Crea una instància de *AtrBool* amb *atributBool* = Paraula.

Afegeix el nou *AtrBool* al Set d'atributs del *TipusAtribut* corresponent.

Afegeix el nou *AtrBool* al Set *atributsItem*.

1.28. StrategyDouble

Classe filla de l'interfície *StrategyTipus*

1.28.1. crearAtribut

Crea una instància de *AtrDouble* amb *atributDouble* = Paraula.

Afegeix el nou *AtrDouble* al Set d'atributs del *TipusAtribut* corresponent.

Afegeix el nou *AtrDouble* al Set *atributsItem*.

1.29. StrategyInt

Classe filla de l'interfície *StrategyTipus*

1.29.1. crearAtribut

Crea una instància de *AtrInt* amb *atributInt* = Paraula.

Afegeix el nou *AtrInt* al Set d'atributs del *TipusAtribut* corresponent.

Afegeix el nou *AtrInt* al Set *atributsItem*.

1.30. CONTROLADORS DOMINI

1.30.1. CtrlDomini

Aquesta classe és un singleton, ja que només volem una instància d'aquesta.

S'encarrega de totes les operacions on estan implicades les classes de la capa de domini. Delega cada operació al controlador de la classe específica. Gestiona el pas de dades al CtrlPersistencia i al CtrlPresentacio.

Com a atributs té un *UActiu* el qual corresponent a l'usuari que ha iniciat sessió, i totes les instàncies dels controladors generats per les relacions de composició.

1.3.0.1.1. inicialitzarCtrlDomini

Inicialitza tots els controladors adjacents al CtrlDomini (per la relació de composició) i el controlador de persistència. Per això es crida a aquest mètode a la constructora del CtrlDomini.

1.3.0.1.2. validarPath

Crida a l'operació *validarPath* del CtrlPersistencia per veure si el path que li passa el CtrlPresentacio és vàlid, és a dir, que conté els arxius de base de dades: *items.csv* i *ratings.db.csv*.

1.3.0.1.3. setTipusAtributs

Aquest mètode fa diferents crides a mètodes del CtrlPersistencia, i d'altres controladors de la capa de domini.

Aquest mètode s'encarrega de carregar tota la base de dades (*items*, *usuaris*, *valoracions*, *tipusAtributs*) a partir de les dades proporcionades pel CtrlPersistencia.

A més, per determinar els *tipusDada* i *calculable* de cada *TipusAtribut*, utilitza la configuració que li ha passat com a paràmetre el CtrlPresentacio.

1.3.0.1.4. canvisTipusAtributArxiuItems

Aquest mètode fa diferents crides a mètodes del CtrlPersistencia, i d'altres controladors de la capa de domini.

Aquest mètode s'encarrega de comprovar si existeix l'arxiu de configuració dels *TipusAtribut* que crea la propia aplicació, mitjançant la crida al mètode *canvisTipusAtributArxiuItems* del CtrlPersistencia.

A més també s'encarrega tota la base de dades (*items*, *usuaris*, *valoracions*, *tipusAtributs*) a partir de les dades proporcionades pel CtrlPersistencia.

La diferència entre aquest mètode i el mètode *setTipusAtributs* és que aquest obté dades extres del CtrlPersistencia, ja que té en compte tots els altres usuaris que han

iniciat sesió: les seves valoracions i items creats. A més la configuració de cada *TipusAtribut* l'obté també de la base de dades, i no del CtrlPresentacio.

1.3.0.1.5. carregarRecomanacions

Aquest mètode carrega totes les dades proporcionades per *dataRecomanacions* en el map *itemsRecomanatsPerUsuari*.

1.3.0.1.6. carregarUsuarisActius

Aquest mètode carrega tots els usuaris actius proporcionats per *dataUsuarisActius* en el map *usuarisActius*.

1.3.0.1.7. getPosicioId

Aquest mètode retorna la posició (en les columnes de l'arxiu *items.csv*) on està el nom del *TipusAtribut* del qual utilitzem per identificar els ítems.

1.3.0.1.8. afegirValoracionsUA

Aquest mètode afegeix totes les valoracions proporcionades per *dataRatingsUA* al map *valoracions*.

1.3.0.1.9. registraUsuari

Assigna el valor a l'atribut *usuariActiu*. A més crida al ctrlDUsuari i CtrlPersistencia perquè afegeixin un usuari a la seva base de dades.

1.3.0.1.10. iniciarSesio

Assigna el valor a l'atribut *usuariActiu*.

1.3.0.1.11. getNomTipusAtributs

Retorna un HashSet amb els noms de tots els TipusAtribut, cridant al mètode *llegeixLlíniesArxiu*, del CtrlPersistencia , passant-li com a paràmetre el path.

1.3.0.1.12. getTipusAtributs

Retorna un map amb dades dels TipusAtribut, cridant el mètode *getTipusAtributs* del CtrlDConfiguracio.

1.3.0.1.13. setTipusAtributId

Aquest mètode crida al mètode *setTipusAtributId*, del CtrlDConfiguracio, perquè es guardi el valor del nom del *TipusAtribut* del qual diferenciarem tots els ítems. és a dir, el id.

1.3.0.1.14. almenysUnaValoracioFeta

Retorna si l'usuari actiu té almenys una valoració feta, cridant al mètode *almenysUnaValoracioFeta* de CtrlDValoracio.

1.3.0.1.15. nomsItemsRecomanats

Retorna un HashSet amb els noms de tots els ítems recomanats de l'usuari actiu.

1.3.0.1.16. setTipusAtributTitol

Aquest mètode crida al mètode *assignarTitol*, del CtrlDConfiguracio, perquè es guardi el valor del nom del *TipusAtribut* del qual diferenciarem els títols de tots els ítems-

1.3.0.1.17. existeixItem

Retorna si existeix l'ítem a la base de dades, cridant al mètode *existeixItem* de CtrlDItem.

1.3.0.1.18. getIdItem

Retorna el id (*Integer*) de l'ítem amb el títol passat com a paràmetre, cridant al mètode *getIdItem* de CtrlDItem.

1.3.0.1.19. getValorsTipusAtributDeItem

Retorna un *HashMap*, on la clau és el nom del *TipusAtribut* i el valor és la dada de l'atribut, cridant el mètode *getValorsTipusAtributDeItem* de CtrlDItem. Per tots els atributs de l'ítem amb el títol passat com a paràmetre.

1.3.0.1.20. creaValoracio

Aquest mètode comunica als CtrlDValoracio, CtrlUsuari i CtrlPersistencia perquè afageixin una valoració a l'ítem feta per l'usuari actiu.

1.3.0.1.21. eliminaValoracio

Aquest mètode comunica als CtrlDValoracio, CtrlUsuari i CtrlPersistencia perquè eliminin una valoració a l'ítem feta per l'usuari actiu.

1.3.0.1.22. lecturaHeader

Aquest mètode interpreta les dades del ArrayList *header*, amb l'ajuda del map *configUser*, per determinar la inicialització del map *tipusAtribut*. *header* representa la primera línia de l'arxiu *items.csv*.

1.3.0.1.23. inicialitzarItems

Aquest mètode carrega el map *items* amb les dades proporcionades per l'ArrayList *atributsItems*.

1.3.0.1.24. mapeigRatings

Aquest mètode carrega el *mapColumnes* amb les dades proporcionades per l'*ArrayList* *dadesArxiu*. *mapColumnes* representa el primer *ArrayList* de *dadesArxiu*.

Retorna un *double* que representa el rating màxim de totes les dades de *dadesArxiu*.

1.3.0.1.25. inicialitzarUsuarisValoracions

Aquest mètode carrega els maps *usuaris*, *valoracions* i *items* amb les dades proporcionades per l'*ArrayList* *dataArxiu*.

1.3.0.1.26. comptarQuotes

Retorna quantes vegades el *String paraula* té el caràcter ' \''.

1.3.0.1.27. consultaTipusTipusAtributs

Retorna un *HashMap*, on la clau és el nom del *TipusAtribut* i el valor és el seu *tipusDada*, cridant el mètode *getTipusTipusAtributs* de *CtrlDConfiguracio*.

1.3.0.1.28. consultaValoracionsUsuari

Retorna un *HashMap* amb les dades de cada valoració feta per l'usuari actiu.

1.3.0.1.29. eliminarUsuariActiu

Comunica als *CtrlDUsuari*, *CtrlDValoracio*, *CtrlPersistencia* perquè eliminin l'usuari actiu i les seves relacions de les seves bases de dades.

1.3.0.1.30. eliminarItem

Comunica als *CtrlDUsuari*, *CtrlDValoracio*, *CtrlDItem*, *CtrlPersistencia* perquè eliminin l'ítem i les seves relacions de les seves bases de dades.

1.3.0.1.31. creaItem

Comunica als *CtrlDUsuari*, *CtrlDItem*, *CtrlPersistencia* perquè afegeixin l'ítem a les seves bases de dades.

1.30.2. CtrlDominiConfiguracio

Classe singleton. S'encarrega de la gestió de la configuració dels *TipusAtribut*.

Com a atributs té un *String nomId* que representa el nom del *TipusAtribut* que representa el id dels ítems, un *String nomTitol* que representa el nom del *TipusAtribut* que representa el titol dels ítems, un *String path* que representa el path del directori on estan tots els arxius, i un *Map<Pair<Integer, String>, TipusAtribut> tipusAtributs* que consiteix en tots els *TipusAtributs* indexats per la columna que tenen a l'arxiu *items.csv* i el seu nom.

1.3.0.2.1. getDadesPerFitxer

Retorna un *Map* on la clau és el nom del *TipusAtribut* i el valor és un *Pair* on el *first()* és el *tipusDada* i el *second()* és el *calculable*, per cada *TipusAtribut* del *tipusAtribut*.

1.3.0.2.2. getTipusTipusAtributs

Retorna un *HashMap* on la clau és el *nom* del *TipusAtribut* i el valor és el *tipusDada*, per cada *TipusAtribut* de *tipusAtribut*.

1.30.3. CtrlDominiItem

Classe singleton. S'encarrega de la gestió dels items.

Com a atributs té un *Integer idMax* que representa el id amb el valor més gran de tots els ítems del *Map items*, i un *Map<Integer, Item> items* que consta de tots els ítems.

1.3.0.3.1. getTitolsItems

Retorna un *HashSet* amb tots els títols dels ítems amb els *id* del *HashSet itemsRecomanats* passat com a paràmetre.

1.3.0.3.2. existeixItem

Retorna si existeix l'ítem amb el títol *nom* al *Map items*.

1.3.0.3.3. getIdItem

Retorna el *id (Integer)* de l'ítem amb títol *titol* al *Map items*.

1.3.0.3.4. getValorsTipusAtributDeItem

Retorna un *HashMap* on la clau és el *nom* del *TipusAtribut* i el valor és la dada de *l'Atribut*, per tots els *Atributs* que conté l'ítem amb títol *titol* al *Map items*.

1.3.0.3.5. eliminarItem/afegirItem

Aquests dos mètodes s'encarreguen de la gestió del *Map items*.

1.30.4. CtrlDominiUsuari

Classe singleton. S'encarrega de la gestió d'usuaris.

Com a atributs té un *Integer idMax* que representa el id més gran de tots els usuaris del *Map usuaris*, i un *Map<Integer, Usuari> usuaris* que consta de tots els usuaris.

1.3.0.4.1. registraUsuari

Comprova si l'usuari amb nom *nom* i contrasenya *contrasenya* existeix a *usuaris* i en cas de que no, l'afegeix.

1.3.0.4.2. iniciarSesio

Retorna un UActiu en cas de que l'usuari amb nom *nom* i contrasenya *contrasenya* existeix a *usuaris*.

1.3.0.4.3. afegirValoracio/eliminarValoracio

Aquests dos mètodes s'encarreguen de la gestió de valoracions fetes per cada usuari.

1.3.0.4.4. checkCreador

Retorna si l'usuari amb id *id* és el creador de l'ítem *item*.

1.3.0.4.5. eliminarItem/crearItem

Aquests dos mètodes s'encarreguen de la gestió d'ítems creats per cada usuari.

1.30.5. CtrlDominiValoracio

Classe singleton. S'encarrega de la gestió de les valoracions.

Com a atributs té un *Map<Pair<Item, Usuari>, Valoracio> valoracions* que representen totes les valoracions.

1.3.0.5.1. afegirValoracio/eliminarValoracio

Aquests dos mètodes s'encarreguen de la gestió de valoracions de *valoracions*.

1.3.0.5.2. almenysUnaValoracioFeta

Retorna si existeix una valoració feta de l'usuari *usuariActiu*.

1.3.0.5.3. eliminarUA

Elimina totes les valoracions fetes per l'usuari *usuariActiu* de *valoracions*.

1.3.0.5.4. eliminarItem

Elimina totes les valoracions de l'ítem *item* de *valoracions*.

1.30.6. CtrlDominiValoracio

Classe singleton. S'encarrega de la creació de les recomanacions, la gestió d'aquestes, i per tant, el refresh de les recomanacions quan calgui.

Aquesta classe disposa d'un atribut privat en format Set, el qual enmagatzema els items que son recomanacions de l'usuari que té la sessió iniciada.

Aquesta classe disposa de 5 principals funcions públiques.

La primera, és un simple getter, que retorna l'atribut privat que és el set de items recomanats. Totes les altres, s'encarreguen d'actualitzar la llista de recomanacions.

Dos d'aquests s'encarregaràn d'actualitzar la llista de recomanacions quan s'hagi eliminat o afegit una valoració (una es cridarà quan s'hagi afegit una valoració, i l'altre quan s'hagi eliminat).

I les dues últimes es cridaràn quan s'elimini o s'afegeixi un ítem, ja que quan això passi també s'haurà de refrescar la llista que conté les recomanacions.

2. PERSISTÈNCIA

2.1 CONTROLADORS PERSISTÈNCIA

2.1.1 CtrlPersistencia

Aquesta classe és un singleton, ja que només volem una instància d'aquesta.

S'encarrega de totes les operacions on estan implicades les classes de la capa de persistència. Delega cada operació al controlador de la classe específica.

Gestiona la persistència de les dades, és a dir la seva lectura i el gravat d'aquestes en els diferents arxius.

2.1.1.1. inicialitzarCtrlPersistencia

Inicialitza tots els controladors adjacents al CtrlPersistencia (per la relació de composició). Per això es crida a aquest mètode a la constructora del CtrlPersistencia.

2.1.1.2. validaPath

Valida el path cridant al mètode *comprovacioFitxers* del CtrlPersistenciaConfiguracio.

2.1.1.3. inicialitzarControladors

Crida el mètode de *inicilaitzarControlador* per a cada controlador de persistència creat en el mètode inicialitzarCtrlPersistencia.

2.1.1.4. canvisTipusAtributArxiuItem

Retorna true si ha hagut algun canvi a l'arxiu items.csv.

2.1.1.5. llegeixLliniesArxiu

Retorna les 3 línies de l'arxiu amb path = path, per identificar les paraules en una línia s'utilitza la funció split() que separa tots els elements entre el separador.

2.1.1.6. llegeixArxiu

Retorna totes les línies de l'arxiu amb path = path, per identificar les paraules en una línia s'utilitza la funció split() que separa tots els elements entre el separador.

2.1.1.7. carregarUsuaris

Crida al mètode de *afegirUsuaris* del CtrlPersistenciaUsuari.

2.1.1.8. afegirValoracio

Crida al mètode *afegirValoracio* del CtrlPersistenciaValoracio, el qual afegeix una valoració de un item a un usuari.

2.1.1.9. afegirUsuariActiu

Crida al mètode *afegirUsuariActiu* del CtrlPersistenciaUsuari, on s'afegeix un nou usuari actiu.

2.1.1.10. eliminarValoracio

Crida al mètode *eliminarValoracio* del CtrlPersistenciaValoracio, on s'elimina una valoració feta per un usuari a un ítem.

2.1.1.11. getTipusAtribut

Retorna el resultat de la crida al mètode *getTipusAtributs* de CtrlPConfiguracio.

2.1.1.12. getNomId

Retorna el resultat de la crida al mètode *getNomId* de CtrlPConfiguracio.

2.1.1.13. getNomTitol

Retorna el resultat de la crida al mètode *getNomTitol* de CtrlPConfiguracio.

2.1.1.14. eliminarUA

Comunica als CtrlPUusuari, CtrlPRecomanacio, CtrlPValoracio perque eliminin totes les relacions que tingui l'usuari amb id *idUser* a les seves bases de dades.

2.1.1.15. setHeaderVal

Crida el mètode detectarHeader de CtrlPValoracio.

2.1.1.16. eliminarItem

Comunica als CtrlPItem, CtrlPRecomanacio, CtrlPValoracio perque eliminin totes les relacions que tingui l'ítem amb id *idItem* a les seves bases de dades.

2.1.1.17. afegirItem

Crida el mètode afegirItem de CtrlPItem.

2.1.2. CtrlPersistenciaConfiguracio

Classe singleton. S'encarrega de la lectura i escriptura de l'arxiu configuracio.txt

2.1.2.1. comprovarCanvisItems

Retorna si hi ha hagut canvis a l'arxiu items.csv.

En cas de que no existeixi l'arxiu configuracio.txt retorna true.

En cas de que existeixi l'arxiu configuracio.txt, comprova que les dades que hi ha siguin iguals que la primera línia de l'arxiu items.csv (representada en *l'ArrayList nomTipusAtributs*).

2.1.2.2. comprovacioFitxers

Comprova que els fitxers items.csv i ratings.db.csv existeixin en el path passat com a paràmetre.

2.1.2.3. getNomId

Retorna un *String* que correspon a la primera línia de l'arxiu configuracio.txt, que representa el nom del TipusAtribut que representa els id dels items.

2.1.2.4. crearFitxerConfig

Crea un fitxer.txt anomenat “configuracio” amb les línies escrites:

Primera línia: *nomId*

Segona línia: *nomTitol*

Tercera línia: “*nom,tipusDada,calculable*”

2.1.2.5. afegirTipusAtribut

Escriu a partir de l’última línia, per cada *TipusAtribut* (de *tipusAtributs*), *nom*, *tipusDada*, *calculable*.

2.1.2.6. getTipusAtributs

Retorna un *Map* on la clau és el nom del *TipusAtribut* i el valor es un *Pair* on el *first()* és *tipusDada* i el *second()* és *calculable*.

2.1.2.7. getNomTitol

Retorna un *String* que correspon a la segona línia de l'arxiu configuracio.txt, que representa el nom del TipusAtribut que representa els titol dels items.

2.1.3. CtrlPersistenciaItem

Classe singleton. S’encarrega de la lectura i escriptura de l’arxiu items.csv.

2.1.3.1. eliminarItem/afegirItem

Aquests dos mètodes s’encarreguen d’afegir i eliminar items del fitxer items.csv.

2.1.4. CtrlPersistenciaRecomanacio

Classe singleton. S’encarrega de la lectura i escriptura de l’arxiu recomanacions.csv.

2.1.4.1. crearFitxerRecomanacions

Crea un fitxer.csv anomenat “recomanacions” amb la primera línia escrita: “*idUser,i,idItems*”.

2.1.4.2. eliminarRecomanacionsUsuari/afegirRecomanacionsUsuari

S’encarreguen de la gestió de conjunts de recomanacions per usuaris.

2.1.4.3. eliminarRecomanacioItem

Elimina una recomanació identificada per *idItem*, de tots els conjunts de recomanacions de tots els usuaris.

2.1.4.3. getRecomanacionsUsuari

Retorna un *ArrayList* amb tots els id de les recomanacions de l'usuari actiu.

2.1.5. CtrlPersistenciaUsuari

Classe singleton. S'encarrega de la lectura i escriptura de l'arxiu usuarisActius.csv.

2.1.5.1. crearFitxerUsuarisActius

Crea un fitxer.csv anomenat “usuarisActius” amb la primera línia escrita: “*idUser,nom,contrasenya*”.

2.1.5.2. afegirUsuariActiu/eliminarUsuariActiu

Aquests dos mètodes s'encarreguen d'afegir i eliminar usuaris segons els seus id.

2.1.6. CtrlPersistenciaValoracio

Classe singleton. S'encarrega principalment de la lectura i escriptura de l'arxiu valoracionsUA.csv.

Com a atribut té un *final Map<Integer, String> header* que representa la primera línia de l'arxiu.

2.1.6.1. crearFitxerValoracionsUA

Crea un fitxer.csv anomenat “valoracionsUA” amb la primera línia on consten els elements del *header*.

2.1.6.2. detectarHeader

Llegeix la primera línia de l'arxiu ratings.db.csv per carregar les dades del *header*.

2.1.6.3. afegirValoracio

Escríu una valoració (*userId*, *rating*, *comentari*, *itemId*) a l'última línia de l'arxiu valoracionsUA.csv.

2.1.6.4. eliminarValoracio

Esborra les valoracions de l'arxiu en funció del *tipus*.

- *userId*: esborra les valoracions fetes per *idUser*.
- *itemId*: esborra les valoracions fetes a l'ítem *idItem*.
- *valoracio*: esborra la valoració identificada per *idUser*, *idItem*.

3. PRESENTACIÓ

3.1 CtrlPresentacio

CtrlPresentacio és l'únic controlador de la capa de presentació. Per tant, aquesta classe controla totes les classes de la capa de presentació, i és l'única que necessita crear l'arxiu Main.

Per tant, com que la capa de presentació és l'encarregada de la interfície gràfica, aquesta és l'encarregada de decidir i deixar decidir a l'usuari les accions que realitzarà el programa en el seu transcurs, de forma que és aquest controlador el que haurà de crear una instància del controlador de domini, i comunicar-se amb aquesta, ja sigui per enviar accions, o com per rebre dades, per exemple imprimirlers.

Així doncs, aquesta classe controlador contindrà totes les funcions amb que la capa de presentació és comunicarà amb la capa de domini.

Però, com que totes aquestes funcions l'única cosa que fan és cridar a una funció del controlador de domini, per estalviar-nos escriure la mateixa explicació dues vegades, explicarem aquestes funcions a l'apartat del controlador de domini.

Tot i això, hem de destacar que la constructora d'aquest controlador només ha de crear una instància de la vista principal de la interfície gràfica, és a dir una instància de la classe VistaPrincipal.

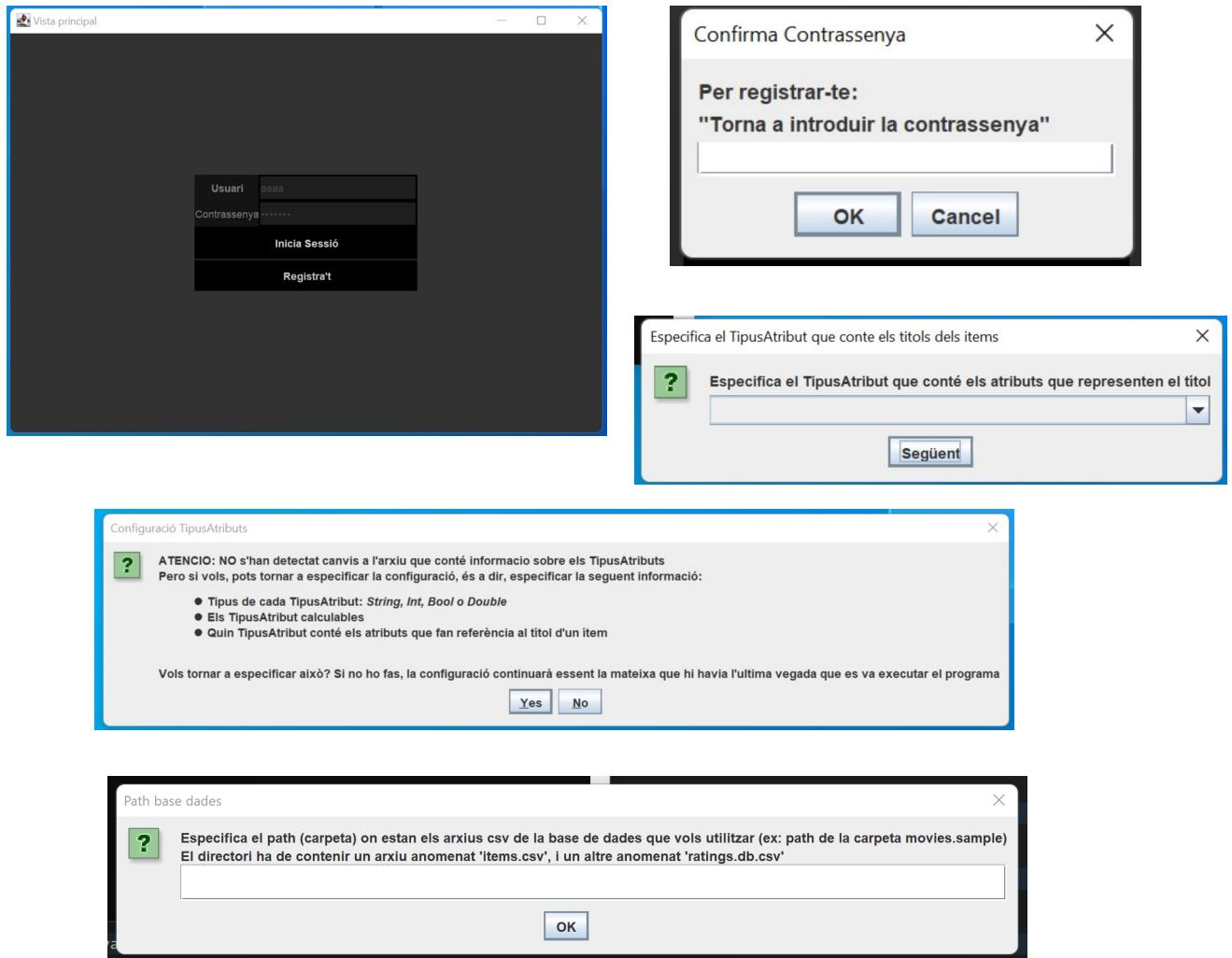
3.2 VistaPrincipal

Aquesta classe és l'encarregada de la vista principal de la interfície gràfica. És a dir, és l'encarregada de mostrar per pantalla tots aquells panells / elements els quals formen part de l'inici del programa, i per tant de tots aquells que hi son presents en algun moment fins que s'inicia una sessió d'un usuari.

Aquests elements comprenen la construcció de la vista del panell que s'utilitza per introduir les dades d'inici de sessió i registe, i popups com: confirmació de la contrassenya en cas de registre, especificació del path de la carpeta on hi ha la base de dades (fitxers csv externs) amb

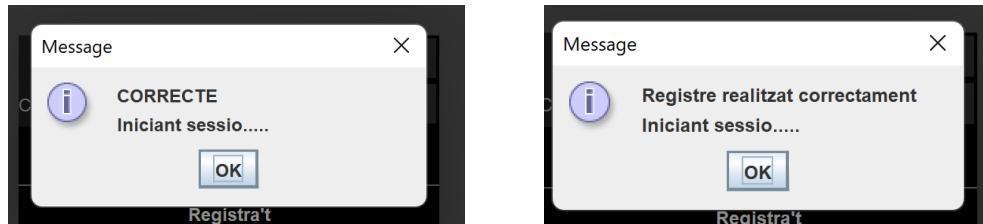
els quals es vol treballar, especificació dels tipus als TipusAtributs d'una base de dades a la qual s'han realitzat canvis, etc...

Alguns d'aquests elements, visualment són així:

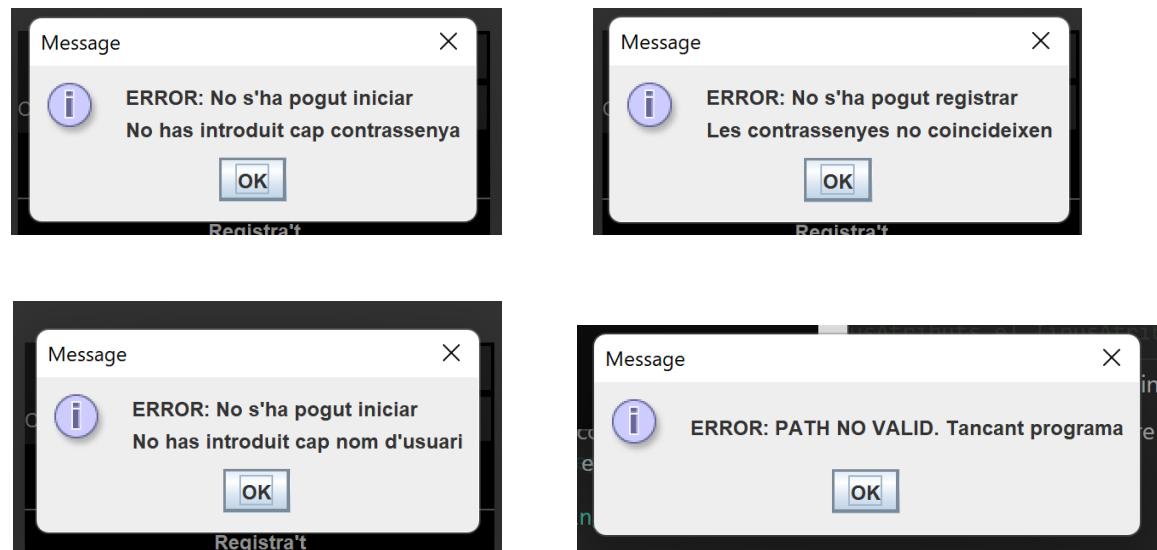


Per tant, pel que fa a la comunicació d'aquesta classe amb el controlador de domini, aquesta classe cridarà a mètodes del controlador de presentació com: **especificaPath**, **registraUsuari**, **iniciaSesio**, **canvisTipusAtributsArxiuItems** (per comprovar si s'han fet canvis en els TipusAtribut a l'arxiu d'ítems d'una base de dades), **getNomTipusAtributs** (per mostrar per pantalla els nom de cada TipusAtribut perquè l'usuari pugui especificar el tipus de cada un d'aquests en cas que s'hagin produït canvis, o que l'usuari opcionalment ho vulgui fer), i **setTipusAtributId**, **setTipusAtributTitol** i **setTipusATipusAtributs** per especificar aquests.

Un cop s'hagi iniciat sessió, o s'hagi registrat un usuari (ja que el registre fa l'inici de sessió automàticament), la classe crearà una instància de VistaSesioIniciada. Per tant, això es produirà després d'un dels següents missatges:



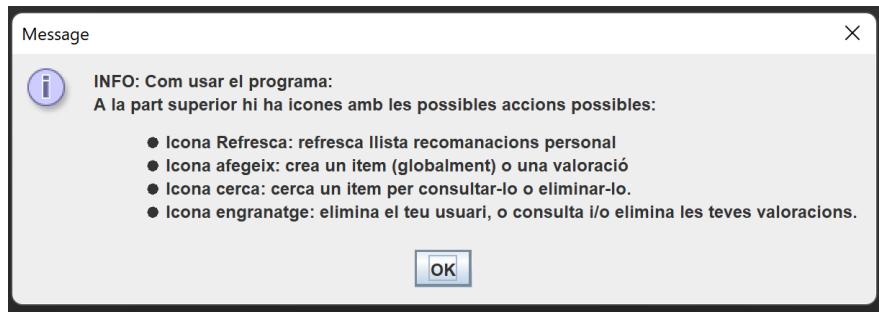
Cal destacar, també, que totes les excepcions seràn tractades com popups de la interfície gràfica. Algunes d'aquestes excepcions, a la classe VistaPrincipal, son:



3.3 VistaSesioIniciada

Una vegada haguerem iniciat sessió, aquesta classe serà l'encarregada de la interfície gràfica durant la resta de l'execució del programa. Aquesta classe, per tant, serà l'encarregada de mostrar i controlar tots aquells elements i panells que permeten que l'usuari faci aquelles accions que vulgui un cop hagi iniciat sessió, i per tant controlar les classes panellAdministraValoracions, panellBusca, panellRecomanacio.

El primer que realitzarà la classe VistaSesioIniciada, és mostrar-nos una guia de les accions que podem fer:

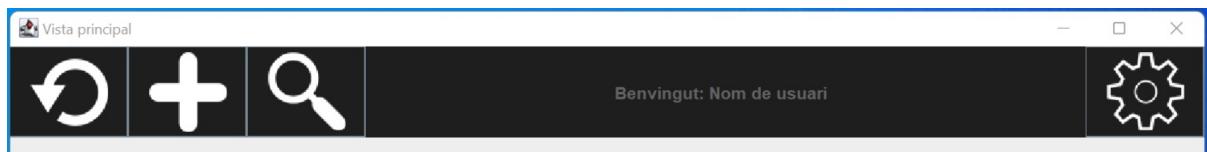


Tal com s'ha explicat en el document de casos d'ús d'aquesta entrega, totes les accions que pot realitzar l'usuari un cop ha iniciat la sessió, es poden agrupar en 4 grups. Llavors, el panell principal d'aquesta classe, sempre mostrarà per pantalla un panell superior, amb 4 icones que representin aquests 4 grups. Llavors, per realitzar alguna de les accions d'algun d'aquests grups, només caldrà clicar a la icona.

Aquests 4 grups son:

- Icôna **Refresca**: Per veure i consultar els ítems recomanats.
- Icôna **afegeix**: Per crear una valoració o un ítem.
- Icôna **cerca**: Per consultar un ítem, o/i eliminar-lo.
- Icôna **engranatge**: Per eliminar el propi usuari, o consultar o/i eliminar valoracions del propi usuari.

Per tant, quan s'hagi iniciat una sessió, o registrat un usuari, aquest panell superior sempre serà visible, i tindrà una aparença similar a aquesta:

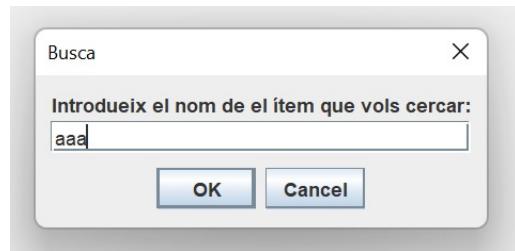


El fet de fer clic a algun d'aquests botons, farà, o que el panell inferior del frame d'aquesta classe canviï i mostri les possibles accions de la icona que a clicat, o que directament aquestes accions es mostrin en forma de popups.

Per tal d'evitar una gran complexitat de la classe VistaSesioIniciada, hem decidit que per a aquells botons en que les seves accions es mostrin en el panell inferior, i per tant canviïn

aquest, que programarem aquests panells en classes diferents, les quals seran creades per VistaSesioIniciada quan es vulgui. Això és el cas de:

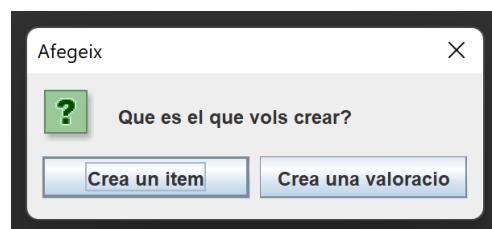
- El botó icona Recomanacions.
- El botó icona Busca. Tot i que aquí abans mostrarem un popup amb el nom de l'ítem que es vol buscar:



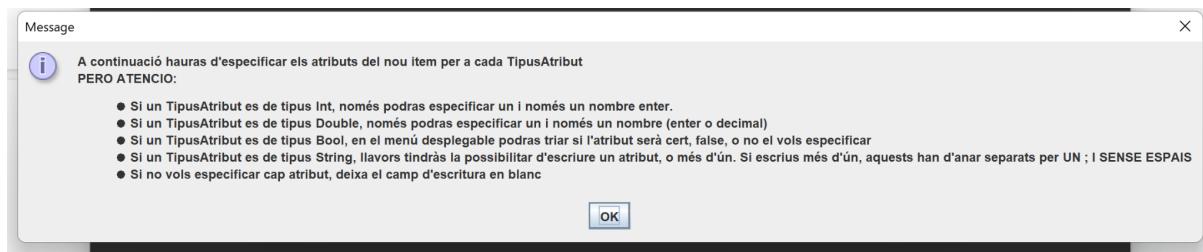
- El botó icona engranatge. Tot i que aquí, com que no necessitem cap panell adicional per eliminar l'usuari, primer preguntarem si el que es vol es eliminar l'usuari o consultar / eliminar propies valoracions, mitjançant popups:



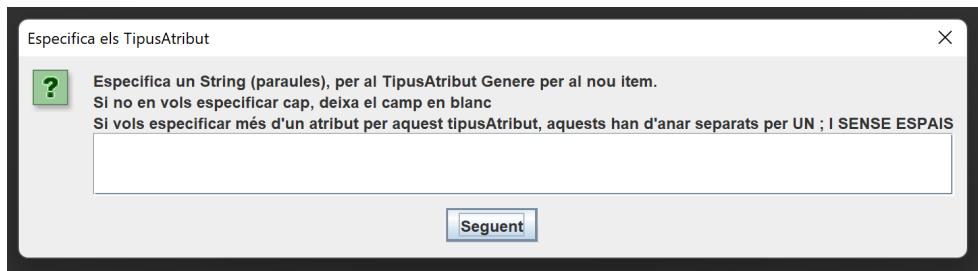
Pel que fa al botó icona afegir, aquest s'ha implementat totalment amb popups. Per tant, quan li donem al botó, se'ns mostrarà un popup com el següent:



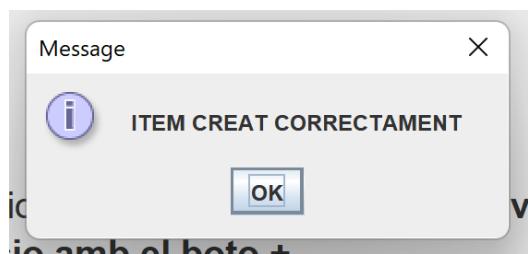
En cas que vulguem crear un ítem, ens apareixerà primer un missatge sobre el que haurem de fer per crear l'ítem:



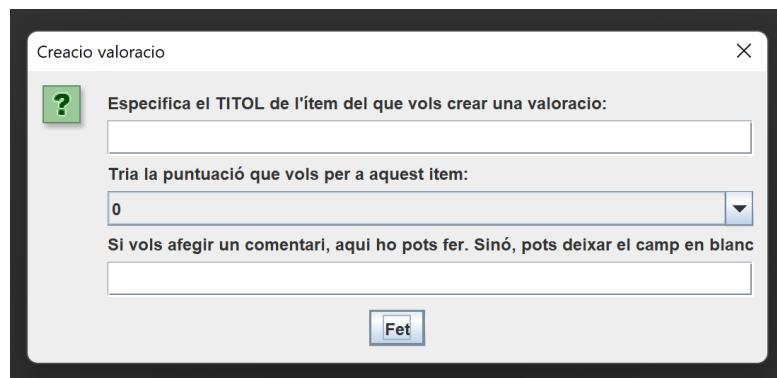
I, a continuació, haurem d'especificar els atributs per a cada TipusAtribut, per a l'ítem. Per exemple, per a un TipusAtribut:



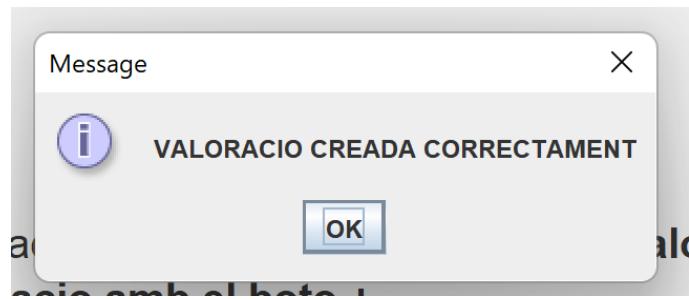
En cas que l'ítem s'hagi creat correctament:



I en el cas, que volguem crear una valoració, VistaSesioIniciada ens demanarà el titol de l'ítem, que seleccionem una puntuació (de 0 a 5) i un comentari, el qual és opcional:



En cas que la valoració s'hagi creat correctament:



Cal destacar, que, en aquest panell, de la mateixa forma que en VistaPrincipal, totes les excepcions seràn tractades com popups de la interfície gràfica. Algunes d'aquestes excepcions, a la classe VistaSesioIniciada , son:



Pel que fa a la comunicació d'aquesta classe amb el controlador de domini, aquesta classe cridrà a mètodes del controlador de presentació com: **consultaTipusTipusAtributs** (per mostrar a l'usuari els noms dels TipusAtributs perquè quan aquest crei un ítem pugui establir cap/un/multiples atributs per TipusAtribut), **creaItem** (perquè el controlador de domini crei un ítem), **creaValoracio** (perquè el controlador de domini crei una valoració), **eliminaUsuari** (podem eliminar l'usuari directament amb el popup principal de el botó engranatge).

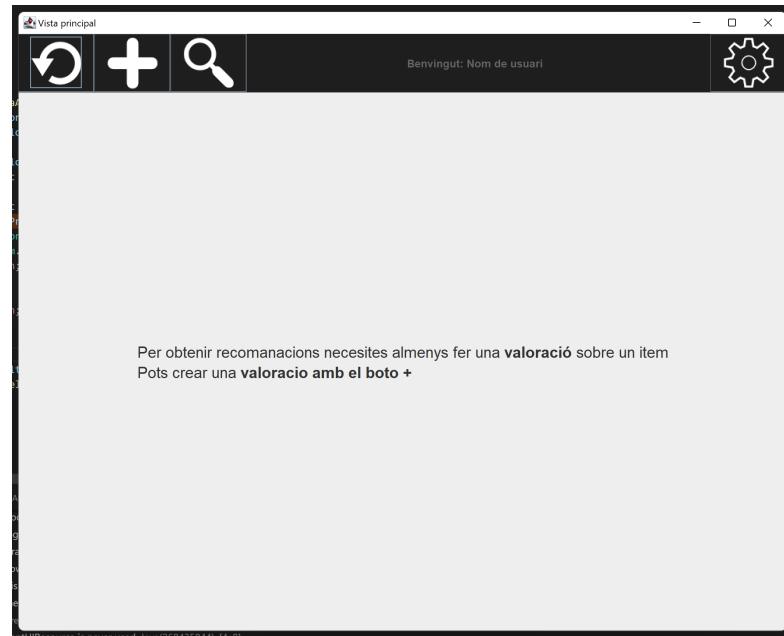
3.4 panellRecomanacio

Aquesta és la classe encarregada de mostrar el panell amb informació de les recomanacions de l'usuari que ha iniciat la sessió.

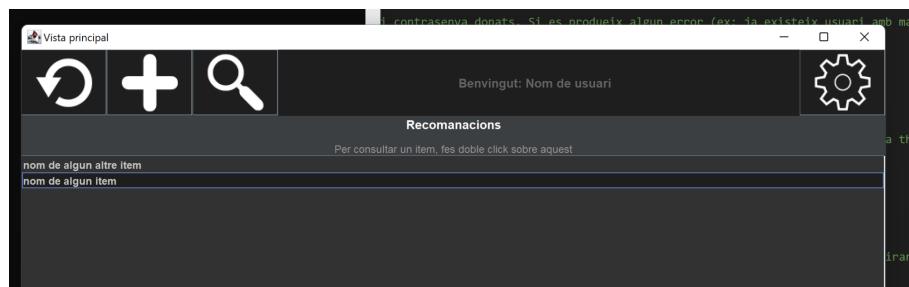
Aquest panell té dos comportaments:

- En el cas que l'usuari que té sessió iniciada no tingui cap valoració, com que realitzar recomanacions a un usuari que no té recomanacions portaria a recomanacions de

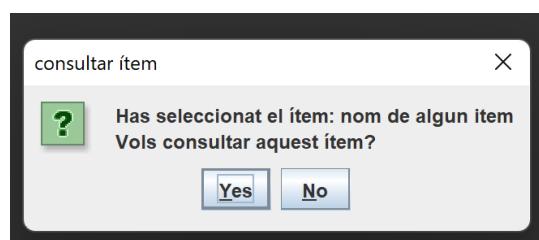
molt baixa qualitat, de fet, recomanacions sense sentit, li mostrarem a l'usuari un missatge informant que per obtenir recomanacions, primer haurà de fer almenys una valoració:



- En el cas que l'usuari tinguia almenys una valoració, llavors mostrarem un conjunt de recomanacions, en que es mostrarà el titol de cada ítem que s'hagi recomanat:



Tal com diu a la interfície, si volem consultar algun d'aquests items, fem doble click sobre l'ítem. Llavors sen's mostrerà el popup:



Si li donem que si, llavors això ens canviàrà el panell inferior de el de la classe panellRecomanacio, que és on estavem, a panellBusca.

El canvi entre panells inferiors el podem fer facilment fent crides a VistaSesioIniciada. Aquesta classe conté mètodes que permeten canviar facilment entre les diverses classes panells.

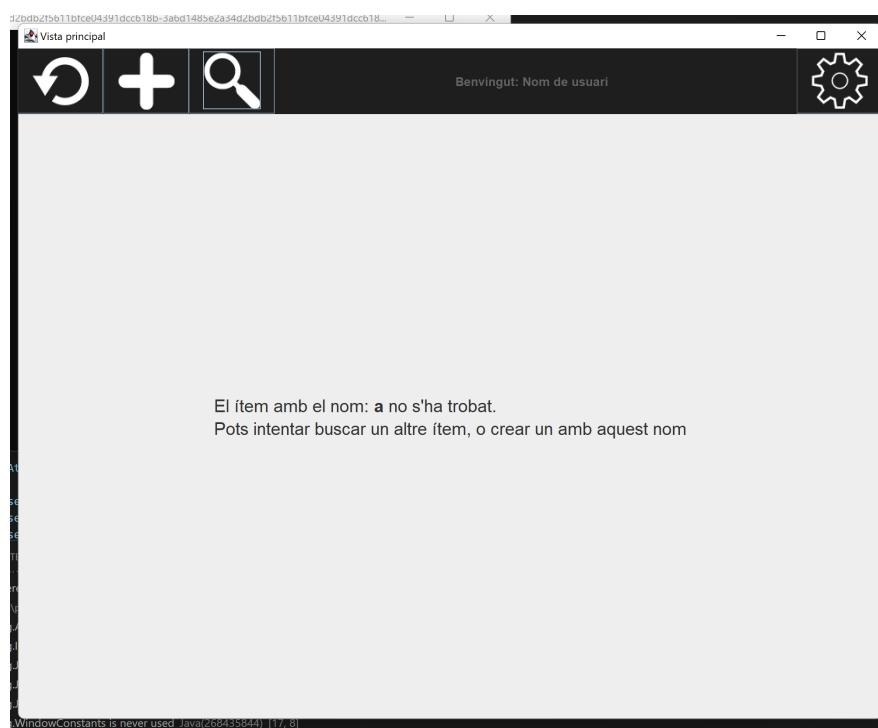
3.5 panellBusca

Aquesta és la classe encarregada de mostrar informació sobre un ítem.

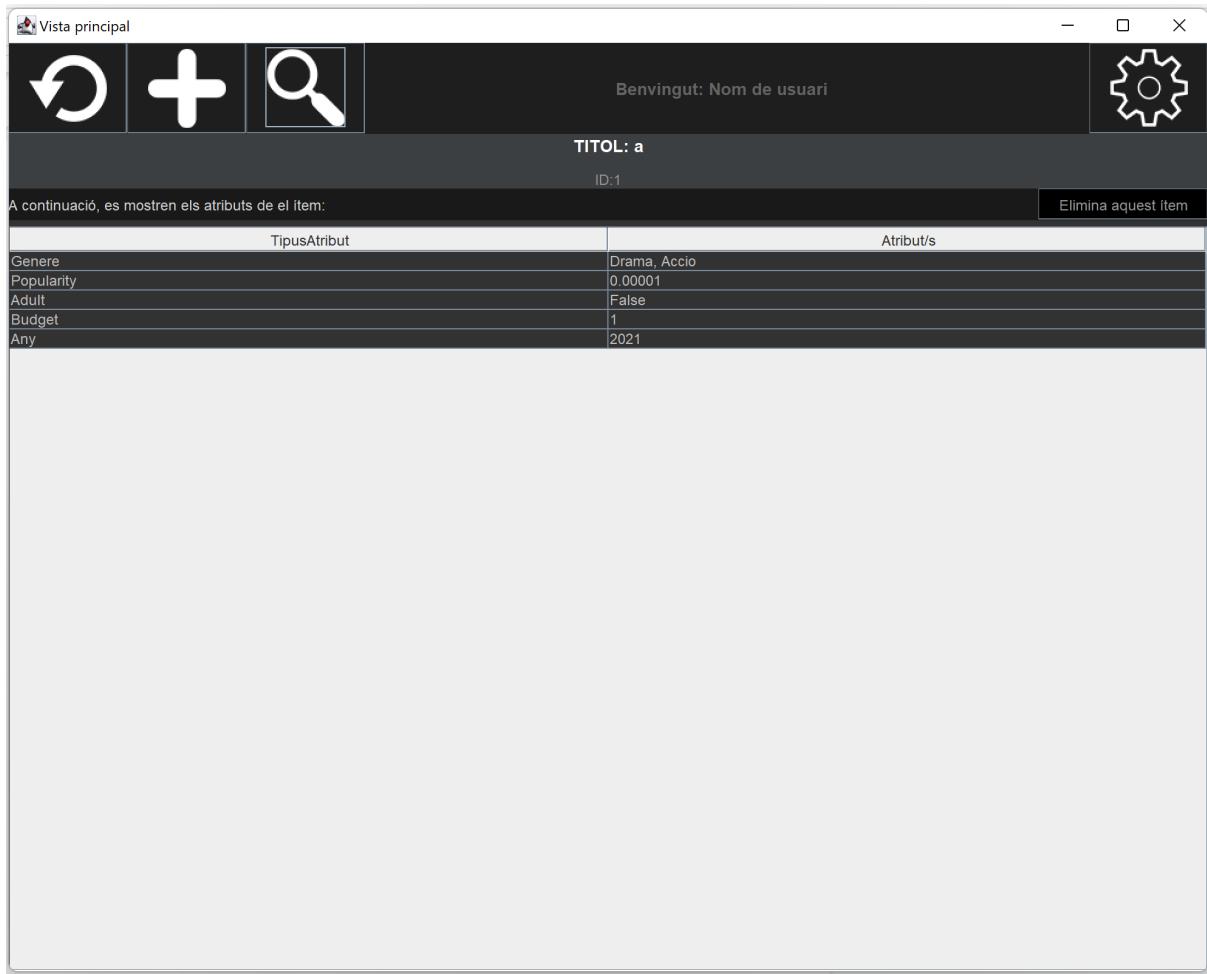
Com que aquest panell, es pot mostrar, o clicant al botó Icona Busca, o fent doble click a un ítem de recomanacions, llavors l'ítem del qual mostrerà informació aquesta classe, serà o aquell que s'ha fet doble click, o aquell en que el nom s'ha introduit en el popup.

Aquest panell, de la mateixa manera que panellRecomanacio, té dos comportaments. En aquest cas, els comportaments són:

- En cas que l'ítem que es vulgui cercar no s'hagi trobat a la base de dades, llavors mostràrà un missatge com aquest:



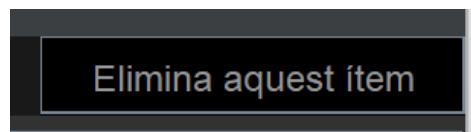
- En cas que l'ítem si que s'hagi trobat, llavors es mostrerà un panell com el següent:



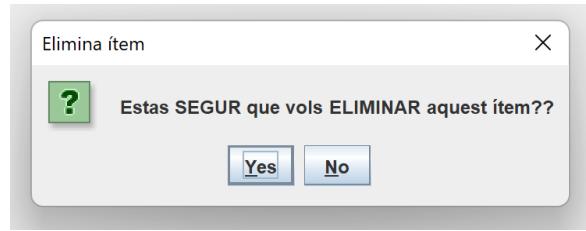
Com podem observar, el panell ens donarà la següent informació:

- Títol de l'ítem.
- ID de l'ítem.
- Atributs per a cada cada TipusAtribut del qual tingui almenys un atribut l'ítem.

A més sota l'ID de l'ítem del qual estem veient la informació, a la dreta tenim un botó que ens permet eliminar-lo:



Si cliquem a aquest botó:

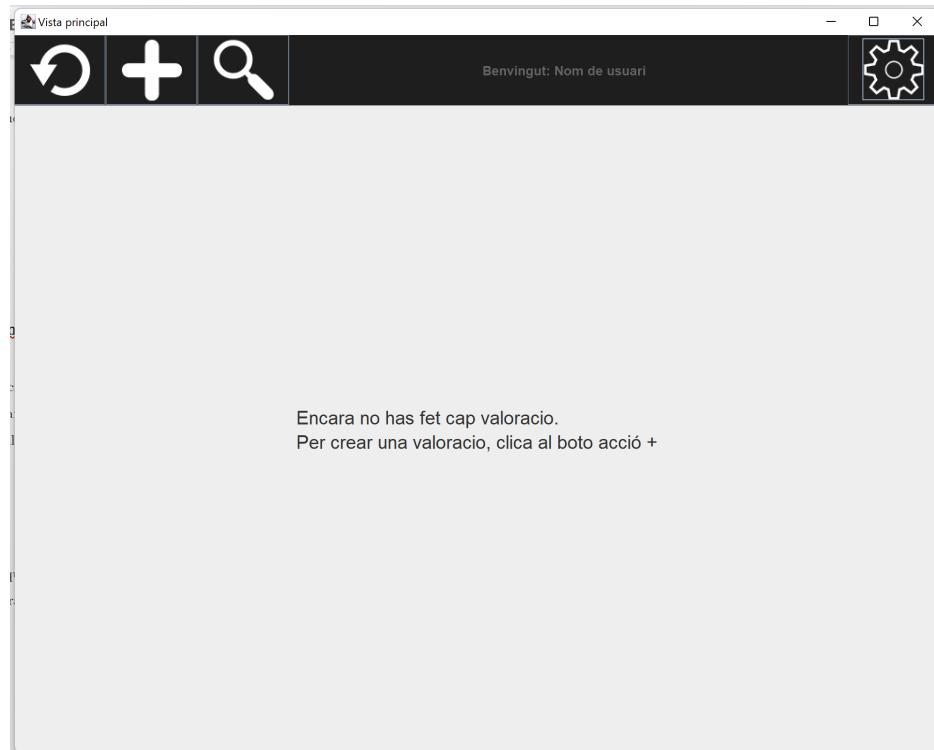


3.6 panellAdministraValoracions

Aquesta és la classe encarregada de mostrar les valoracions de l'usuari que té la sessió iniciada, i per tant, consultar la informació d'aquestes. A més, també ens dóna la possibilitat d'eliminar les valoracions.

Aquesta classe, de la mateixa manera que amb les dos anteriors, té dos comportaments, aquests són:

- En cas que l'usuari no tingui cap valoració, llavors no podem obtenir informació de cap valoració, de forma que es mostrerà el missatge:



- En cas que l'usuari tingui una o més valoracions, llavors es mostrarà el panell:

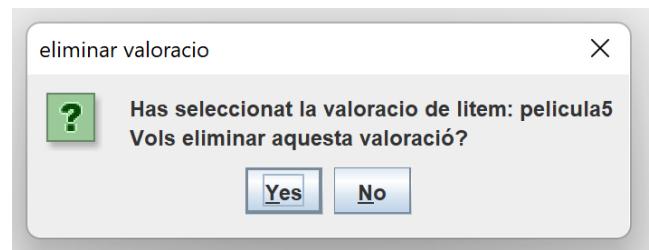
The screenshot shows a software window titled "Vista principal". At the top, there are three icons: a circular arrow, a plus sign, and a magnifying glass. To the right of these icons, it says "Benvingut: Nom de usuari" and features a gear icon. Below this, a section titled "Les teves valoracions" displays a table of reviews. The table has columns for "Titol item", "Puntuacio", and "Comentari". The data is as follows:

| Titol item | Puntuacio | Comentari |
|------------|-----------|------------|
| pelicula5 | 2 | comentari5 |
| pelicula4 | 5 | comentari4 |
| pelicula3 | 3 | |
| pelicula2 | 4 | comentari2 |
| pelicula1 | 5 | |

Al panell, podem observar la informació:

- Una taula amb la informació de cada valoració. Per a cada valoració:
 - L'ítem a la qual pertany.
 - La puntuació que té la valoració.
 - Un comentari. Potser no hi ha.

Si fem doble click a una valoració, tal com la interfície gràfica diu,



Llavors podrem eliminar la valoració que haguem clicat.