



Le génie pour l'industrie

Département de génie logiciel et des T.I.

RAPPORT DE LABORATOIRE

Numéro du laboratoire	03
Étudiant(s) et code permanent	Alexandre Laroche LARA12078907 Carl-Henri Codio CODC11157808 Esdras Metayer METE21058607 DIALLO Amadou Oury DIAA26018801 Sana Bouhadida BOUS27518509
Numéro d'équipe	06
Cours	GTI660
Groupe	01
Chargés(es) de laboratoire	Mirna Awad
Date	9 avril 2019

TABLE DES MATIÈRES

TABLE DES MATIÈRES	2
1. INTRODUCTION	3
2. ANALYSE	4
3. CONCEPTION	6
4. DIAGRAMME DE CLASSES	8
Le diagramme de l'application multimédia	8
Les patrons de conception utilisés	9
5. IMPLÉMENTATION et ALGORITHMES	10
Démarche de la conception	10
Description des classes	10
Algorithme principal	11
6. DISCUSSION	13
Points forts	13
Points faibles	14
Les motivations pour la conception et le design	14
Les choix de l'implémentation	14
Les améliorations possibles	14
7. STRATÉGIE DE DÉTECTION	15
Terminologie	15
La segmentation en plan	15
Algorithme proposé	16
8. CHOIX DES DESCRIPTEURS ET ANNOTATIONS	16
Descripteur	16
Annotations manuelles	17
Prix gagnés (Award)	17
Rang (Ranking)	17
9. DISCUSSION SUR L'ARCHITECTURE CLIENT/SERVEUR	18
10. MANUEL DE L'UTILISATEUR	20
11. CONCLUSION	34
12. RÉFÉRENCES	35

1. INTRODUCTION

Le but de ce laboratoire consiste à développer un système de base de données respectant une architecture trois tierces. Ce système proposera à l'utilisateur d'interroger, de modifier et d'interagir avec une base de données multimédia. Plus précisément, il s'agit d'un système de gestion de téléchargement de films.

Dans les deux premiers laboratoires, les efforts ont été concentrés sur la conception et la création de la base de données multimédia de location de films, ensuite le développement et la création des interfaces utilisateurs d'interrogation ainsi que l'élaboration de requêtes pour la recherche par métadonnées.

L'objectif de ce dernier laboratoire consiste à compléter l'application en y ajoutant des fonctionnalités additionnelles soit de nouvelles fonctionnalités de recherche sur du contenu multimédia en utilisant un descripteur calculé et deux annotations manuelles ainsi que la possibilité de consulter une vidéo en streaming à l'aide du lecteur multimédia vlc. Il sera aussi proposé dans ce rapport un algorithme permettant de trouver automatiquement, dans une séquence vidéo, la séquence de 10 secondes où il y a le plus d'action et de variations.

Les prochains paragraphes présenteront les différentes sections en commençant par les travaux d'analyse, de conception et le diagramme de classes, ensuite l'implémentation et l'algorithme, la discussion, la stratégie de détection, le choix du descripteur et des annotations, la discussion sur l'architecture client/serveur et finalement le manuel de l'utilisateur.

2. ANALYSE

L'application multimédia qui sera proposée à l'issue de ce laboratoire doit permettre d'interagir et de communiquer avec la base de données conçue au premier laboratoire afin de répondre au mieux aux exigences des utilisateurs. Ainsi, il s'agit de développer une application interactive pour la consultation de contenu multimédia. L'application doit proposer une interface conviviale qui permettra à l'utilisateur d'interroger, de modifier et d'interagir avec la base de données et d'effectuer un ensemble de requêtes. La conception et l'implémentation de l'application de téléchargement de films ont été réalisées à travers trois laboratoires.

Au premier laboratoire, un modèle conceptuel a été conçu dans le but de présenter l'ensemble des classes, leurs attributs et les différentes relations reliant les classes entre eux. Par exemple les classes Rôle, Client et Réalisateur, Scénariste qui héritent de la classe Personnes puisqu'elles ont toutes des attributs communs tels que: *idpersonne*, *nom*, *prénom*, *dateDeNaissance* et autres. Une relation chevauchante et complète entre les classes RÉALISATEUR, RÔLE, CLIENT, SCÉNARISTE et PERSONNE a été identifiée.

Le modèle conceptuel a ensuite été transformé en un modèle relationnel. En effet, les classes ont été transformées en tables. Les attributs, la clé primaire et/ou étrangère ont été définis pour chaque table. Aussi, un traitement spécial a été accordé aux classes qui présentent des relations de multiplicité plusieurs à plusieurs (N à N), telle que Genre et Films, Pays et Films en ajoutant une table intermédiaire.

La conception du modèle relationnel a aussi permis d'identifier plusieurs contraintes d'intégrité référentielle qui ont été prévues lors de la création de la base de données. Elles sont définies par plusieurs éléments dans une table comme les PRIMARY KEY, FOREIGN KEY, NOT NULL, CHECK et autres.

Une interface interactive a aussi été développée afin de permettre la connexion à la base de données, l'exécution des scripts SQL pour la création des différentes tables et des contraintes, ainsi que l'insertion des données provenant des fichiers XML.

Au deuxième laboratoire, une application multimédia a été développée afin de permettre à un utilisateur d'interagir plus efficacement avec la BD pour la recherche de métadonnées. Une première interface a été développée afin de permettre à l'utilisateur de se connecter à l'application en entrant ses informations d'identification (le login et le mot de passe).

Une deuxième interface doit permettre à un client de modifier et de mettre à jour son profil à savoir ses informations de connexion, ses informations personnelles, son adresse, ainsi que les informations reliées à sa carte de crédit.

Une autre interface doit permettre à un client d'effectuer une recherche de film selon plusieurs critères soit le titre, un intervalle d'année, un intervalle de durée, la langue, la personne et le genre. La même interface donnera aussi la possibilité de rechercher un scénariste, un réalisateur ou un acteur. Ainsi, le client peut sélectionner un film, pour ensuite l'ajouter ou le retirer de son profil.

Une interface doit permettre au client de visualiser les détails sur une personne trouvée après une recherche, aussi de visualiser la photo de cette même personne. D'autres éléments ont aussi été développés comme une barre d'état avec un composant graphique pour illustrer l'état de la connexion avec la base de données (fermée, en cours, indéterminée). Aussi, une barre de menu a été intégrée à l'interface graphique afin de permettre à l'utilisateur de naviguer facilement entre les différentes fonctionnalités ainsi qu'une fenêtre d'aide comportant les raccourcis des différentes fonctionnalités de l'application.

Au troisième laboratoire, l'application doit permettre à un utilisateur de consulter une vidéo en streaming à l'aide du lecteur multimédia vlc. Le lecteur vidéo comprendra les fonctions de lecture telles que pause, jouer, arrêter ainsi qu'un curseur permettant de se déplacer dans la vidéo. La fenêtre de consultation devra permettre de consulter la vidéo référencée par l'URL.

L'application doit en plus intégrer une nouvelle fonctionnalité de recherche au niveau du contenu multimédia. Pour ce faire, une nouvelle interface de recherche sur le contenu doit être développée pour 10 films. La recherche se fera selon des critères qui portent sur deux annotations manuelles choisies et un descripteur calculé. Les annotations et le descripteur devront être conçus avec un formalisme équivalent à MPEG 7.

3. CONCEPTION

Afin de répondre aux exigences du présent laboratoire, trois nouvelles colonnes de type XML ont été ajoutées dans la table FILMS de la base de données pour décrire le contenu d'un poster. La première colonne comprendra l'annotation le prix gagné (Award) ayant les balises Catégorie, Prix gagné et Année. La structure sera donc la suivante:

```
<award>
  <winner>...</winner>
  <annee>...</annee>
  <categorie>...</categorie>
</award>
```

La deuxième colonne comprendra l'annotation le plus haut rang occupé pour les cinq premières semaines (ranking) ayant les balises Rang, Semaine, Année. La structure sera donc la suivante:

```
<ranking>
  <annee>...</annee>
  <mois>...</mois>
  <semaine>... </semaine>
  <rang>...</rang>
</ranking>
```

La troisième colonne comprendra le descripteur calculé Dominant Color qui contient les valeurs RGB pour la couleur dominante d'un poster et ce pour les 10 films choisis. La structure sera donc la suivante:

```
<color>
  <red>..</red>
  <green>..</green>
  <blue>..</blue>
</color>
```

Il est important de souligner que les annotations ainsi que le descripteur ont été conçus chacun dans un fichier XML séparé. En effet, cette décision favorise la performance de l'application en termes de rapidité et d'amélioration de temps de réponse lors de la recherche de contenu.

Aussi, il est à noter que pour effectuer des comparaisons visuelles au niveau de l'interface, une librairie XML type a été implémentée pour récupérer et afficher du contenu XML dans la base de données.

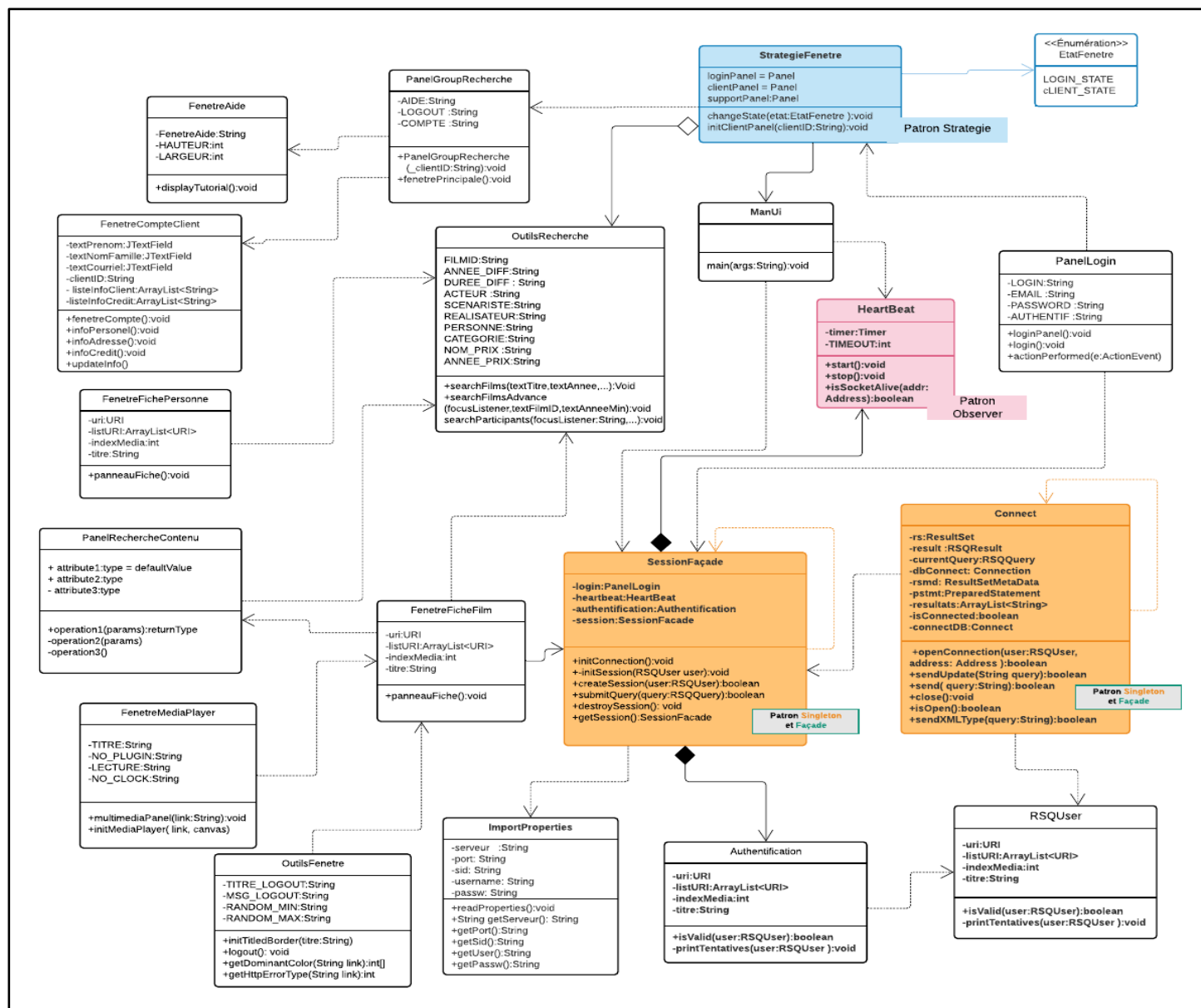
Une attention particulière a été portée sur les liens multimédias des films (pour les posters et les vidéos), les liens en question ne fonctionnent pas pour des problèmes de sécurité. Alors, un système a été mis en place pour récupérer le type d'erreur et de le traiter en conséquence.

En ce qui concerne la consultation de vidéo, la fenêtre de lecture vidéo a été implémentée grâce au framework VLC Java. En effet, le lecteur vlc permet d'intégrer toutes les fonctionnalités de lecture vidéo comme pause, jouer, arrêter ...

La nouvelle interface de recherche qui sera développée permettra à un utilisateur d'effectuer une recherche de films selon soit les deux annotations (classement et le prix gagné (Awards)) ou le descripteur la couleur dominante. La recherche de films selon le prix gagné peut se faire à travers des critères comme catégorie, année et prix gagné alors que la recherche de films selon le classement peut se faire à travers des critères comme l'année, la semaine, le rang. En ce qui concerne la recherche selon la couleur dominante, il est possible d'afficher l'image importée et l'image de sa couleur dominante pour faciliter la comparaison visuelle. Aussi, il est possible de modifier la mesure de tolérance ainsi que d'afficher les valeurs des couleurs RGB et de les modifier pour obtenir une nouvelle couleur et faciliter les tests de comparaison de couleur dominante.

Pour tous les films retournés par la recherche, il est possible d'afficher la fiche complète du film en question comprenant, en plus des informations de base, son classement dans le box-office, les prix gagnés ainsi que les valeurs RGB et la couleur dominante de son poster, la différence de pourcentage entre le poster et l'image de la couleur dominante importée.

Le diagramme de l'application multimédia



Le diagramme de classe présente le schéma des classes qui constituent l'application et les différentes relations entre elles. Mais seuls les éléments (classes, attributs, méthodes et liens) pertinents ont été ajoutés dans ce diagramme afin de faciliter la compréhension.

Les patrons de conception utilisés

L'application contient un patron observateur pour gérer pour aviser l'état de la connexion, un patron Stratégie pour la gestion des panels, deux patrons singletons pour la session et la connexion et un patron de façade.

La classe **HeartBeat** implémente le patron observer, elle teste de façon continue toutes les deux secondes la connexion, dès qu'elle détecte une connexion elle avise le changement d'état aux labels de l'interface.

La classe **StratégieFenetre** implémente le patron stratégie, elle gère le changement des panels entre PanelRecheGroupe et PanelLogine. Lorsqu'il s'agit d'une connexion, elle appelle la classe **PanelLogin** et s'il s'agit d'une recherche c'est la classe **PanelRechercheGroupe** qui est appelée.

La classe **SessionFaçade** implémente à la fois le patron Singleton et le patron Façade. L'utilisateur doit passer par session pour s'authentifier et permet d'éviter plusieurs sessions.

En ce qui concerne le singleton de la classe **Connect**, il permet de maintenir une seule connexion à la base des données.

5. IMPLÉMENTATION et ALGORITHMES

Démarche de la conception

Vu que l'application est entièrement développée avec le framework Java, il était naturel d'utiliser les méthodes disponibles dans la librairie «*uk.co.caprica.vlcj*» pour faciliter la lecture des annonces et l'affichage des posters selon les exigences du laboratoire. Un package est créé au niveau du serveur afin d'organiser les classes dédiées au MediaPlayer. Afin d'identifier la couleur dominante des posters dans la recherche sur le descripteur calculé, les méthodes offertes dans la librairie «*de.androidpit.colorthief*» sont utilisées pour simplifier l'implémentation.

Les patrons ci-dessous sont utilisés pour cacher la complexité existante au niveau du serveur, pour la création d'un objet unique de connexion ainsi qu'une session unique, pour la création d'objets pouvant effectuer plusieurs traitements différents et la détection des pertes de connexion réseaux à partir du HeartBeat ce depuis le deuxième laboratoire. Il est important aussi de mentionner que les patrons sont visuellement très bien représentés dans le diagramme de classe.

Description des classes

Uniquement les nouvelles classes créées dans cette itération sont décrites dans cette section.

Classe	ControlsPanel
Description	Cette classe permet de contrôler une vidéo avec des options telles que pause, stop, skip, etc. Le panneau regroupe les options pour faciliter son implémentation avec le panneau de lecture vidéo.

Classe	FenetreMediaPlayer
Description	Cette classe permet l'ouverture des fichiers vidéo dans une nouvelle fenêtre. Lecture du contenu avec les options de contrôles.

Classe	QueryContenuLibrary
--------	---------------------

Description	Cette classe contient toutes les requêtes nécessaires à la recherche sur les Contenus. On passe par la Session pour envoyer les requêtes à la base de données.
--------------------	--

Classe	OutilsRecherche
Description	Cette classe regroupe tous les outils nécessaires au support de la recherche de Film, Personne ou Contenu. Permet aussi de gérer la recherche et la manipulation des locations du client.

Classe	OutilsFenetre
Description	Cette classe regroupe plusieurs outils de support au visuel à l'interface utilisateur.

Classe	PanelRechercheContenu
Description	Cette classe définit le panneau qui permet la recherche de contenu pour trouver des titres de films, la recherche par awards, par classement et par couleur similaire au Poster d'un Film.

Algorithme principal

Le pseudocode suivant décrit le mécanisme mis en place dans l'application lorsqu'un usager démarre une recherche dans le contenu des prix gagnés. Afin de faciliter l'écriture du pseudocode seul, les classes et méthodes utilisées dans la recherche des prix gagnés sont représentées dans l'algorithme ci-dessous.

Rechercher un film à partir des prix gagnés
Étape #1

Début classe **FenetreRecherche**

Déclarer DefaultListModel<String>: listFilmAwards

Fonction **actionPerformed**

Appel : fonction **searchAwards** de la classe **OutilsRecherche**

Fin fonction

Fin

Étape #2

Début classe **OutilsRecherche**

Déclare String : contenu recherché

Fonction **searchAwards** (

Déclarer DefaultListModel<String>: listeFilmAwards,

String focusListener)

Nettoyer **listeFilmAwards**

Si focusListener est contenu recherché

Appel : fonction **findAwardsNomPrix** de la classe **QueryContenuLibrary**

Appel : des fonctions getSession().getResultats() de la classe **SessionFacade**

Pour i allant de 0 à la longueur du résultat

Déclarer String titreFilm

titreFilm ← les éléments de résultats trouvés

Si **listeFilmAwards** n'a pas titreFilm

ajouter le titreFilm dans la liste

Fin fonction

Fin

Étape #3

Début classe **QueryContenuLibrary**

Fonction **findAwardsNomPrix**(chaîne de caractères : NomPrix)

Déclarer String: requete

requête ← SELECT titre FROM films WHERE

REGEXP_LIKE (award.extract('/award/winner/text()').getStringVal(), nomPrix);

Appel : les fonctions **SessionFacade.getSession().submitQuery**(requete)

Fin fonction

Fin

Étape #4

Début classe **SessionFacade**

Déclarer objet **Session**: SessionFacade

Fonction static **getSession ()**

Retourner session

Fin fonction

Fonction ArrayList<String> **getResultats()**

Retourner résultats

Fin fonction

Fonction logique **submitQuery**(chaîne de caractères : query)

Si une connexion est ouverte

Appel : des fonctions getConnect().sendQuery(query) de la classe **Connect**

Fin fonction

Fin

6. DISCUSSION

Points forts

- À ce stade du développement, il est possible à un utilisateur de lire le contenu multimédia depuis l'application. Cette dernière facilite la recherche au niveau des contenus multimédias à partir de l'interface, l'ajout d'une nouvelle colonne pour décrire le contenu multimédia de type XMLType a été effectué. Ces aspects pris en compte facilitent la recherche avancée.
- L'implémentation de l'affichage d'un poster à partir de l'URL est bien fonctionnelle, avec la différence en pourcentage entre un poster et l'image importée.
- Si les champs de recherche sont vides, on désactive les boutons de recherche pour éviter tous problèmes (donc pas de recherche de champs vide. Évite le traitement de requêtes inutiles). Aussi, en exemple, un champ année supporte juste quatre(4) caractères et supporte juste les chiffres.

Points faibles

- Si l'application permettait d'intégrer du code avec les langages xslt ou xquery, cela pourrait faciliter la vie des utilisateurs en retournant avec plus d'aisance des résultats sur les recherches dans les contenus.
- Un développement de l'application avec Maven aurait permis de mieux gérer les dépendances.

Les motivations pour la conception et le design

En ce qui concerne l'implémentation des techniques de lecture de vidéo dans l'application avec VLC, pour ce faire l'utilisation du framework vlcj a été nécessaire. Ce dernier présente des avantages comme le support complet de vidéo à 360 degrés, modification de la hauteur, du lacet, du roulis et du champ de vision, pour ne citer que ceux-là. Le vlcj est développé et testé sur les plateformes Linux ainsi que Windows.

Les choix de l'implémentation

L'application contient un exportateur XML des résultats de la liste des informations sur un film. Au moyen du contrôleur Façade, elle cherche les informations dans les tables de la Base de données et les retourne. D'autres méthodes utilitaires sont définies pour créer un nœud de texte, des sous-groupes avec attribut. La classe QueryContentLibrary implémente des requêtes qui facilitent la recherche de contenu. On passe par la session pour envoyer les requêtes vers la base de données dans le but de respecter le patron de conception Façade.

Les améliorations possibles

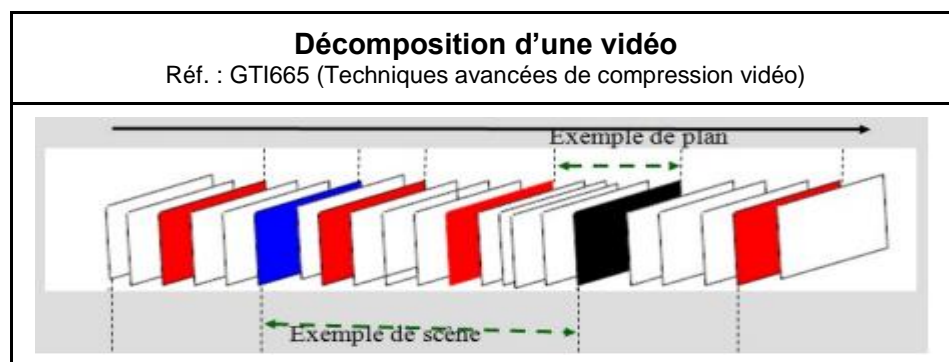
Dans les prochaines itérations, l'interface de l'application devrait s'adapter à toutes les dimensions d'écran, car avec cette version, sur certains ordinateurs, tout dépend de la résolution, le bord inférieur de la fenêtre principale reste caché à l'utilisateur. D'autres évolutions pourraient aussi être envisagées pour la recherche de contenu en utilisant des technologies proches de XML.

7. STRATÉGIE DE DÉTECTION

L'analyse de séquences de vidéo se révèle utile dans plusieurs domaines comme la vidéosurveillance, le sport, la biologie et la chimie, etc. Suivant la thèse doctorale de Lefèvre: Les méthodes peuvent être basées sur les pixels, les histogrammes, un découpage en blocs, des caractéristiques robustes extraites dans l'image, une information liée au mouvement ou encore une combinaison de ces différentes approches. Dans les paragraphes suivants, un algorithme permettant de trouver automatiquement, dans une séquence vidéo, la séquence de 10 secondes où il y a le plus d'action ou le plus de variations basées sur les variations de l'intensité des pixels des images sera présentée.

Terminologie

Par définition une vidéo est une succession d'image (frame) regroupée en scènes et prise. Les scènes ou segments sont des regroupements logiques des prises d'un événement. Les prises sont des successions d'images capturées en continu selon les notes du cours 9 sur les bases de données vidéo et audio. Dans la littérature, l'unité de base pour le traitement et la manipulation des séquences vidéo c'est le plan. De ce fait, la scène devient une collection de plans adjacents dans le temps sémantiquement liés.

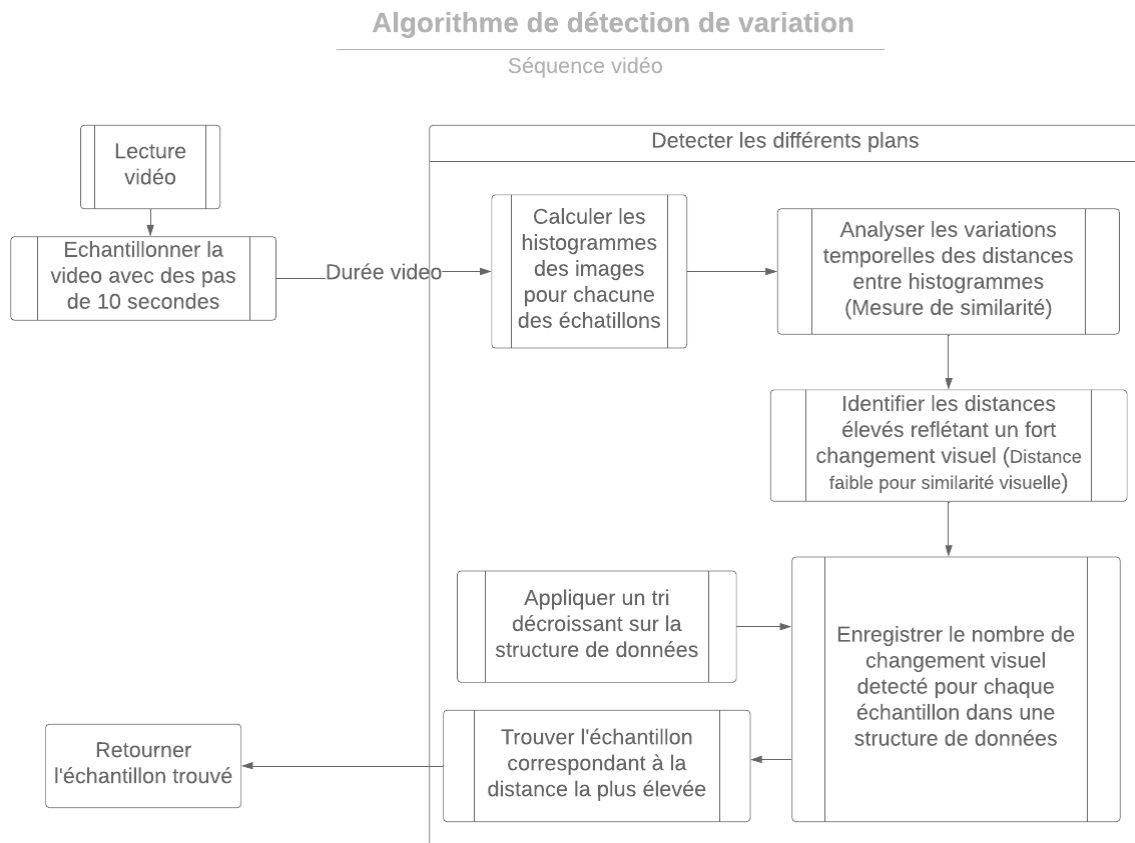


La segmentation en plan

Selon Wikipédia, une segmentation en plans consiste à détecter les différents plans d'une vidéo. Pour cela, une méthode consiste à calculer un histogramme pour toutes les images de la vidéo, et d'étudier ensuite les variations temporelles des distances entre histogrammes consécutifs. Une distance faible étant synonyme de similarité visuelle, et une distance élevée reflétant un fort changement visuel. Les changements de plans sont alors détectés comme étant les pics de ces variations temporelles.

Algorithme proposé

Une approche algorithmique de la détection d'une séquence de 10 secondes dans une vidéo où il y a le plus de variation tenant compte des considérations ci-dessus se traduit comme suit dans une structure de bloc:



8. CHOIX DES DESCRIPTEURS ET ANNOTATIONS

Descripteur

Dans la liste des descripteurs proposés, le choix de la couleur dominante est le plus évident compte tenu de la facilité offerte par les librairies Java dans le traitement des images. L'utilisation de la couleur dominante permettra à un utilisateur de trouver un film pour lequel la couleur dominante de son poster est similaire à un autre poster choisi aléatoirement. Les composants RVB (Rouge Vert Bleu) des images sont récupérés individuellement pour chacun des posters disponibles dans la base de données pour les dix films choisis. Une mise à jour de la fiche des films est réalisée via une requête XQuery afin de les rendre disponibles sous forme de métadonnées calculées selon les exigences du laboratoire.

Pour assurer la similarité dans la recherche des couleurs dominants la notion de pourcentage d'erreur entre les posters disponibles dans la base de données et l'image utilisée pour la recherche est implémentée. Grâce à cette variable modifiable via l'interface de recherche, l'utilisateur a la possibilité d'augmenter la marge d'erreur ou réduire la similarité pour aller chercher le plus de films ayant un poster semblable au poster utilisé.

Annotations manuelles

Les deux annotations choisies sont ***prix gagnés (Award)*** et le ***plus haut rang occupé (Classement)***.

Prix gagnés (Award)

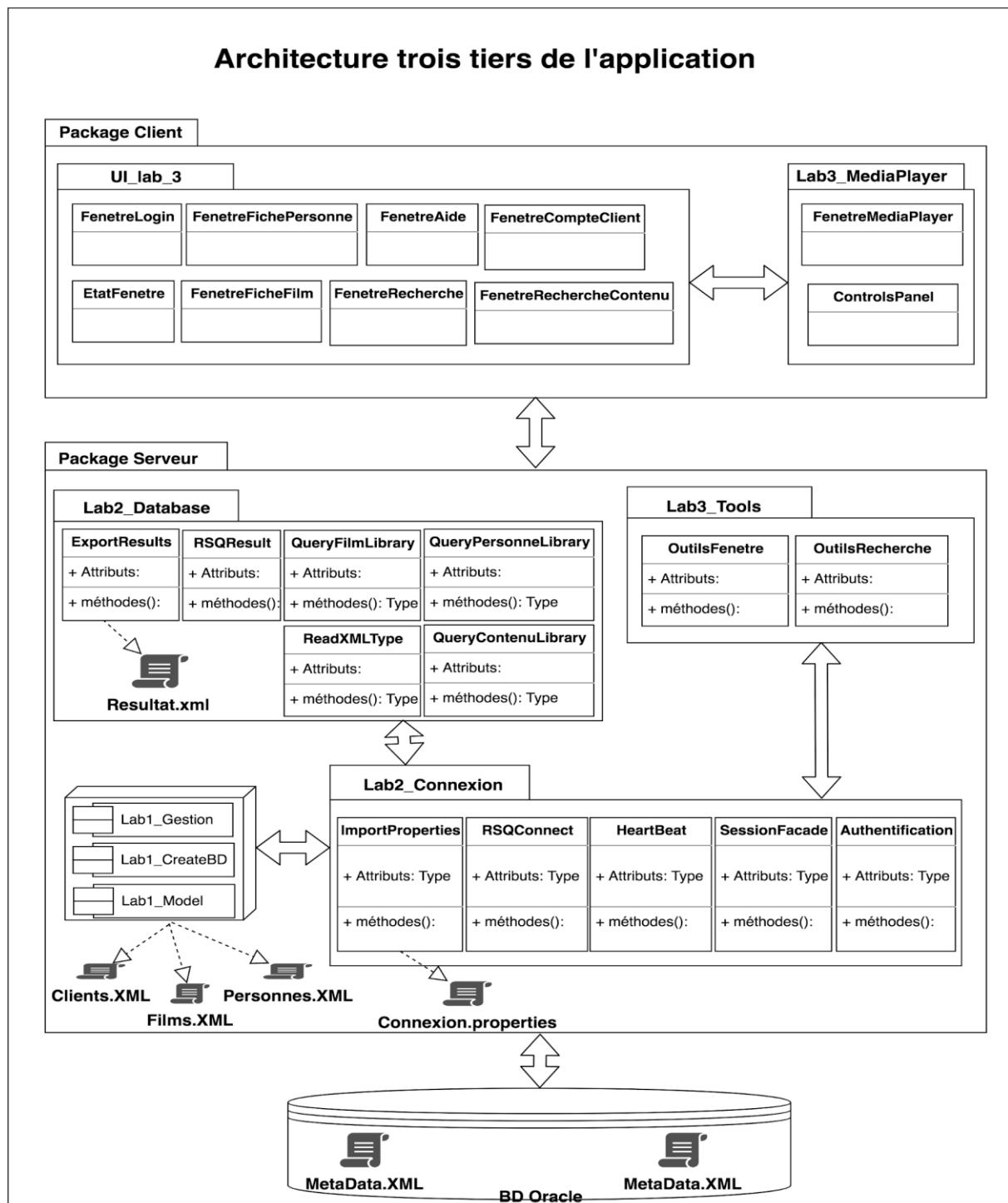
Généralement les promoteurs utilisent la nomination de leur film pour en faire la promotion dans la bande-annonce. Lorsque le prix (Award) est gagné, les gens sont naturellement portés à en faire la location. D'où l'intérêt de donner la possibilité aux clients de faire des recherches pour trouver les films qui ont gagné un prix en particulier (exemple : le **ASCAP Award** est gagné par les films **Cars, Batman & Robin, etc.**) ou dans une catégorie spécifiée (exemple : les films **Cars, Batman & Robin** ont gagnés dans la catégorie **Top Box Office Films**) en utilisant la recherche dans le contenu. L'utilisateur

Rang (Ranking)

Les films qui ont la cote élevée sur une période limitée dans le temps, une semaine, un mois, etc.. Font le plus souvent plus d'argent que les autres dans cet intervalle de temps. Dans un système de location de film, il est recommandé de donner la possibilité aux gens de visualiser le rang qu'occupait un film dans le **box-office**. D'où l'intérêt de donner la possibilité aux clients de faire des recherches pour trouver les films qui ont occupé un rang donné (exemple : les films **Cars, Batman & Robin** ont déjà occupé le premier rang du box-office) ou un rang ainsi que les semaines d'occupation à partir de la date de sortie (exemple : les films **Cars, Batman & Robin** ont occupé le premier rang à leur première semaine de sortie).

9. DISCUSSION SUR L'ARCHITECTURE CLIENT/SERVEUR

Le schéma ci-dessous se veut une représentation à haut niveau des principales classes de l'application. Pour satisfaire les requis du laboratoire une architecture composée de trois couches doit être utilisée soit pour la présentation, le traitement et d'accès aux données. Cette architecture, dite trois tiers, a l'avantage de rendre possible la séparation des routines d'exécution, une meilleure sécurité sur les données et une gestion centralisée des ressources.



Ainsi, le package client contient toutes les classes permettant les interactions entre les utilisateurs et l'application que ce soit pour la recherche de film, l'affichage de la fiche d'un client jusqu'à faire jouer la bande-annonce d'un film.

Le package serveur pour sa part contient la partie fonctionnelle de l'application. Toute la logique pour le traitement des requêtes, la connexion avec la base de données, l'exportation des données de recherche dans un fichier externe sont exécutées en utilisant les classes correspondantes. Cette couche utilise le patron Façade pour les échanges avec l'interface utilisateur et le patron singleton pour créer une instance unique de connexion avec la base de données.

La base de données oracle est utilisée pour persister les données sur les films, personnes et clients initialement contenues dans des fichiers XML. Lors de la première itération du laboratoire, des fonctions ont été implémentées afin d'extraire des fichiers XML les données et les importer dans la base de données. La deuxième itération a permis l'exploitation des informations sous forme de recherche. Dans cette dernière itération des métadonnées sont ajoutées sur des tuples choisis selon les requis du laboratoire, d'où la représentation des fichiers XML directement dans la base de données.

L'inconvénient principal de cette architecture se situe au niveau du serveur, car la performance du système se repose sur le bon fonctionnement du package serveur. Une requête non optimale peut engendrer automatiquement affecter l'expérience utilisateur.

10. MANUEL DE L'UTILISATEUR

Dans un premier temps, il faut spécifier que l'application en question a été construite entièrement en utilisant le langage de programmation *Java*. Au début du projet, il était question d'utiliser le framework *Spring*, mais pour des raisons de connaissances et de ressources, l'équipe a dû orienter le développement vers une solution locale. En d'autres mots, l'application *Java* consiste en un regroupement d'objets *Swing* qui permet à un utilisateur d'accéder avec ses informations personnelles à une interface pour effectuer la recherche de contenu multimédia.

Pour démarrer l'application, l'utilisateur peut exécuter le fichier de type “*.jar*” qui rassemble tous les éléments de l'application ou exécuter la classe *MainUI.java* située dans le package *lab_3_ui* en utilisant un *IDE Java* contenant le code source. Il est important de prendre note que l'application nécessite une version de *Java JRE/JDK* supérieur à 7 (1.7).

Avant de commencer à donner les directives d'utilisation de l'application, il sera pratique pour un développeur de comprendre que plusieurs librairies ont dû être importées dans le code source afin de supporter les différentes fonctionnalités de l'interface. La librairie en question consiste en un regroupement de fichiers de type “*.jar*” dans le dossier *Lib* du code source. Idéalement il aurait été plus logique d'utiliser *Maven* pour l'exécution de l'application et pour l'importation des librairies (dépendances) à partir d'un simple fichier “*pom.xml*”. Ci-dessous, vous retrouverez la liste actuelle des librairies de l'application.

- | | |
|--------------------------------|---|
| - ojdbc8.jar: | Connexion à une base de données Oracle (12c). |
| - vlcj-3.0.1.jar: | Lecture de contenu multimédia (vidéo et image). |
| - vlcj-4.0.6-test-sources.jar: | Icônes retrouvées dans la fenêtre de lecture vidéo. |
| - vlcj-natives-4.0.2.jar: | Supporte le framework VLC et ses librairies. |
| - jna-3.5.2.jar: | Supporte la librairie vlcj-natives. |
| - color-thief-1.1.2.jar: | Obtention de la couleur dominante d'une image. |
| - xmlparserv2-12.2.0.1.jar: | Permet la lecture d'un champ XMLType de la DB. |
| - xdb.jar: | Permet de récupérer les champs XMLType de la DB. |
| - xmlpull_1_1_3_4c.jar: | Permet la lecture d'un Format XML ou fichier XML. |

L'application en question avec les librairies actuellement mises en place fonctionne sur les systèmes d'exploitation *Windows* et *MacOs*. Pour *Windows*, les versions supérieures à *Windows Vista* sont supportées. Pour *MacOs*, les versions supérieures à *Mac Tiger* sont supportées. Pour *Mac*, il faut cependant changer dans le code source de la classe *FenetreMediaPlayer.java* le répertoire qui pointe vers le logiciel *VLC* installé sur le poste. Par exemple, sous *Windows*, il faut pointer vers le répertoire source du logiciel *VLC*.

Sans plus tarder, la suite consiste à décrire et donner les directives d'utilisation des différentes parties de l'interface de l'application. Premièrement, au démarrage de l'application, on affiche la page de connexion qui permet à l'utilisateur d'entrer son adresse courriel et son mot de passe pour démarrer une nouvelle session. Le **bouton de connexion** ("Login") s'active seulement lorsqu'il y a une connexion valide vers la base de données et qu'il y a bien une connexion internet. L'interface affiche l'état de la connexion en temps réel afin de faciliter le diagnostic en cas de problème. Aussi, si l'adresse courriel entrée n'existe pas dans la base de données ou que le mot de passe ne correspond pas à l'adresse courriel entrée, un message d'erreur personnalisé sera affiché. La figure ci-dessous permet de visualiser la situation décrite.



Pour vérifier en temps réel l'état de la connexion avec la base de données, un système "heartbeat" a été mis en place. La figure ci-dessous montre ce qu'il se passe en arrière-plan de l'application lorsque la connexion à internet est soudainement rompue. Dans la figure, on voit une rupture de connexion ainsi qu'une tentative forcée vers la base de données effectuée par l'application.

```
*****
*                               *
*      BIENVENU !               *
*                               *
*  Recherche de Films et de Personnes. *
*  Location Films et modification compte.*
*                               *
*****
(Connect) Connection String: jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 EQUIPE119 eVWNC13U
(Connect) Connection en cours ...

*****
*                               *
*      Connexion à la DB réussi ! *
*                               *
*****
(Heartbeat) Connexion DB impossible - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 No route to host: connect
(Heartbeat) Connexion DB impossible - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 No route to host: connect
(Heartbeat) Connexion DB impossible - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 No route to host: connect
(Heartbeat) Connexion DB impossible - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 No route to host: connect
(Heartbeat) Connexion DB impossible - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 No route to host: connect
(Heartbeat) Tentative Connexion DB - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 gti660ora12c.logti.etsmtl.ca
(Heartbeat) Tentative Connexion DB - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 gti660ora12c.logti.etsmtl.ca
(Heartbeat) Tentative Connexion DB - jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660 gti660ora12c.logti.etsmtl.ca
```

Par la suite, lorsque les informations d'authentification ainsi que de connexion sont valides, l'utilisateur tombe sur la **page principale** de recherche de contenu. À haut-niveau, l'interface est composée d'une barre d'entête, une section pour la recherche de titres film, une section de recherche-personnes, une section de recherche par contenu film et une section pour la gestion des locations de films. La figure ci-dessous illustre la page principale en question. Les différentes sections de cette page seront traitées un peu plus loin.

The screenshot displays a web application interface with a top navigation bar and a main content area. The top bar includes a home icon, a user account section with 'Mon Compte' and 'Déconnexion' links, and a status bar showing user information and connection details.

INFORMATION

[997690] Email: RobertCFlores21@gmail.com Heure de Connexion: 23:03:20
 État Connexion: **Ouvert** Serveur: jdbc:oracle:thin:@gti660ora12c.logti.etsmtl.ca:1521:GTI660

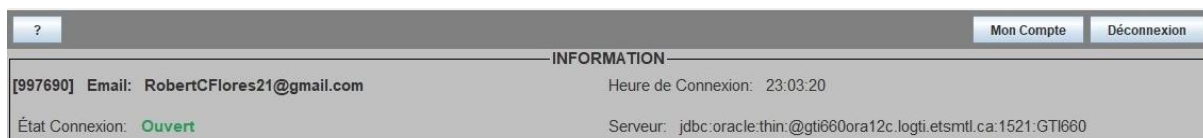
The main interface is divided into several sections:

- Recherche Film / Recherche Contenu**: Two tabs for searching by film or content.
- PERSONNE**: Search form for names and identifiers, with a 'Search Personne' button.
- PARTICIPANT**: Search form for roles (Director, Screenwriter, Actor) with a 'Search Participant' button.
- FILM**: Search form for film titles, languages, summaries, years, durations, and genres, with 'Search Film' and 'Export xml' buttons.
- HISTORIQUE RECHERCHE**: A list of recent searches with details like title, director, and year.
- RESULTATS - PERSONNES**: A list of search results for people, including Mark Steven Johnson, Steven Soderbergh, Steven Spielberg, Robert Stevenson, and Steven Brill.
- RESULTATS - FILMS**: A list of search results for films, including 'The Apartment', 'Spartacus', and 'Psycho'.
- LOCATION FILMS**: A section showing the total rental price for selected items (Indiana Jones and the Last Crusade, Batman & Robin, Cars) as 37.4 \$.
- Buttons**: 'Louer Film' (Rent Film) and 'Retirer Film' (Remove Film) buttons at the bottom.

En tout temps, si l'utilisateur clique sur la fermeture de la fenêtre principale, l'application demandera la confirmation de l'utilisateur. Cela permet à l'application de tout fermer adéquatement, inclut la connexion au serveur, la connexion du lecteur multimédia, etc.

Tout comme la page de connexion, si les champs de recherche sont vides, on désactive les boutons de recherche afin d'éviter l'exécution de requêtes vides et inutiles. En fait, tous les champs ont des restrictions d'entrées d'information. Les champs qui s'attendent à recevoir des chiffres n'accepteront que l'entrée de chiffre et vice-versa. Par exemple, une année ne devrait pas avoir plus de quatre chiffres et devrait être sous l'année actuelle. Pour les champs qui supportent uniquement les caractères de type alphabets, les lettres majuscules et minuscules sont supportées et n'auront pas d'impacts sur la recherche.

Pour ce qui est de la **barre d'entête** de l'application, elle sert principalement à donner à l'utilisateur la possibilité de gérer ses informations de compte, de fermer la session en cours et d'afficher certaines informations de connexion. La figure ci-dessous illustre la barre d'entête en question.



Au niveau du **panneau information** qui fait partie de la barre d'entête de l'application, on affiche l'identifiant et le courriel électronique de l'utilisateur actuel de la session. Aussi, l'adresse du serveur, l'état de la connexion avec la base de données et l'heure de connexion est affiché à des fins de contrôle et de tests de développement.

Le premier **bouton d'aide** ("?",) situé sur la gauche du panneau permet d'ouvrir une page qui illustre de façon visuelle les différentes parties de l'application. En d'autres mots, la page d'aide permet d'aider les nouveaux utilisateurs de l'interface à comprendre son fonctionnement.

Le **bouton de déconnexion** situé complètement à droite ("Déconnexion") permet à l'utilisateur de fermer une session en cours à tout moment. Si la connexion est fermée et que l'utilisateur souhaite se déconnecter de sa session, l'application gère très bien la situation.

Lorsque le **bouton compte** ("Mon Compte") est sélectionné, une nouvelle fenêtre s'ouvre et affiche l'information de l'utilisateur. La fenêtre en question permet aussi à l'utilisateur de modifier ses informations de compte. Si certaines informations entrées ne correspondent pas aux contraintes de la base de données, un message d'erreur sera affiché au client. Aussi, si la page du compte client est soudainement fermée après avoir changé certains champs et sans avoir effectué une sauvegarde, aucun changement ne sera apporté dans la base de données. La figure ci-dessous donne un aperçu de la fenêtre du compte utilisateur.

The screenshot shows a window titled "Fenetre du Compte" with a blue header bar. The window contains four sections of user information, each with a title and a form:

- INFO COMPTE**:
 - Identifiant: 997690
 - Courriel: RobertCFlores21@gmail.com
 - Mot de Passe: eishie3meiH
- INFO PERSONNEL**:
 - Prenom: Robert
 - Nom Famille: Flores
 - Telephone: 604-897-3621
 - Anniversaire: 1953-07-22
 - Forfait: D
- ADRESSE**:
 - Adresse: 3380 Glover Road
 - Ville: Langleyyy
 - Province: BC
 - Code Postal: V3A 4P6
- INFO CREDIT**:
 - Carte: MasterCard
 - Numéro: 5567 9361 3607 8358
 - Exp-Mois: 5
 - Exp-Annee: 2011

At the bottom of the window are two buttons: "Update" and "Cancel".

Lorsque l'utilisateur appuie sur le bouton de mise à jour de l'information, l'application demande à l'utilisateur de confirmer son choix. Si l'utilisateur appuie sur le bouton d'annulation (Cancel), aucun changement ne sera sauvegardé dans la base de données.

Pour ce qui est de la page principale de l'application, on retrouve sous la barre de l'entête de l'application deux onglets. Le premier onglet ("Recherche Film") porte sur la recherche de films et de personnes. Le deuxième onglet porte sur la recherche de contenu de films ("Recherche contenue") orienté vers la recherche des métadonnées.

Si on sélectionne le premier onglet de la page principal, on perçoit vers le haut la recherche par personne en tout genre d'un film ainsi que la recherche spécifique par réalisateur, scénariste et acteur. La **recherche d'une personne** en tout genre supporte la recherche mixte entre le nom et l'identifiant d'une personne. Si seulement un des deux champs est utilisé, le champ vide sera ignoré lors de l'exécution de la requête vers la base de données.

La **recherche spécifique** par réalisateur, scénariste ou acteur ne supporte pas la recherche mixte. En effet, si l'utilisateur écrit des caractères dans un champ et qu'il se met à écrire soudainement dans un autre champ de la même section, l'application supprimera les caractères de l'autre champ. La raison est simple, l'application cherche à effectuer une recherche spécifique pour un type de personne. En d'autres mots elle exécutera la requête pour seulement un des trois champs. La figure ci-dessous illustre cette première section de la page principale de l'application qui consiste en la recherche par personne.

Recherche Film		Recherche Contenu
PERSONNE		RESULTATS - PERSONNES Jack Klugman Jack Crocicchia Jack McBrayer Jack Hawkins Jackie Cooper
Nom	jack	
Identifiant	14	
Search Personne		
PARTICIPANT		
Réalisateur		
Scénariste		
Acteur		
Search Participant		

Si l'utilisateur double clic sur un des résultats de la recherche par personne, une nouvelle fenêtre s'ouvrira et affichera la **fiche-personne** qui contient toute l'information sur la personne sélectionnée. La figure de droite illustre la fiche-personne.

Si la fiche de cette personne contient du contenu multimédia, l'utilisateur pourra cliquer sur le lien et l'application affichera le contenu. La lecture des données de type CLOB a été possible grâce à une méthode particulière mise en place dans l'application. En d'autres mots, l'application supporte les textes allant jusqu'à deux milliards de caractères.

The screenshot shows a window titled "Personne: Steven Soderbergh". It contains the following information:

- Personne:**
 - ID: 1752
 - Nom: Steven Soderbergh
 - Photo: [Lien au Média](#)
- Naissance:**
 - Anniversaire: 1963-01-14 00:00:00.0
 - Lieu: Atlanta, Georgia, USA
- Biographie:**

Steven Soderbergh was born on January 14, 1963, in Atlanta, Georgia, USA, the second of six children, and while still at a very young age, his family moved to Baton Rouge, Louisiana, where his father, Peter Soderbergh, was a professor and the dean of the College of Education at Louisiana State University. While still in high school, around the age of 15, Soderbergh enrolled in the university's film animation class and began making short 16-millimeter films with second-hand equipment, one of which was the short film "Janitor." After graduating high school, he went to Hollywood, where he worked as a freelance editor. His time there was brief, and shortly after, he returned home and continued making short films and writing scripts. His first major break was in 1986 when the rock group Yes assigned him to shoot a full-length concert film for the band, which eventually earned him a Grammy nomination for the video Yes: 9012 Live (1985) (V). Following this achievement, Soderbergh filmed Winston (1987), the short-subject film that he would later expand into Sex, Lies, and Videotape (1989), a film that earned him the Cannes Film Festival's Palme d'Or Award, the Independent Spirit Award for Best Director, and an Oscar nomination for Best Original Screenplay. Over the next six years, he was married to actress Betsy Brantley and had a daughter named Sarah who was born in 1990. Also during this time, he made such films as Kafka (1991), King of the Hill (1993), Underneath (1995), and Gray's Anatomy (1996), which many believed to be disappointments. In 1998, Soderbergh made Out of Sight (1998), his most critically and commercially successful film since Sex, Lies, and Videotape (1989). Then, in

Par la suite, au milieu de la page principale, on retrouve la section de **recherche film**. Pour cette section, l'application offre une recherche simple et une recherche avancée. La **recherche simple** en question supporte la recherche mixte avec plusieurs champs et ne prendra pas compte des champs vides lors de l'exécution d'une requête comme la recherche simple d'une personne discutée un peu plus tôt.

De plus, vous constaterez que lorsqu'il y a plus d'un film dans la section des résultats films, l'application active le bouton d'**exportation XML** ("Export XML"). Lorsque ce bouton est sélectionné, l'application exportera la totalité de l'information des films situés dans la section des résultats films dans un fichier de type XML. Le fichier en question respecte les standards du XML et sera situé dans le dossier *xmlExport* de l'application. La figure ci-dessous illustre la section de recherche simple de films.

The screenshot shows a window titled "Recherche" with two tabs: "Recherche" and "Recherche avancée". The "Recherche" tab is active, showing a "FILM" section with the following search criteria:

- Titre: act
- Langue:
- Résumé:
- Année: 19
- Duée:
- Genre: ☐ Action

At the bottom of the search criteria are two buttons: "Search Film" and "Export.xml". To the right of the search criteria is a section titled "RESULTATS - FILMS" containing a list of film titles:

- Deep Impact
- Contact
- Fatal Attraction
- Sister Act

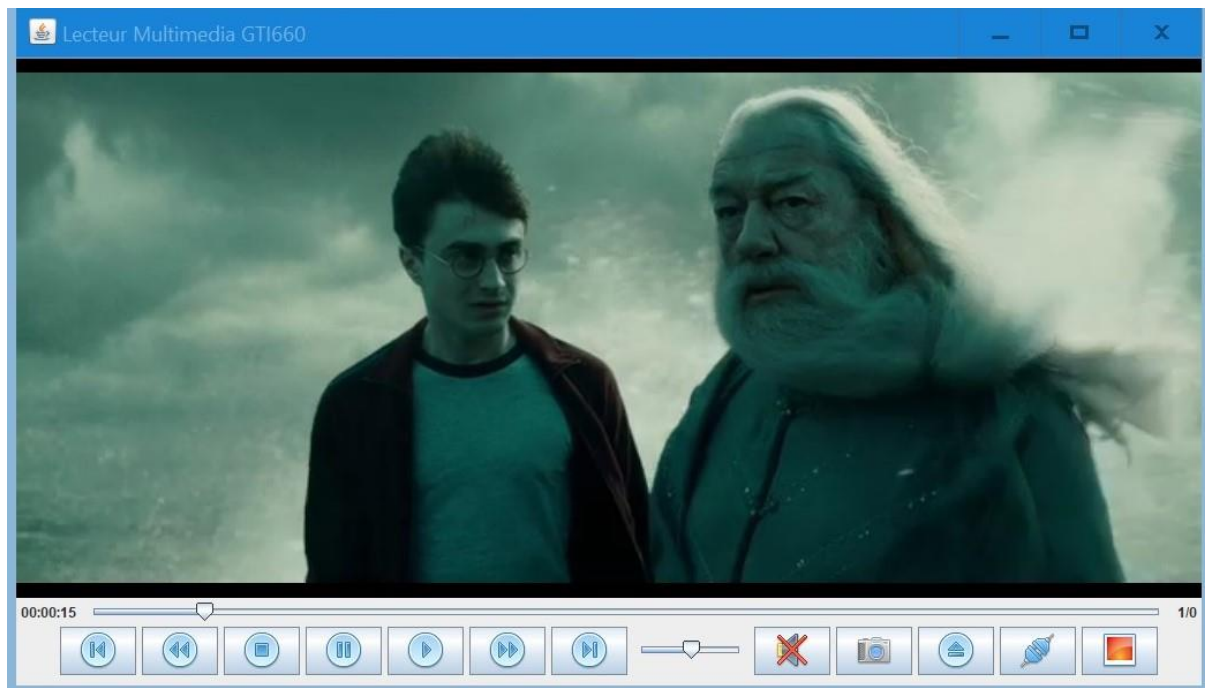
Pour la **recherche avancée**, elle est orientée de façon à offrir la possibilité à l'utilisateur d'effectuer une recherche entre deux valeurs. Par exemple, l'utilisateur pourrait vouloir trouver tous les films entre les années 60 et 70. Les résultats obtenus sont dirigés vers le même panneau des résultats de la recherche simple de film. Pour chaque recherche simple ou spécialisée, la liste qui contient les résultats film est réinitialisée. La figure ci-dessous illustre la recherche avancée de films.

Tout comme la recherche par personne, si l'utilisateur double clique sur un des résultats obtenus de la recherche de film, la **fiche film** sera affichée dans une nouvelle fenêtre. Non seulement la fenêtre affichera l'information de type *VARCHAR* et *XMLType* de la base de données, elle affiche aussi la couleur dominante du "poster" du film sélectionné ainsi que ses valeurs RGB.

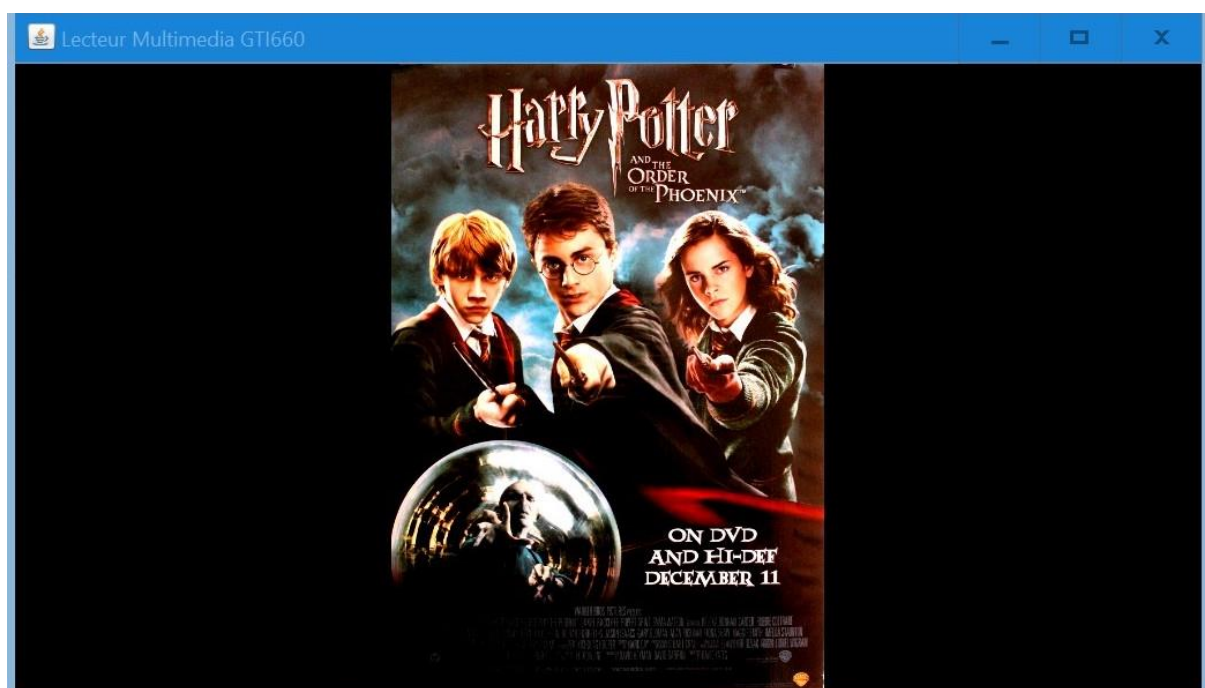
Si la fiche du film contient du contenu multimédia, l'utilisateur pourra cliquer sur le lien et l'application affichera l'image dans la fenêtre de lecture de contenu multimédia grâce au framework *VLCJ* mis en place. La figure de droite illustre la fiche film en question.

La figure ci-dessous illustre la fenêtre de **lecture de contenu multimédia** de l'application. Comme il a été mentionné un peu plus tôt, la lecture vidéo aura le panneau de contrôle au bas de l'écran, ce qui ne sera pas le cas avec la lecture d'image. La fenêtre de lecture vidéo

supporte la mise en pause, l'arrêt, le déplacement du curseur temporel, le retour à l'état initial lorsque la vidéo termine la lecture, le contrôle du volume, etc.



Il faut noter qu'un système de détection des erreurs *HTTP* a été mis en place afin de détecter les liens de contenu multimédia invalide. De cette façon, l'utilisateur ne se ramassera jamais avec une fenêtre de contenu multimédia noir sans contenu. En d'autres mots, lorsque l'utilisateur cliquera sur un lien multimédia qui ne fonctionne pas, l'application va afficher une erreur 404, 403, etc. Vidéo ou image, c'est la même gestion d'erreurs.



Pour ce qui est de la troisième partie de la page principale du premier onglet, elle permet à l'utilisateur de gérer la location de films, effectuer une recherche mixte entre une personne et un film et afficher l'historique de recherche effectuée par l'utilisateur.

Concernant la **location de films**, on constate trois boutons au bas de la fenêtre. Le bouton de location ("Louer Film") et le bouton de retrait ("Retirer Film") permettent à l'utilisateur de garder en mémoire certains films de la base de données et de les gérer. En d'autres mots, l'utilisateur peut retirer ou associer des films à son compte personnel. Ce qui veut dire qu'à l'ouverture de l'application ou d'une nouvelle session, l'utilisateur pourra retrouver les films qu'il avait gardés en mémoire dans la section location lors d'une session antérieure. Puisque chaque film est associé à un prix, l'application affiche dans le panneau des films loués le solde et la quantité de films loués.

Bref, pour louer un film, l'utilisateur n'a qu'à sélectionner un des films du panneau des résultats film et cliquer sur le bouton de location. L'application activera le bouton pour louer un film seulement lorsqu'il y a des résultats dans le panneau de recherche film. Dans le même ordre d'idées, l'application activera le bouton pour retirer un film seulement si la liste des films loués n'est pas vide. Si la liste de sauvegarde n'est pas vide, l'utilisateur n'a qu'à sélectionner un de ses films et appuyer sur le bouton de retrait.

De plus, cette même section est composée d'un bouton de **recherche mixte entre une personne et un film** ("Search All") afin de trouver le titre des films avec un genre et un réalisateur ou un scénariste ou un acteur ou le nom d'une personne. Les résultats de cette combinaison seront affichés dans le panneau des résultats film. L'application activera le bouton en question seulement si un genre de la section recherche film a été sélectionné et qu'un des champs de la section recherche personne ne possède des caractères alphabets.

Pour permettre à l'utilisateur de garder l'**historique** de chaque recherche effectuée durant la session, l'application affiche tous les titres des champs utilisés ainsi que les caractères entrés et la date à laquelle la recherche a été effectuée. La figure ci-dessous montre un aperçu global de la section qui regroupe la location de film, la recherche mix entre une personne et un genre et l'affichage de l'historique de recherche.

Dans un autre ordre d'idées, nous allons désormais discuter du deuxième onglet situé sous la barre de l'entête de l'application qui porte sur la recherche film à partir de son contenu XMLType. Cette section est majoritairement composée de deux grandes sections. La première section permet d'effectuer des recherches basées sur les prix qu'ont gagnés les films ainsi que leurs classements. La deuxième section consiste à la recherche des films qui ont un "poster" similaire à une image importée.

The screenshot displays two main panels. The left panel, titled 'HISTORIQUE RECHERCHE', contains a scrollable list of search entries. Each entry shows the search criteria (e.g., 'Titre: cars', 'Nom Réalisateur: steven', 'Année: 1960') and the time taken for the search (e.g., 'Nouvelle recherche (23:03:30)'). Below this list is a 'Search All' button. At the bottom of the left panel are two buttons: 'Louer Film' and 'Retirer Film'. The right panel, titled 'LOCATION FILMS', shows the 'Prix total des locations (3) : 37,4 \$' and a list of films: 'Indiana Jones and the Last Crusade', 'Batman & Robin', and 'Cars'.

En ce qui concerne la recherche, film par **prix** et par **classement** l'application ne supporte pas la recherche mixte. En d'autres mots, l'utilisateur peut seulement chercher un film en utilisant un champ à la fois. Comme la recherche spécialisée d'une personne et la recherche avancée d'un film, lorsque l'utilisateur entre des caractères dans un champ et qu'il se met à écrire dans un autre champ de la même section, l'autre champ sera vidé. Il y a cependant une exception, si le champ "semaine" et le champ "rang" ne sont pas vides, une requête différente sera exécutée pour effectuer une recherche basée sur ces deux informations. La figure ci-dessous illustre la section de films par prix et par classement.

Recherche Film		Recherche Contenu	
AWARD		RESULTATS - AWARDS	
Catégorie	<input type="text" value="Animation"/>	<div>Cars Harry Potter and the Prisoner of Azkaban Harry Potter and the Order of the Phoenix Harry Potter and the Goblet of Fire</div>	
Winner Price	<input type="text"/>		
Année Prix	<input type="text"/>		
<input type="button" value="Search Awards"/>			
CLASSEMENT		RESULTATS - CLASSEMENT	
Rang	<input type="text" value="1"/>	<div>George of the Jungle Cars Harry Potter and the Half-Blood Prince Harry Potter and the Prisoner of Azkaban Batman & Robin Jurassic Park III Harry Potter and the Chamber of Secrets Harry Potter and the Order of the Phoenix Harry Potter and the Sorcerer's Stone Harry Potter and the Goblet of Fire Jurassic Park</div>	
Semaine	<input type="text"/>		
Mois	<input type="text"/>		
<input type="button" value="Search Classement"/>			

Pour tous les films trouvés par l'application dans les panneaux de résultats de films, l'utilisateur peut en tout temps double cliquer sur un film pour visualiser la fiche complète du film et vérifier les résultats de recherches.

Pour ce qui est de l'autre moitié de la recherche par contenu, il sera question de permettre à l'utilisateur d'effectuer une comparaison au niveau des **couleurs primaires** des "posters" film. Pour assurer le bon fonctionnement, plusieurs fonctionnalités ont été mises en place.

Dans un premier temps, il est possible pour l'utilisateur d'ajuster manuellement le niveau de tolérance des valeurs RGB. Par exemple, si la valeur rouge d'une couleur est de 30 et que la tolérance est initialisée à 10, l'application va chercher les valeurs rouges entre 20 et 40. En d'autres mots, l'application va chercher les valeurs de plus ou moins 10 par rapport aux valeurs RGB identifiées.

De plus, l'utilisateur peut insérer des valeurs rouge, verte et bleu dans les champs de la couleur dominante pour avoir un aperçu de la couleur et effectuer rapidement des recherches sans même avoir importé une image. Il faut spécifier que l'application contraint l'utilisateur à entrer des valeurs seulement entre 0 et 255. Comme d'habitude, pour effectuer une recherche, le champ de tolérance et les champs R,G et B ne peuvent être vides. Le bouton de recherche sera activé seulement si tous les champs sont remplis. La figure ci-dessous donne un exemple de la situation.


The figure displays two side-by-side screenshots of a web application interface titled "MAIN COLOR". Each interface has a header section with "Tolérance Couleur (+/-)" and "Couleur Dominante". Below the header, there are input fields for "Image" and "Couleur Dominante". The "Couleur Dominante" section includes RGB value inputs (R, G, B) and a color preview box. At the bottom, there are two buttons: "File Chooser" and "Search Similar Color".

Left Screenshot: The "Tolérance Couleur (+/-)" field is empty. The "Couleur Dominante" section shows R: 50, G: 129, B: 150, and a corresponding blue color preview.

Right Screenshot: The "Tolérance Couleur (+/-)" field contains the value "10". The "Couleur Dominante" section shows R: 50, G: 129, B: 150, and a corresponding blue color preview.

Sinon, l'utilisateur peut simplement importer une image de format ".JPG" et l'application va automatiquement calculer et afficher la couleur dominante ainsi que ses valeurs RGB.

Effectuons un test pour montrer le bon fonctionnement de la recherche par couleur dominante. Si l'utilisateur sélectionne le bouton de gauche ("File Chooser") pour importer une image (JPG) sauvegardée localement sur son poste de travail, il peut rapidement avoir un aperçu de la couleur dominante sans oublier les valeurs RGB. Puisque tous les champs sont remplis, le bouton de recherche s'active et peut exécuter la recherche. Dans la figure ci-dessous, on constate que l'application a identifié deux résultats de films.

MAIN COLOR		RESULTATS - COLOR
Tolérance Couleur (+/-)	Couleur Dominante	Harry Potter and the Half-Blood Prince Batman & Robin
<input type="text" value="10"/>	R <input type="text" value="50"/> G <input type="text" value="48"/> B <input type="text" value="61"/>	
Image	Couleur Dominante	
		
<input type="button" value="File Chooser"/>	<input type="button" value="Search Similar Color"/>	

Si le client double clique sur le film portant le titre "Harry Potter and the Half-Blood Prince" dans le panneau des résultats, l'application va afficher la fiche complète du film ainsi que l'information concernant la couleur dominante du "poster" "Harry Potter". Dans la figure ci-dessous, on montre une partie de la fiche film qui mon l'information utile à la comparaison. En effet, l'utilisateur pourra constater que la couleur du "poster" a une différence de 11,42% avec celle de l'image importée de "Batman". Aussi, il pourra confirmer que les valeurs RGB de l'image importée (50, 48,61) respectent bien la tolérance de plus ou moins 10 avec les valeurs RGB du "poster" (40, 52,57). La figure ci-dessous représente une partie de la fiche film du film "Harry Potter".

Catégorie:	Top Box Office Films, Best Villain, Best Family Film, Best Live Action Family Film, Best DVD Release, Best Film		
Poster color			
RGB Primaire:	R: 40	G: 52	B: 57
	Différence : 11,42 %		

11. CONCLUSION

En résumé, l'objectif de ce laboratoire était de concevoir une application de gestion de base des données multimédia, plus précisément il s'agit d'un système de location de films tout en respectant l'architecture trois tierce. Le point de départ était d'analyser les exigences pour déterminer les besoins spécifiques de l'application, les fonctionnalités à ajouter comme consulter une vidéo en streaming à l'aide du lecteur multimédia vlc intégré dans l'interface ainsi que des nouvelles fonctionnalités de recherche multimédia et l'utilisation des patrons de conception.

Par la suite, la conception a permis d'avoir une visibilité sur la composition de l'application à travers l'architecture, le diagramme de classes et d'apporter des solutions telles que l'intégration du lecteur multimédia VLC et la description du contenu multimédia de type «XML Type» ainsi que l'intégration du Descripteur visuel MPEG7.

Puis, l'implémentation consacre la réalisation de la conception, l'intégration de l'algorithme principale, l'intégration du descripteur visuel MPEG7, l'intégration des modules de lecture et de recherche multimédia ainsi que les interfaces les plus significatives de l'application.

En ce qui concerne l'application, elle respecte les exigences de qualité telles que la performance, la convivialité, la fiabilité et la maintenabilité par exemple le code est bien commentées et modulaires par conséquent il est facile à maintenir et extensible.

Lors de l'implémentation, plusieurs difficultés ont été surmontées, le premier choix de la technologie était spring boot lors du deuxième laboratoire, très vite un problème de conversion des données s'est posé et cette piste a été abandonnée pour revenir en java. En plus, les compétences des membres de l'équipe sont diverses, tout le monde n'est pas à l'aise avec la programmation, mais chacun a apporté sa contribution en fonction de ses compétences d'où la réussite du laboratoire.

12. RÉFÉRENCES

<https://sqbd.developpez.com/tutoriels/cours-complet-bdd-sql/?page=algebre-relationnelle#LVI-E>

Infinite Hidden Markov Models for Unusual-Event Detection in Video, Iulian Pruteanu-Malinici et Al.

Thèse doctorale de Sebastien Lefèvre sur séquence de vidéo

[Un Algorithme robuste de Segmentation vidéo pour des Applications Temps réels](#)