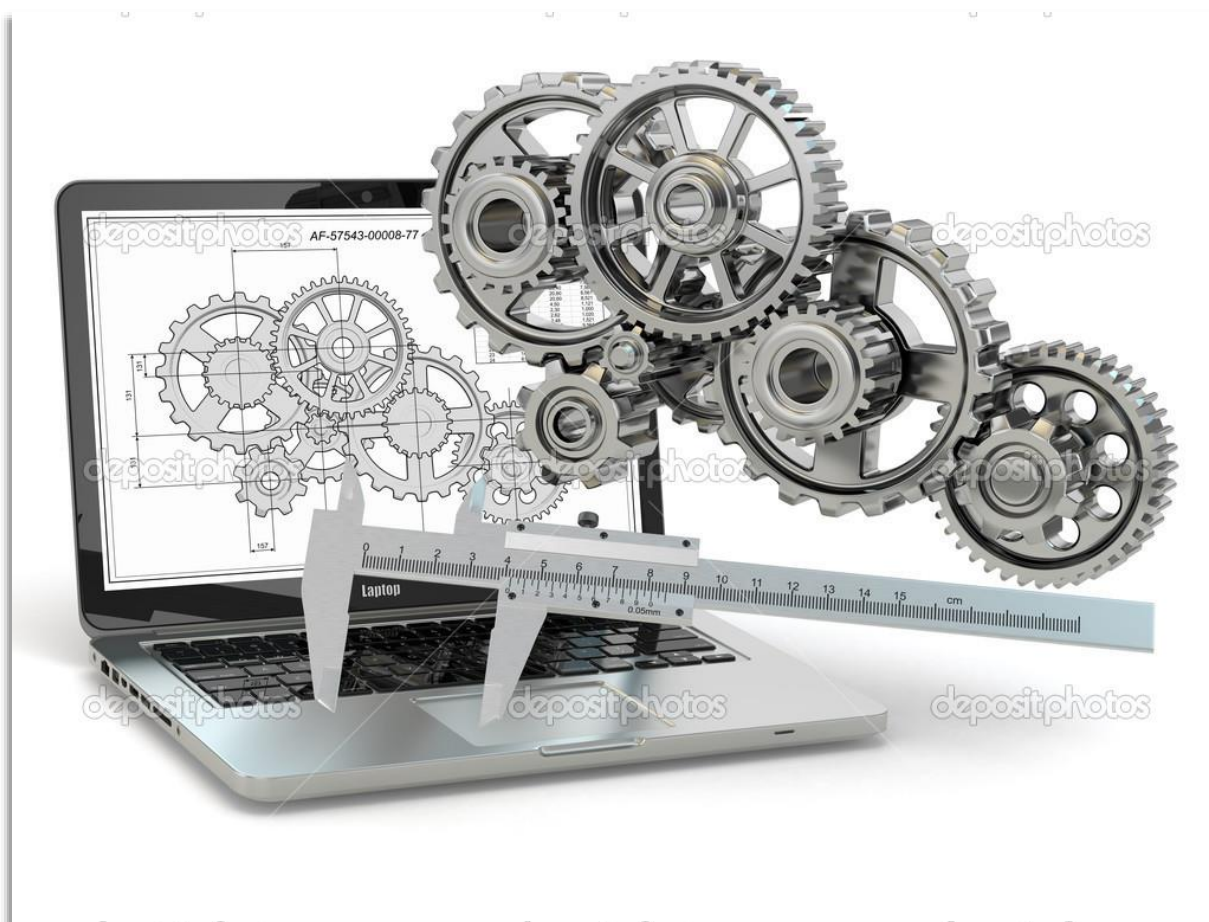


# Gestion audio



Eric-Nicolas Preisig / CIN4A  
ETML - Lausanne  
110 heures

## **Chef de Projet**

M. Bertrand Sahli – [bertrand.sahli@vd.ch](mailto:bertrand.sahli@vd.ch)

## **Experts**

M. Sylvain Rollinet – [sylvain.rollinet@gmail.com](mailto:sylvain.rollinet@gmail.com)  
M. Jonathan Melly – [jonathan.melly@vd.educanet2.ch](mailto:jonathan.melly@vd.educanet2.ch)

# Table des matières

<b>1</b>	<b>SPÉCIFICATIONS.....</b>	<b>3</b>
1.1	TITRE .....	3
1.2	DESCRIPTION .....	3
1.3	MATÉRIEL ET LOGICIELS À DISPOSITION .....	3
1.4	PRÉREQUIS.....	4
1.5	CAHIER DES CHARGES .....	4
1.6	LES POINTS SUIVANTS SERONT ÉVALUÉS .....	4
1.7	VALIDATION ET CONDITIONS DE RÉUSSITE .....	4
<b>2</b>	<b>PLANIFICATION INITIALE.....</b>	<b>5</b>
<b>3</b>	<b>ANALYSE.....</b>	<b>5</b>
3.1	OPPORTUNITÉS .....	5
3.2	DOCUMENT D'ANALYSE ET CONCEPTION .....	7
3.2.1	<i>Maquette.....</i>	<i>7</i>
3.2.2	<i>Architecture .....</i>	<i>12</i>
3.2.1	<i>Librairie utilisées.....</i>	<i>13</i>
3.2.2	<i>Structure du code .....</i>	<i>13</i>
3.2.3	<i>Base de données.....</i>	<i>16</i>
3.2.1	<i>Schéma évènementiel.....</i>	<i>17</i>
3.3	CONCEPTION DES TESTS .....	24
3.4	PLANIFICATION DÉTAILLÉE .....	24
<b>4</b>	<b>RÉALISATION .....</b>	<b>24</b>
4.1	DOSSIER DE RÉALISATION.....	24
4.2	MODIFICATIONS.....	24
<b>5</b>	<b>TESTS.....</b>	<b>24</b>
5.1	DOSSIER DES TESTS .....	24
<b>6</b>	<b>CONCLUSION .....</b>	<b>24</b>
6.1	BILAN DES FONCTIONNALITÉS DEMANDÉES .....	24
6.2	BILAN DE LA PLANIFICATION.....	25
6.3	BILAN PERSONNEL .....	25
<b>7</b>	<b>DIVERS.....</b>	<b>25</b>
7.1	JOURNAL DE TRAVAIL .....	25
7.2	BIBLIOGRAPHIE .....	25
7.3	WEBOGRAPHIE .....	25
<b>8</b>	<b>ANNEXES .....</b>	<b>26</b>
8.1	CAHIER DES CHARGES .....	26
8.2	GANTT .....	29

# 1 SPÉCIFICATIONS

Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le 1er jour.

Le cahier des charges est approuvé par i-CQ VD. Il est en outre présenté, commenté et discuté avec le candidat. Par sa signature, le candidat accepte le travail proposé.

Le candidat a connaissance de la feuille d'appréciation avant de débiter le travail.

Le candidat est entièrement responsable de la sécurité de ses données.

En cas de problèmes graves, le candidat avertit au plus vite les deux experts et son chef de projet.

Le candidat a la possibilité d'obtenir de l'aide, mais doit le mentionner dans son dossier de projet.

A la fin du délai imparti pour la réalisation du TPI, le candidat doit transmettre par courrier électronique le dossier de projet aux deux experts et au chef de projet. En parallèle, une copie papier du rapport doit être fournie sans délai en trois exemplaires. Cette dernière doit être en tout point identique à la version électronique.

## 1.1 Titre

Gestionnaire de playlists audio et de flux radios

## 1.2 Description

Développer une application desktop en C# selon le pattern MVVM afin de gérer des playlists audio et des flux radios publics diffusés sur Internet. Il s'agira notamment :

- d'assurer l'installation de l'application sur les OS Microsoft 7 à 10.
- d'exploiter les données audios locales (sans les dupliquer).
- de créer un lecteur audio « user friendly ».
- de stocker les données de l'application dans une base de données SQLite.

## 1.3 Matériel et logiciels à disposition

PC standard de l'ETML (Windows 7 – 64bit)

Player de machines virtuelles (vmware ou équivalent)

Suite Microsoft Office 2013

IDE C# (Visual Studio Community 2015 ou 2017, SQLite Precompiled Binaries for Windows)

## 1.4 Prérequis

Compétences élémentaires en base de données (SQL) et en programmation (C#) → Modules 103, 104, 105, 303.

Compétences bureautiques → Modules 301 et 302

## 1.5 Cahier des charges

Cahier des charges complet en annexe

## 1.6 Les points suivants seront évalués

- La méthodologie de travail ainsi que la conformité des tâches réalisées par rapport au planning initial.
- Le choix des « métadonnées » nécessaires à l'indexation des chansons.
- La pertinence du modèle de données réalisé.
- La mise en œuvre de tests (notamment les tests unitaires) pour vérifier les fonctionnalités implémentées.
- La journalisation des tâches ainsi que la rédaction régulière de la documentation.

## 1.7 Validation et conditions de réussite

La grille d'évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, ...).

En plus de cela, le travail sera évalué sur les trois points spécifiques suivants :

- Le point spécifique n° 1 → L'extraction de métadonnées de divers format audio.
- Le point spécifique n° 2 → La recherche de WebRadio.
- Le point spécifique n° 3 → Le (re)scan des contenus audios.

## 2 PLANIFICATION INITIALE

<b>Date Début :</b>	lundi 1 mai 2017
<b>Date Fin:</b>	mercredi 7 juin 2017
<b>Nombre de semaines:</b>	6
<b>Nombre de Périodes/Semaine :</b>	33
<b>Nombre de 1/4 heure/Période:</b>	3

Liste des tâches	
<b>Tâche obligatoire:</b>	Absence - Imprévu
<b>1</b>	Planification initiale
<b>2</b>	Arborescence de fonctions et de classes
<b>3</b>	Analyse
<b>4</b>	Création du MPD
<b>5</b>	Création de la base de données
<b>6</b>	Mise en place de l'architecture MVVM
<b>7</b>	Création des tests unitaires
<b>8</b>	UI
<b>9</b>	Création des objets DTO (Data transfer object)
<b>10</b>	Couche BLL (Business Logic Layer)
<b>11</b>	Couche Présentation
<b>12</b>	Couche DAL (Data Access Layer)
<b>13</b>	Création de l'installateur
<b>14</b>	Tests
<b>15</b>	Manuel utilisateur
<b>16</b>	Documentation
<b>17</b>	Rendez-vous
<b>18</b>	Congé
<b>19</b>	Bugs

*Gantt disponible en annexe*

## 3 ANALYSE

### 3.1 Opportunités

Ce projet me permettra de développer mes capacités en développement .NET. Il me donnera l'opportunité d'approfondir mes connaissances dans l'architecture MVVM.

La principale difficulté sera de fournir une interface simple et ergonomique. Je pense donc partir sur un design flat et minimaliste, permettant ainsi une interface simple et intuitive. J'utiliserais pour cela la librairie Mahapps qui me permettra de fournir une interface épurée dans le style de Windows 8

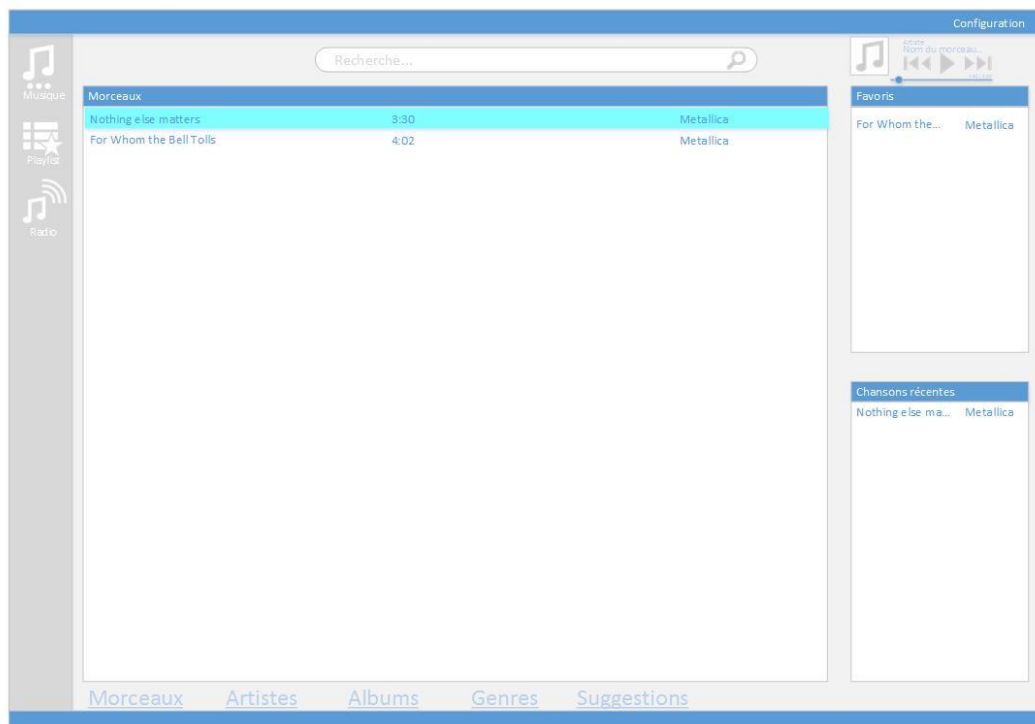
La sauvegarde du contexte de l'application sera un point difficile. Il y a plusieurs options pour pouvoir garder le contexte applicatif après la fermeture de l'application. Je pense créer une classe qui contient toutes les données à sauvegarder. Celle-ci sera enregistrée dans la base de

données. À l'ouverture de l'application je n'aurais alors plus qu'à interpréter les données de la classe pour les restaurer au bon endroit.

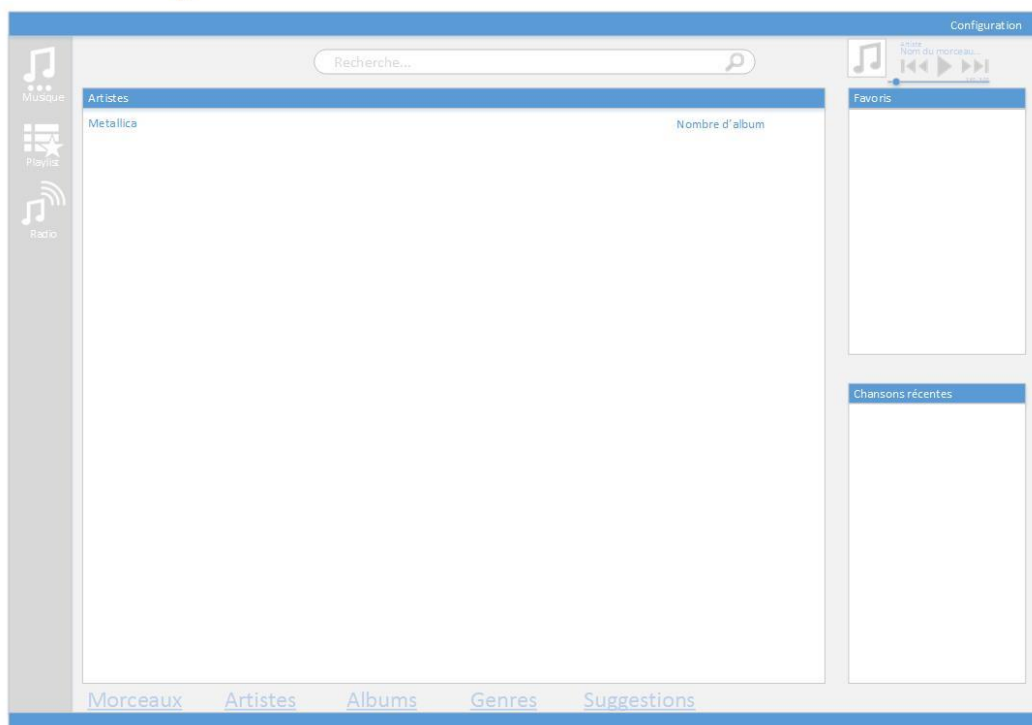
## 3.2 Document d'analyse et conception

### 3.2.1 Maquette

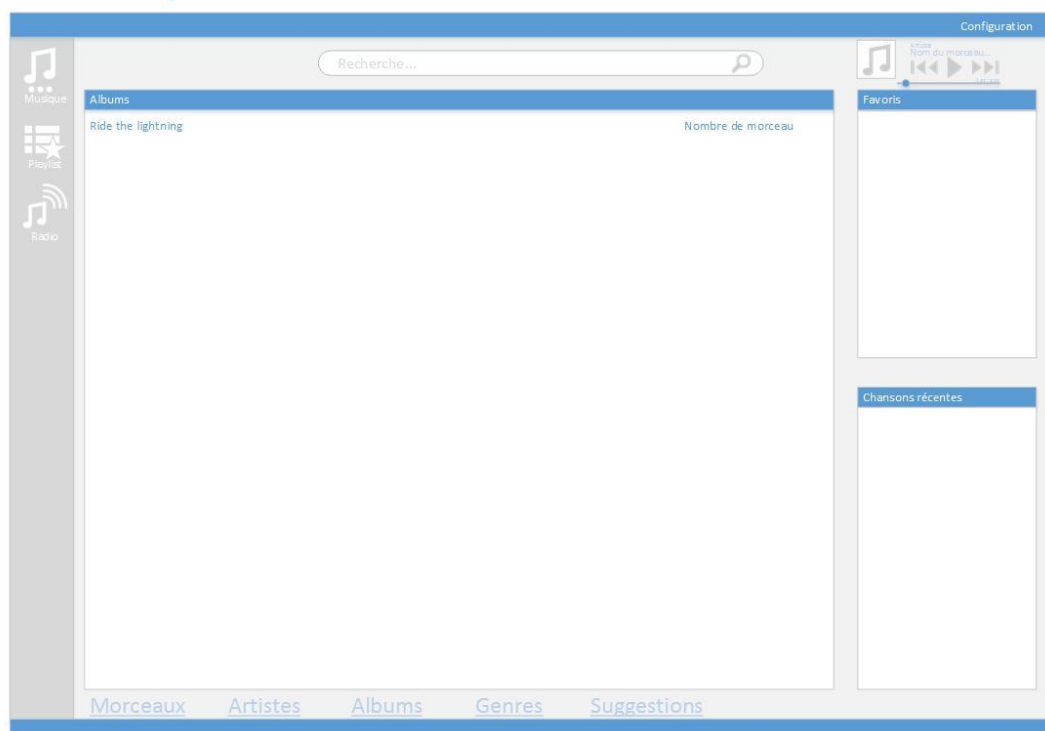
## Musique



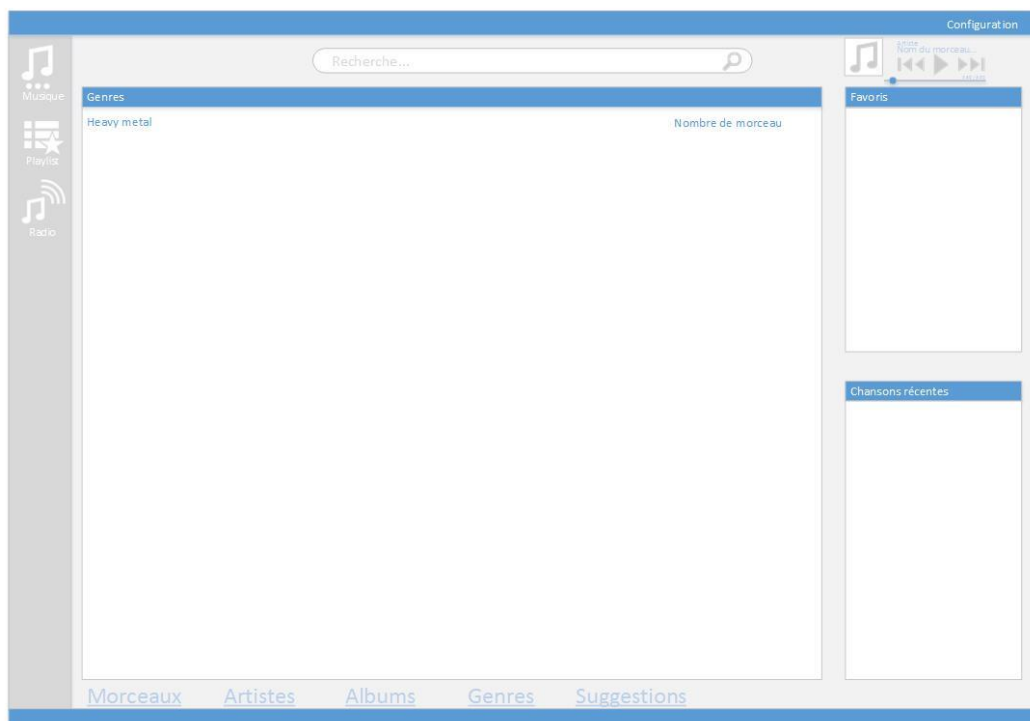
## Musique - Artistes



## Musique – Album

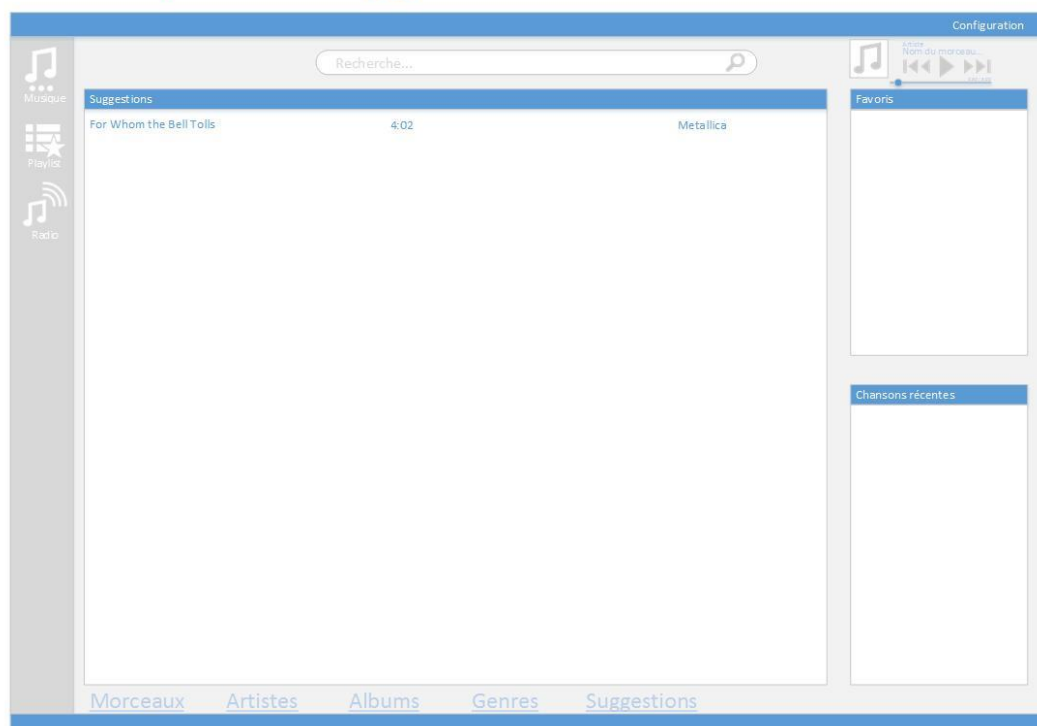


## Musique – Genres

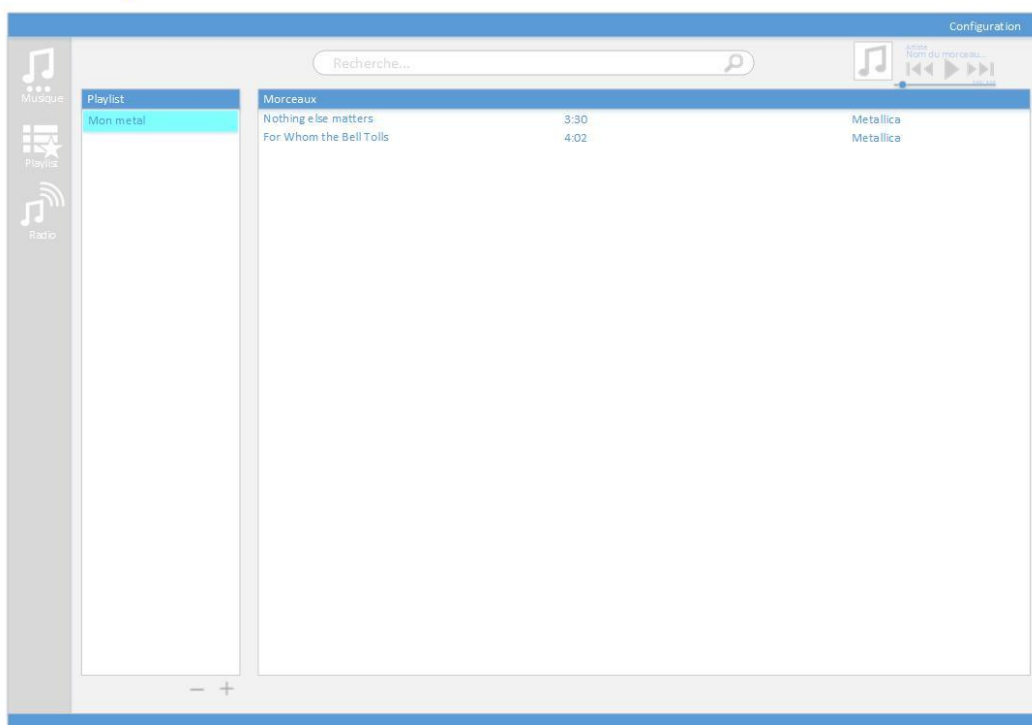




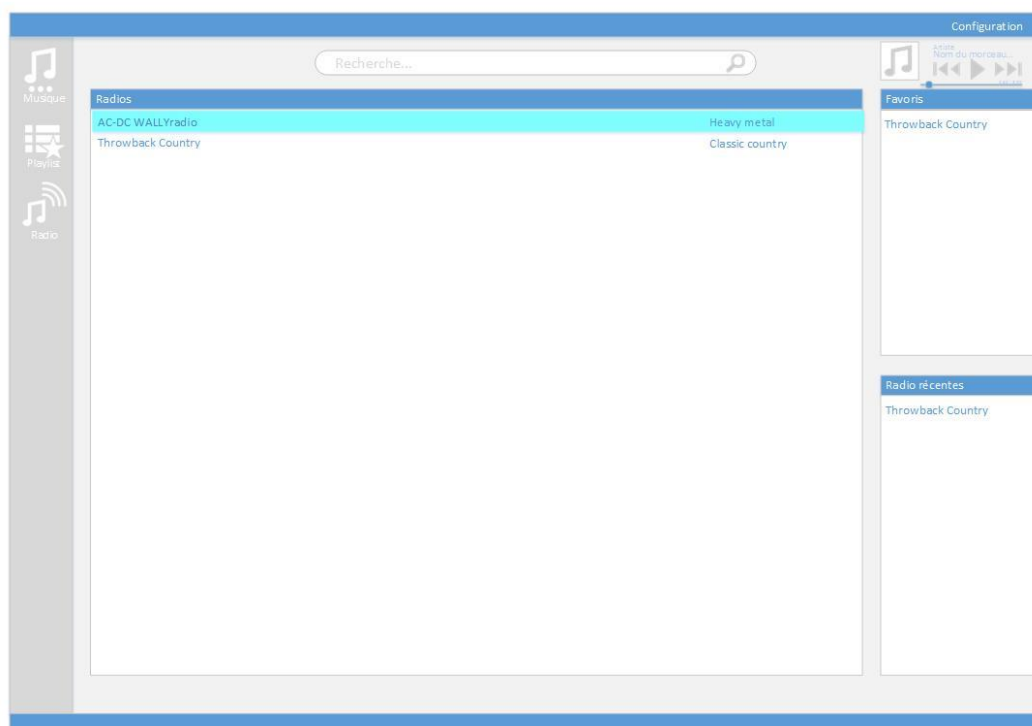
## Musique – Suggestions



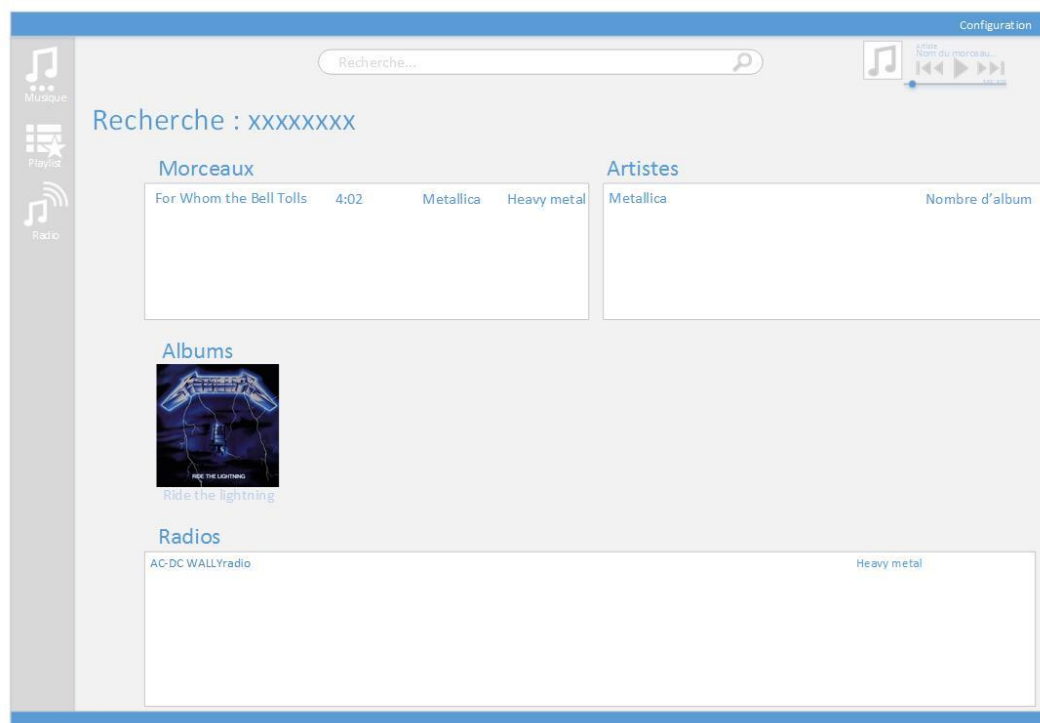
## Playlist




## Radio



## Recherche



## Morceau en cours



Ride the lightning  
Metallica  
Heavy metal  
1983

For Whom the Bell Tolls

Navigation controls: Previous, Play/Pause, Next, Repeat, Shuffle, and a progress bar.

### Configuration

Lancer la musique au démarrage



Synchroniser la musique

Dossiers exclus

— +

Annuler

Enregistrer

### Propriétés



[Sélectionner une image de couverture](#)

Artiste

Album

Date

Genre

Annuler

Enregistrer

### Liste de lecture

For Whom the Bell Tolls



### 3.2.2 Architecture

Je vais utiliser une architecture en multicouche d'une profondeur de 3 couches. Il y aura :

La couche DAL, qui s'occupera de récupérer les données depuis la base de données ainsi que depuis l'API de Web radios.

La couche BLL, qui se chargera de formater les données brutes reçues par la couche DAL

La couche Présentation, servira d'interface humain-machine elle aura la charge d'afficher les données à l'utilisateur ainsi que de récupérer ses entrées.

Quant au design paterne, j'utiliserai MVVM. Les view et les viewModel seront inclus dans la couche de Présentation. Le model sera l'ensemble de la couche DAL et BLL combiné.

Une partie DTO sera communes à toutes les couches, elle aura pour rôle de fournir le model des objets. Si une méthode est commune à toutes les couches, elle pourra être placée dans la zone « Méthode commune »

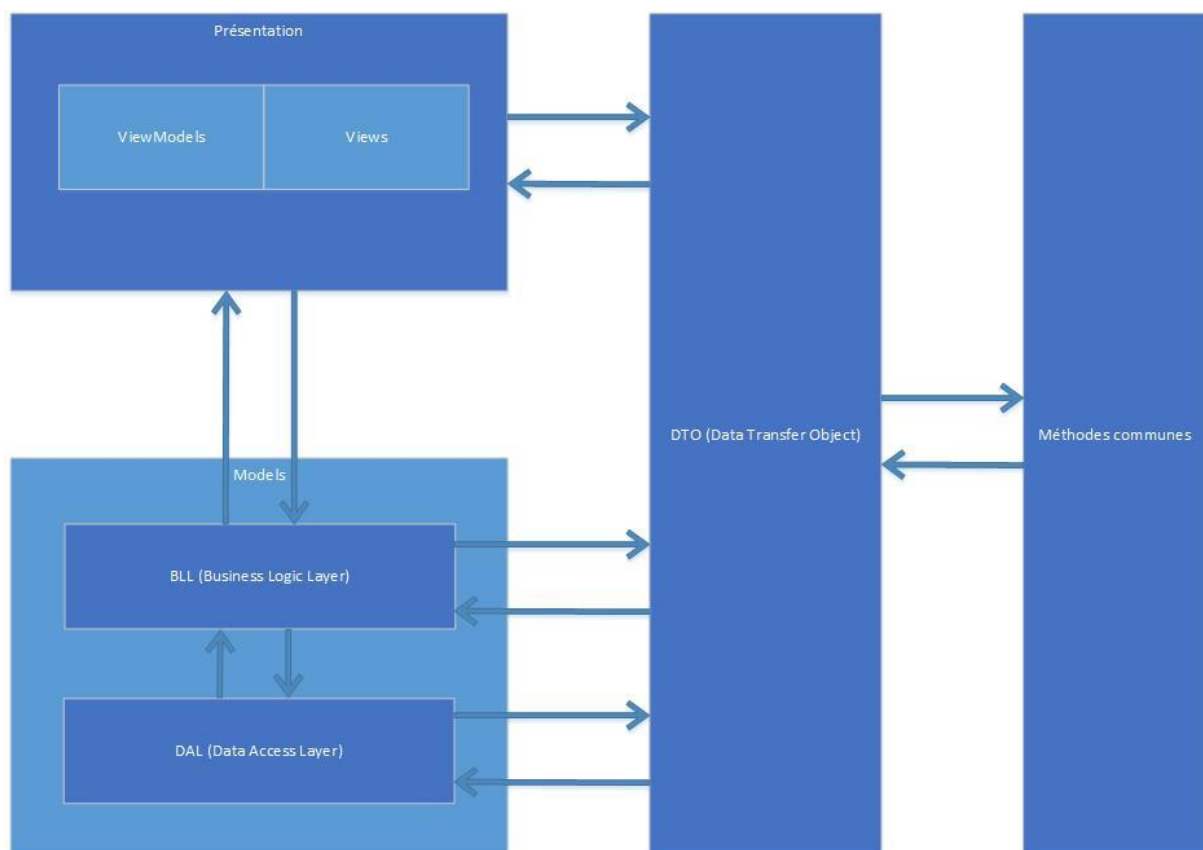


Figure 1 Illustration de l'architecture

### 3.2.1 Librairie utilisées

#### **API ShoutCast**

ShoutCast est une plateforme de radio amateur qui propose plus de 50 000 radios. Elle propose aussi une API que j'utiliserais pour gérer la partie radio de l'application

#### **MvvmLight**

Permet la simplification des fonctionnalités propre à MVVM, comme le « Messenger », les « Command », et les mise à jour de la vue

#### **EntityFramework**

ORM

#### **Mahapps.Metro**

Ajoute des éléments graphiques pour rendre l'interface plus moderne

#### **NAudio**

Permet la lecture des fichiers et flux audio

#### **NUnit**

Ajoute des fonctionnalités aux tests unitaire

#### **System.Data.SQLite**

Permet l'utilisation d'une base de données SQLite. Cette librairie inclus les binaires précompilés

#### **SQLite.CodeFirst**

Permet d'utiliser les actions de EntityFramework avec une base de données SQLite

### 3.2.2 Structure du code

Le code sera structuré de manière suivante :

#### **1. DAL** *Projet Data access layer*

##### **1.1. API** *Dossier*

**1.1.1. Shoutcast** *Contient toute les fonctions concernant l'API de ShoutCast*

##### **1.2. Database** *Dossier*

- 1.2.1. Configuration** Contient la configuration de la base de données, ainsi que le SEED
  - 1.2.2. DbContext** Crée un contexte de la base de données en utilisant une connexion string
  - 1.2.3. Repository** Class générique permettant d'effectuer les actions de base sur la base de données (CRUD)
- 2. BLL** *Projet Business logic layer*
  - 2.1. AlbumData** Permet d'effectuer des actions sur les données relatives aux albums
  - 2.2. ArtistData** Permet d'effectuer des actions sur les données relatives aux artistes
  - 2.3. RadioData** Permet d'effectuer des actions sur les données relatives aux radios
  - 2.4. TrackData** Permet d'effectuer des actions sur les données relatives aux morceaux
  - 2.5. SearchData** Permet d'effectuer des actions de recherches
  - 2.6. FavoriteData** Permet d'effectuer des actions sur les données relatives aux favoris
  - 2.7. GeneralData** Permet d'effectuer des actions sur les données relatives aux fonctionnalités générales de l'application
- 3. Presentation** *Projet Presentation*
  - 3.1. Helper** *Dossier contenant les classes utiles à la logique de l'application*
    - 3.1.1. Context** Gère le contexte applicatif du lecteur, contiendra par exemple la liste de lecture
    - 3.1.2. MusicPlayer** Fonctions relative au lecteur de musique (play, pause, etc)
    - 3.1.3. MusicSync** Fonctions relative à la synchronisation des pistes musicales depuis l'ordinateur
    - 3.1.4. RightClick** Fonctions relative au clic droit
  - 3.2. View** *Dossier contenant les vues de l'application*
    - 3.2.1. Flyout** *Dossier contenant les flyout de l'application*
      - 3.2.1.1. MusicFlyoutView** Flyout qui contiendra une liste soit d'album, soit d'artistes, soit de morceaux
      - 3.2.1.2. PropertyFlyoutView** Flyout contenant les propriétés d'un morceau
      - 3.2.1.3. ReadingFlyoutView** Flyout contenant la liste de lecture
      - 3.2.1.4. RunningFlyoutView** Flyout contenant la vue du morceau en cours
      - 3.2.1.5. SettingFlyoutView** Flyout contenant les options de l'application
    - 3.2.2. List** *Dossier contenant les listes de l'application*
      - 3.2.2.1. AlbumListView** Liste d'albums
      - 3.2.2.2. ArtistListView** Liste d'artistes
      - 3.2.2.3. FavoriteListView** Liste des favoris
      - 3.2.2.4. GenreListView** Liste des genres
      - 3.2.2.5. PlaylistListView** Liste des playlists
      - 3.2.2.6. ReadingListView** Liste des morceaux en lecture
      - 3.2.2.7. RecentListView** Liste des dernières radios écoutées
      - 3.2.2.8. SuggestionListView** Liste de suggestion
      - 3.2.2.9. TrackListView** Liste de morceaux
    - 3.2.3. MusicView** Vues de la page musique
    - 3.2.4. PlaylistView** Vues de la page playlist

- 3.2.5. RadioView** Vues de la page radio
      - 3.2.6. SearchView** Vues de la page recherche
      - 3.2.7. SmallPlayerView** Vues du petit lecteur de musique
    - 3.3. ViewModel** Dossier contenant les viewmodels
      - 3.3.1. MainViewModel** ViewModel parent de tous les autres ViewModels, il est utile en cas de partage entre ViewModels
      - 3.3.2. MainWindowViewModel** Gère l'affichage de la fenêtre principale
      - 3.3.3. MusicViewModel** Gère l'affichage de toutes les vues musique (morceaux, artistes, albums et genre)
      - 3.3.4. PlayerViewModel** Gère l'affichage du petit player, ainsi que l'affichage du grand player dans la vue de la musique en cours
      - 3.3.5. PlaylistViewModel** Gère l'affichage de la vue playlist
      - 3.3.6. PropertyFlyoutViewModel** Gère l'affichage de la vue des propriétés
      - 3.3.7. RadioViewModel** Gère l'affichage de la vue radio
      - 3.3.8. SettingFlyoutViewModel** Gère l'affichage de la vue de configuration
      - 3.3.9. ViewModelLocator** ViewModel locator, fait des singletons des ViewModels si l'on veut l'utiliser plusieurs fois avec le même contexte
    - 3.4. MainWindow** Vue de la fenêtre layout, qui sera active sur toutes les vues
  - 4. DTO** *Projet Data access object*
    - 4.1. Entity** Dossier contenant toutes les entités qui seront dans la base de données
      - 4.1.1. Album** Table Album
      - 4.1.2. Artist** Table Album
      - 4.1.3. Track** Table Album
      - 4.1.4. BaseEntity** Contient les propriétés communes à toutes les tables
      - 4.1.5. Context** Table Context
      - 4.1.6. Playlist** Table Playlist
      - 4.1.7. ExcludeFolder** Table ExcludeFolder
      - 4.1.8. Genre** Table Genre
      - 4.1.9. Radio** Table Radio
    - 4.2. Audio** Contient les propriétés communes entre la table Track et la table Radio
  - 5. Shared** *Projet qui contient les classes susceptibles d'être utilisées dans plusieurs layers*
    - 5.1. MusicFile** Effectue des transformations sur les fichiers audios

### 3.2.3 Base de données

#### Normes ETML

Je n'ai pas respecté les normes de codage ETML au niveau de la base de données car avec l'utilisation d'EntityFramework, la base de données devrait au final correspondre à ma structure de données dans le DTO. Certaines classes étant héritées, les noms des entités ne correspondraient plus au nom de leur table.

#### MCD

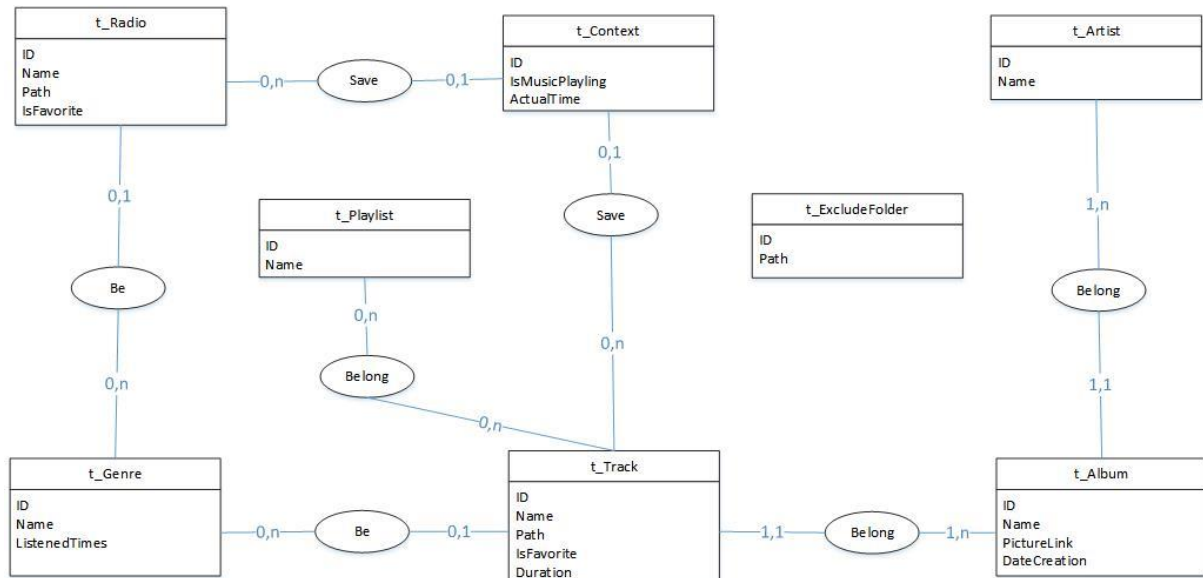


Figure 2 Modèle conceptuel de données

#### MLD

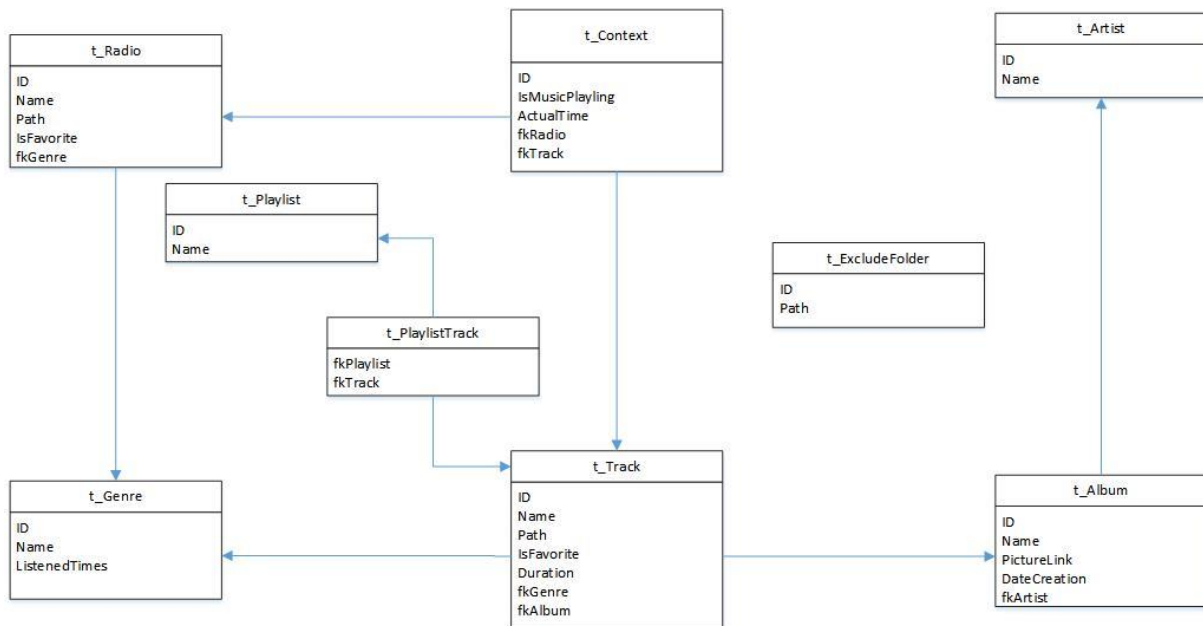
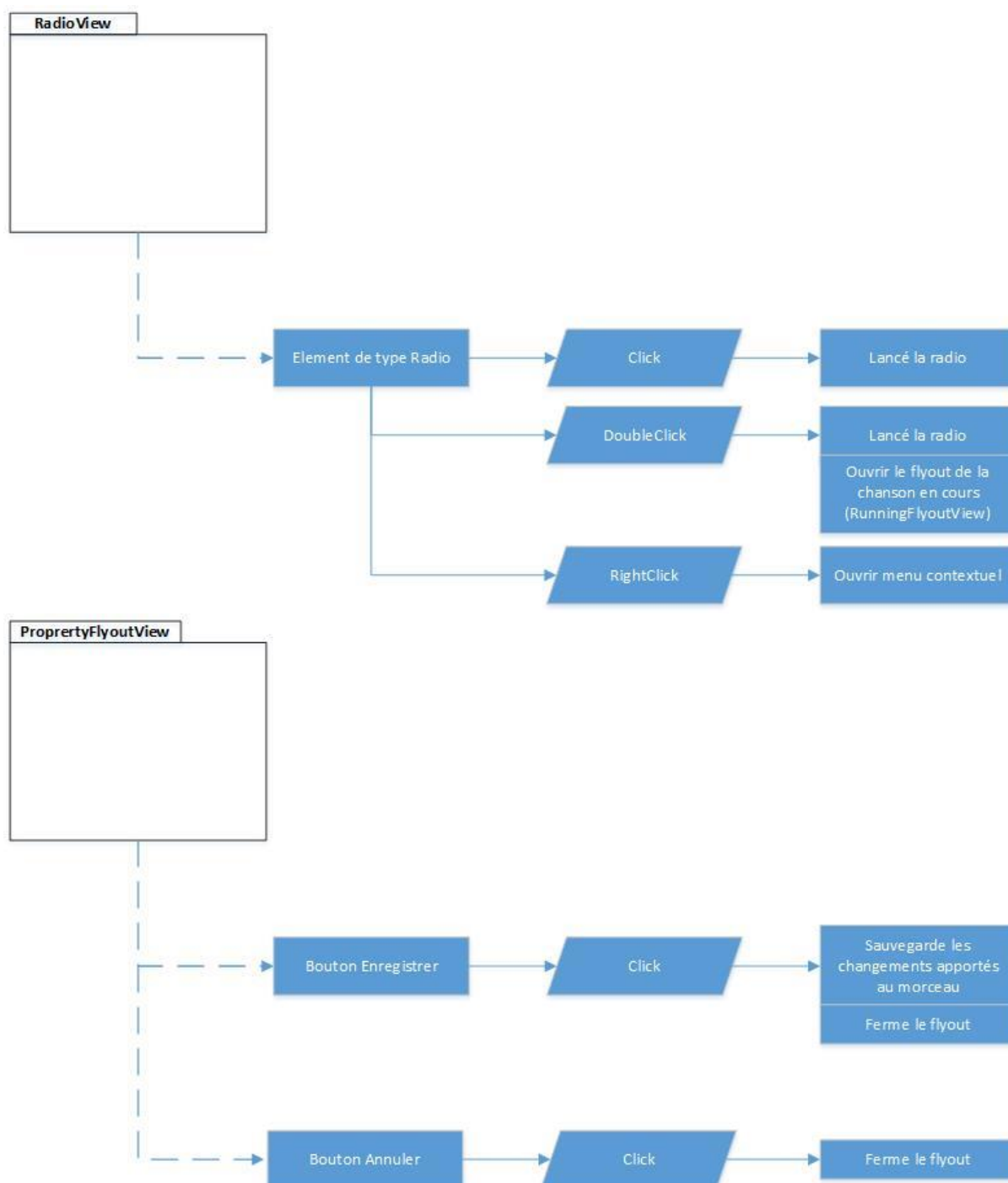
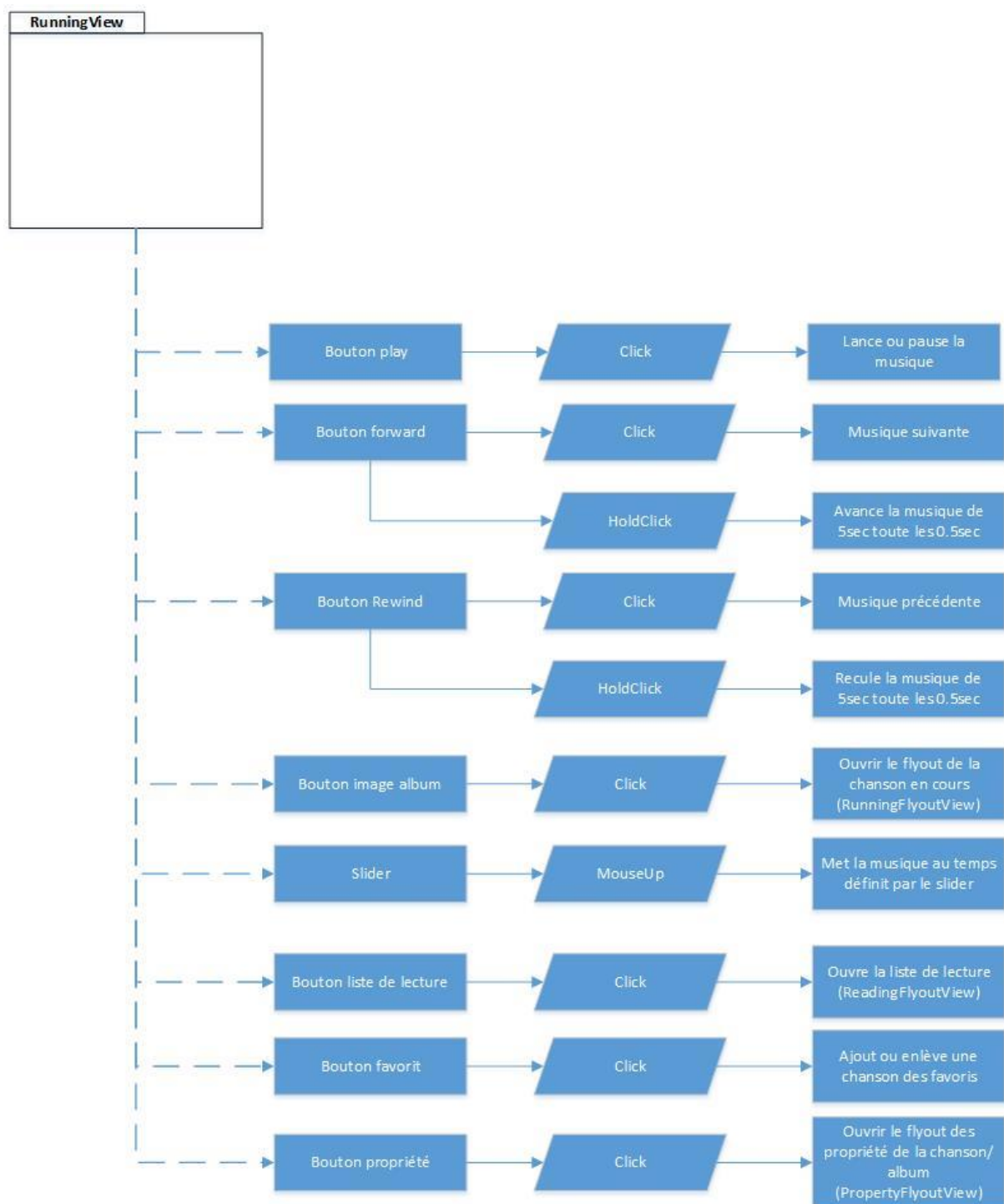


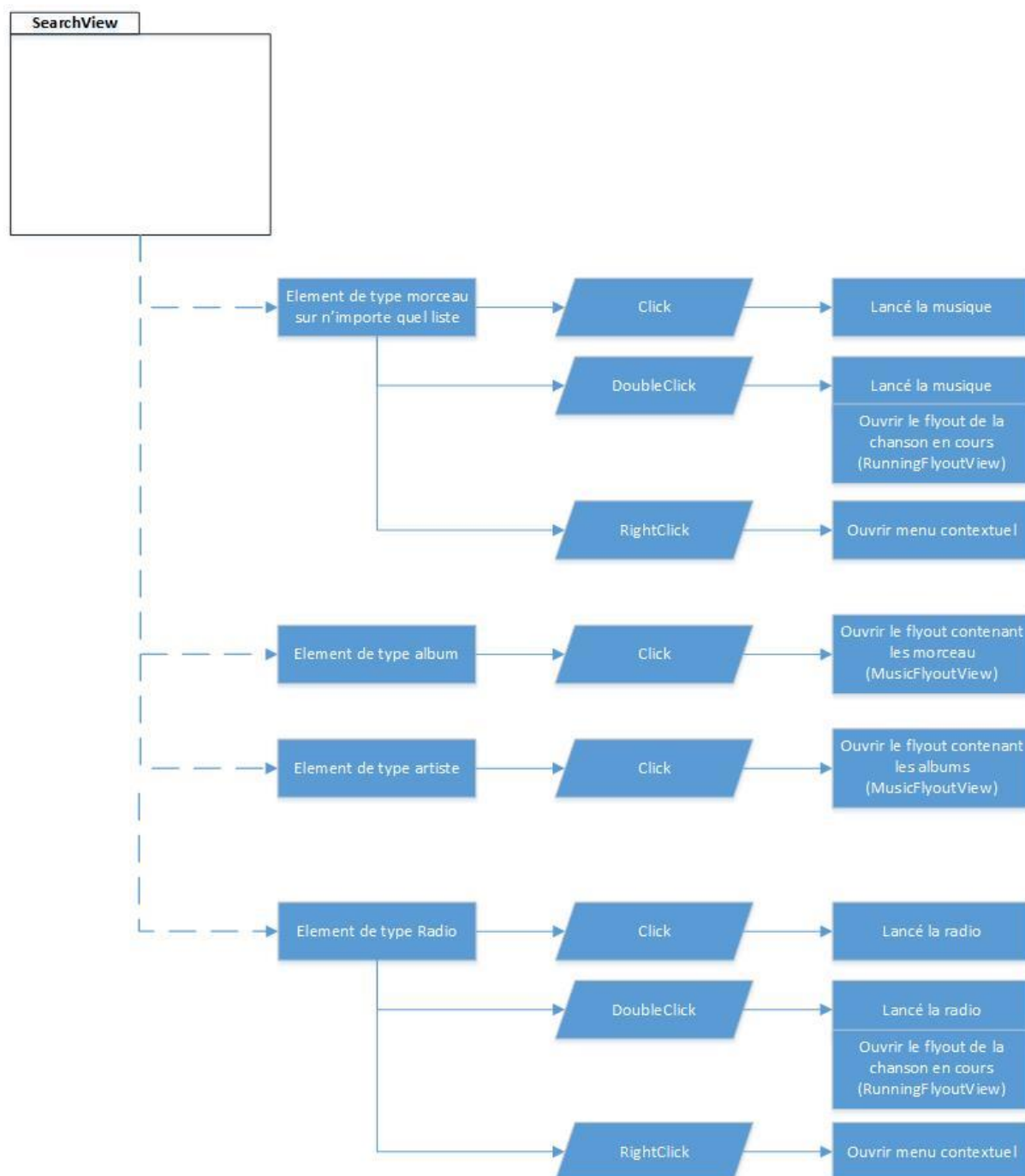
Figure 3 Modèle logique de données



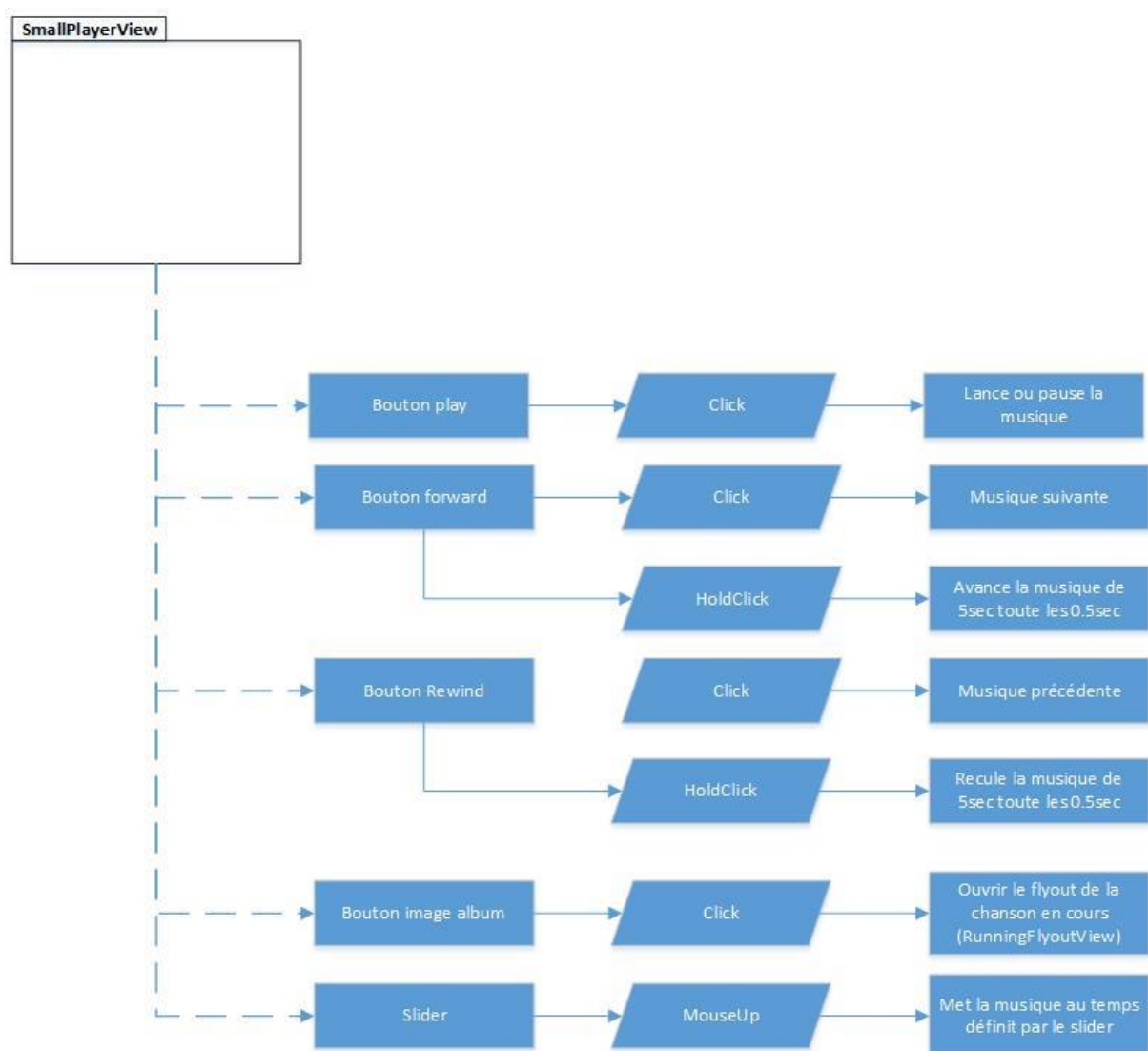
### 3.2.1 Schéma évènementiel

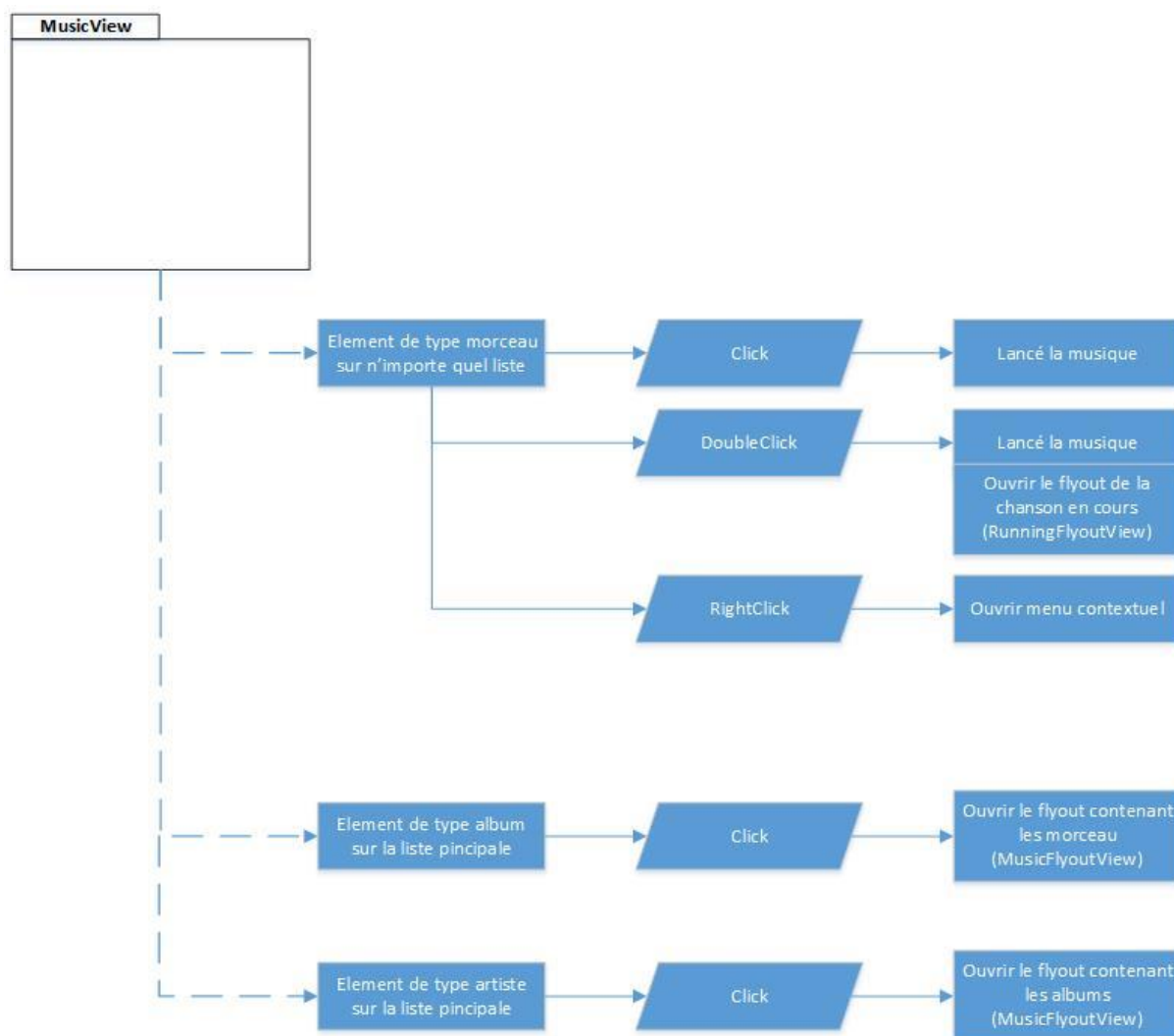


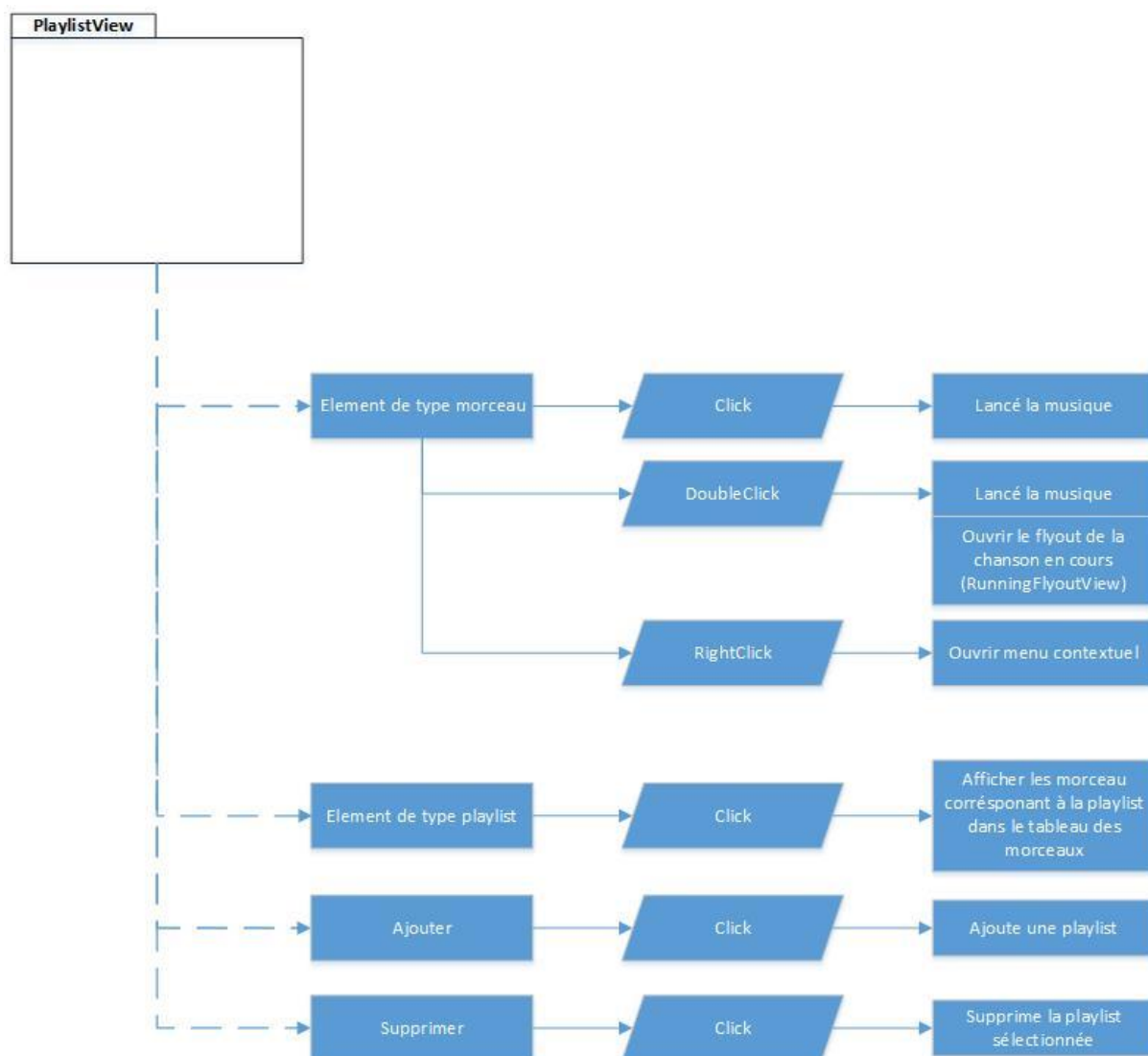












### 3.3 Conception des tests

- Ce paragraphe permet de spécifier la stratégie de test qui sera menée au point 5.1
- Qui, quand, avec quelles données, dans quel ordre, etc.
- Avec quels matériels, quels services, etc.

### 3.4 Planification détaillée

- A ce stade, après l'analyse complète du projet, un planning détaillé et complet (avec tâches, sous-tâches, dépendances, durée, ...) peut être finalisé.
- Le planning détaillé doit s'inscrire dans le planning initial. Il faut que l'on puisse situer cette planification détaillée par rapport à la planification initiale.

## 4 RÉALISATION

### 4.1 Dossier de Réalisation

- Cette partie permet de reproduire ou reprendre le projet par un tiers.
- Pour chaque étape, il faut décrire sa mise en œuvre. Typiquement :
  - Versions des outils logiciels utilisés (OS, applications, pilotes, librairies, etc.)
  - Configurations spéciales des outils (Equipements, PC, machines, outillage, etc.)
  - Code source commenté des éléments logiciels développés.
  - Modèle physique d'une base de données.
  - Arbres des documents produits.
  - Schémas, plans d'adressages, plan de nommage, etc.
- Il faut décrire le parcours de réalisation et justifier les choix.

### 4.2 Modifications

- Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.
- Date, raison, description, etc.

## 5 TESTS

### 5.1 Dossier des tests

- On dresse le bilan des tests effectués (qui, quand, avec quelles données...) sous forme de procédure. Lorsque cela est possible, fournir un tableau des tests effectués avec les résultats **attendus et obtenus, ainsi que** les actions à entreprendre en conséquence (et une estimation de leur durée).
- Si des tests prévus dans la stratégie n'ont pas pu être effectués :
  - raison, décisions, etc.
- Liste des bugs répertoriés avec la date de découverte et leur état :
  - Corrigé, date de correction, corrigé par, etc.

## 6 CONCLUSION

### 6.1 Bilan des fonctionnalités demandées

- Il s'agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.
- Si ce n'est pas le cas, estimer en « % » ou en « temps supplémentaire » le travail qu'il reste à accomplir pour terminer le tout.



## 6.2 Bilan de la planification

- Distinguer et expliquer les tâches qui ont généré des retards ou de l'avance dans la gestion du projet. Indiquer les différences entre les planifications initiales et détaillées avec le journal de travail.

## 6.3 Bilan personnel

- Si c'était à refaire:
  - Qu'est-ce qu'il faudrait garder ? Les plus et les moins ?
  - Qu'est-ce qu'il faudrait gérer, réaliser ou traiter différemment ?
- Qu'est-ce que ce projet m'a appris ?
- Suite à donner, améliorations souhaitables, ...
- Remerciements, signature, etc.

# 7 DIVERS

## 7.1 Journal de travail

- Date, activité (description qui permet de reproduire le cheminement du projet), durée, liens et références sur des documents externes. Lorsqu'une activité de recherches a été entreprise, il convient d'énumérer ce qui a été trouvé, avec les références.

## 7.2 Bibliographie

- Références des livres, revues et publications utilisés durant le projet.

## 7.3 Webographie

- Références des sites Internet consultés durant le projet.

## 8 ANNEXES

### 8.1 Cahier des charges

#### 1 INFORMATIONS GENERALES

Candidat :	Nom : PREISIG	Prénom : ERIC-NICOLAS
	✉ : <a href="mailto:preisiger@etml.educanet2.ch">mailto:preisiger@etml.educanet2.ch</a>	☎ : -
Lieu de travail :	ETML, Sébeillon 12 1004 Lausanne	
Chef de projet :	Nom : SAHLI	Prénom : BERTRAND
	✉ : <a href="mailto:bertrand.sahli@vd.ch">mailto:bertrand.sahli@vd.ch</a>	☎ : 021 316 02 62
Expert 1 :	Nom : ROLLINET	Prénom : SYLVAIN
	✉ : <a href="mailto:sylvain.rollinet@gmail.com">mailto:sylvain.rollinet@gmail.com</a>	☎ :
Expert 2 :	Nom : MELLY	Prénom : JONATHAN
	✉ : <a href="mailto:jonathan.melly@vd.educanet2.ch">mailto:jonathan.melly@vd.educanet2.ch</a>	☎ :
Dates de réalisation :	Du lundi 1 mai à 8h au mercredi 7 juin à 11h25	
Horaire de travail : (Basé sur l'horaire officiel)	Lundi	08h00-11h25 - Pentecôte, 5 juin 2017
	Mardi	- -
	Mercredi	08h00-12h15 13h10-16h35 Exa CG 31 mai 2017 après-midi
	Jeudi	08h00-11h25 12h20-16h35 Pont de l'Ase, 25 mai 2017
	Vendredi	08h00-12h15 13h10-16h35 Pont de l'Ase, 26 mai 2017
Présentation :	Entre le mercredi 14 et jeudi 15 juin 2017	
Nombre d'heures :	Environ 110 heures	
Planning (en H ou %)	Analyse: 15%, Implémentation: 40%, Tests: 20%, Documentation: 25%	

#### 2 PROCÉDURE

Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le 1er jour.

Le cahier des charges est approuvé par i-CQ VD. Il est en outre présenté, commenté et discuté avec le candidat. Par sa signature, le candidat accepte le travail proposé.

Le candidat a connaissance de la feuille d'appréciation avant de débiter le travail.

Le candidat est entièrement responsable de la sécurité de ses données.

En cas de problèmes graves, le candidat avertit au plus vite les deux experts et son chef de projet.

Le candidat a la possibilité d'obtenir de l'aide, mais doit le mentionner dans son dossier de projet.

A la fin du délai imparti pour la réalisation du TPI, le candidat doit transmettre par courrier électronique le dossier de projet aux deux experts et au chef de projet. En parallèle, une copie papier du rapport doit être fournie sans délai en trois exemplaires. Cette dernière doit être en tout point identique à la version électronique.

#### 3 TITRE

Gestionnaire de playlists audio et de flux radios

---

## 4 SUJET

Développer une application desktop en C# selon le pattern MVVM afin de gérer des playlists audio et des flux radios publics diffusés sur Internet. Il s'agira notamment :

- d'assurer l'installation de l'application sur les OS Microsoft 7 à 10.
- d'exploiter les données audios locales (sans les dupliquer).
- de créer un lecteur audio « user friendly ».
- de stocker les données de l'application dans une base de données SQLite.

---

## 5 MATÉRIEL ET LOGICIEL À DISPOSITION

PC standard de l'ETML (Windows 7 – 64bit)

Player de machines virtuelles (vmware ou équivalent)

Suite Microsoft Office 2013

IDE C# (Visual Studio Community 2015 ou 2017, SQLite Precompiled Binaries for Windows)

---

## 6 PRÉREQUIS

Compétences élémentaires en base de données (SQL) et en programmation (C#) → Modules 103, 104, 105, 303.

Compétences bureautiques → Modules 301 et 302

---

## 7 DESCRIPTIF DU PROJET

Concevoir une interface graphique « user friendly » inspirée de Spotify.

Modéliser et implémenter une base de données permettant de soutenir les fonctionnalités de l'application. Soit au minimum :

- o la bibliothèque des contenus audios du PC (vide au 1<sup>er</sup> lancement).
- o les habitudes d'écoute de l'utilisateur (ses chansons favorites, ses styles musicaux préférés, ses listes de lecture, les 10 dernières radios écoutées).

Développer les fonctionnalités suivantes :

- o au 1<sup>er</sup> lancement, scanner les contenus audios du PC et indexer tous les contenus audios découverts dans la bibliothèque « locale ».
- o à la demande, (re)scanner les contenus audios du PC et indexer tous les nouveaux contenus audios découverts dans la bibliothèque « locale ».
- o rechercher, dans « locale », des chansons (au minimum par titre ou groupe).
- o concevoir des playlists (au minimum les identifier et les jouer) et y ajouter la/les chansons sélectionnées (les moyens de sélection seront ergonomiques et efficaces).
- o jouer des chansons (prévoir les actions habituelles : pause, play, slider, stop, ...)
- o jouer des listes de lecture (en plus des actions habituelles, prévoir les actions : précédant, suivant, lecture continue, lecture aléatoire, lecture répétée, ...)
- o rechercher des radios diffusées sur Internet (au minimum par nom ou style musical).
- o se souvenir des 10 dernières radios sélectionnées.
- o écouter le flux de la WebRadio sélectionnées.
- o fournir les informations de la chanson en cours (au minimum le titre, le groupe, l'album, le style musical, une image significative et le passage du temps).
- o reprendre le dernier contexte applicatif à l'ouverture de l'application.

---

## 8 POINTS ÉVALUÉS DURANT LE PROJET

- La méthodologie de travail ainsi que la conformité des tâches réalisées par rapport au planning initial.
- Le choix des « métadonnées » nécessaires à l'indexation des chansons.
- La pertinence du modèle de données réalisé.
- La mise en œuvre de tests (notamment les tests unitaires) pour vérifier les fonctionnalités implémentées.
- La journalisation des tâches ainsi que la rédaction régulière de la documentation.

---

## 9 LIVRABLES

Le candidat est responsable de livrer à son chef de projet et aux deux experts :

- Une planification initiale (à envoyer par mail le 3 mai à 12h15 au plus tard)
- Un rapport de projet (à envoyer par mail, en l'état, toutes les fins de semaine)
- Un journal de travail (à envoyer par mail, en l'état, toutes les fins de semaine)
- Le code source de l'application
- Une archive autoinstallable de l'application compatible MS Windows 7 à MS Windows 10.
- Un guide d'utilisation

---

## 10 POINTS TECHNIQUES ÉVALUÉS SPÉCIFIQUES AU PROJET

La grille d'évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, ...).

En plus de cela, le travail sera évalué sur les trois points spécifiques suivants :

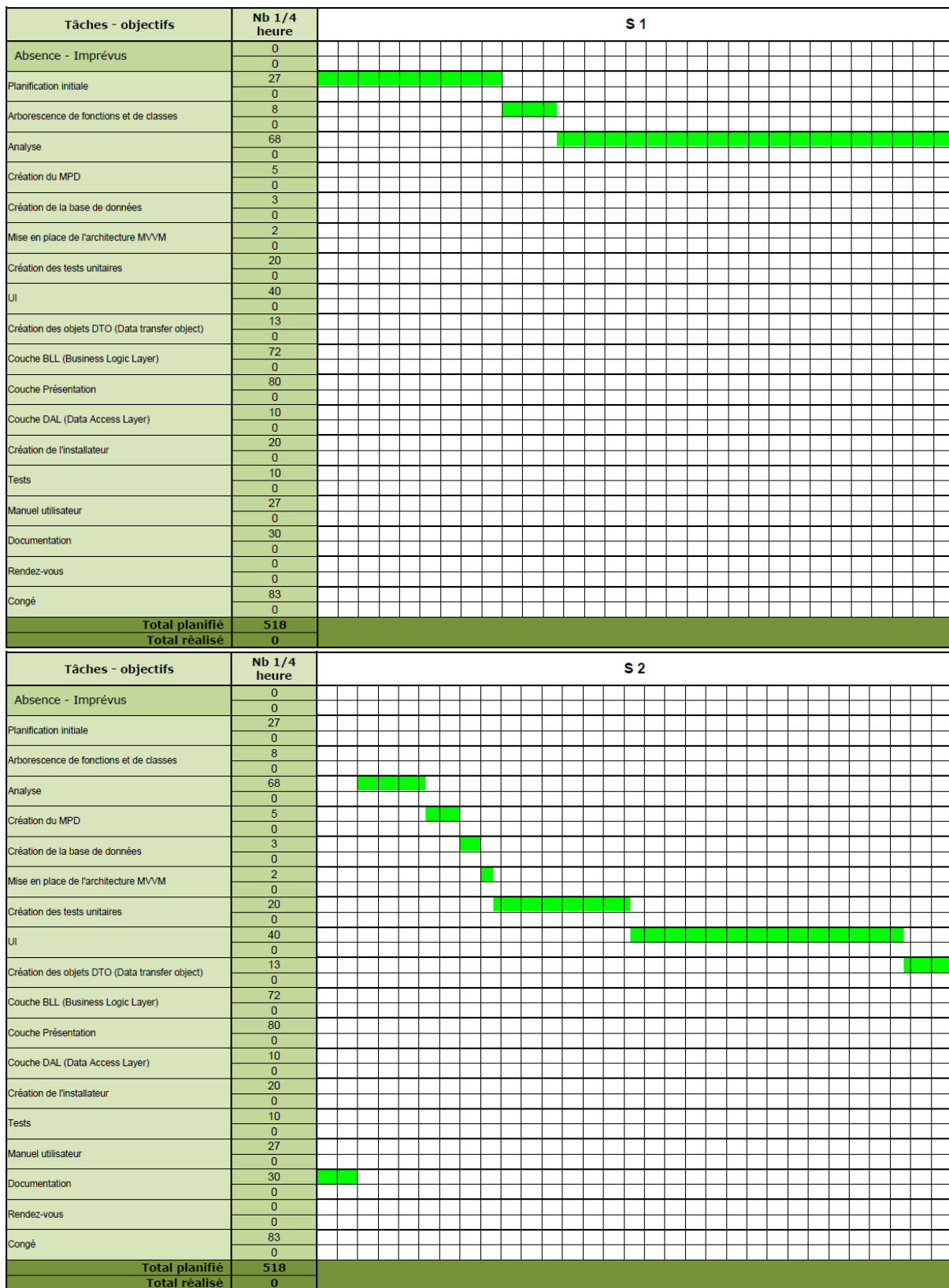
- Le point spécifique n° 1 → L'extraction de métadonnées de divers format audio.
- Le point spécifique n° 2 → La recherche de WebRadio.
- Le point spécifique n° 3 → Le (re)scan des contenus audios.

---

## 11 VALIDATION

	Lu et approuvé le :	Signature :
Candidat :		
Expert n°1 :		
Expert n° 2 :		
Chef de projet :		

## 8.2 Gantt





Tâches - objectifs	Nb 1/4 heure	S 5																			
Absence - Imprévis	0																				
	0																				
Planification initiale	27																				
	0																				
Arborescence de fonctions et de classes	8																				
	0																				
Analyse	68																				
	0																				
Création du MPD	5																				
	0																				
Création de la base de données	3																				
	0																				
Mise en place de l'architecture MVVM	2																				
	0																				
Création des tests unitaires	20																				
	0																				
UI	40																				
	0																				
Création des objets DTO (Data transfer object)	13																				
	0																				
Couche BLL (Business Logic Layer)	72																				
	0																				
Couche Présentation	80																				
	0																				
Couche DAL (Data Access Layer)	10																				
	0																				
Création de l'installateur	20																				
	0																				
Tests	10																				
	0																				
Manuel utilisateur	27																				
	0																				
Documentation	30																				
	0																				
Rendez-vous	0																				
	0																				
Congé	83																				
	0																				
<b>Total planifié</b>	<b>518</b>																				
<b>Total réalisé</b>	<b>0</b>																				

Tâches - objectifs	Nb 1/4 heure	S 6																			
Absence - Imprévis	0																				
	0																				
Planification initiale	27																				
	0																				
Arborescence de fonctions et de classes	8																				
	0																				
Analyse	68																				
	0																				
Création du MPD	5																				
	0																				
Création de la base de données	3																				
	0																				
Mise en place de l'architecture MVVM	2																				
	0																				
Création des tests unitaires	20																				
	0																				
UI	40																				
	0																				
Création des objets DTO (Data transfer object)	13																				
	0																				
Couche BLL (Business Logic Layer)	72																				
	0																				
Couche Présentation	80																				
	0																				
Couche DAL (Data Access Layer)	10																				
	0																				
Création de l'installateur	20																				
	0																				
Tests	10																				
	0																				
Manuel utilisateur	27																				
	0																				
Documentation	30																				
	0																				
Rendez-vous	0																				
	0																				
Congé	83																				
	0																				
<b>Total planifié</b>	<b>518</b>																				
<b>Total réalisé</b>	<b>0</b>																				

- Listing du code source (partiel ou, plus rarement complet)
- Guide(s) d'utilisation et/ou guide de l'administrateur
- Etat ou « dump » de la configuration des équipements (routeur, switch, robot, etc.).
- Extraits de catalogue, documentation de fabricant, etc.