LORANS-CANO Gurvan
LE BERRE Samuel
1ère Année - Groupe A2
Année 2016 / 2017

Projet programmation M2107

Rendu Final

IUT Vannes

M. LEFEVRE, professeur référent

Table des matières

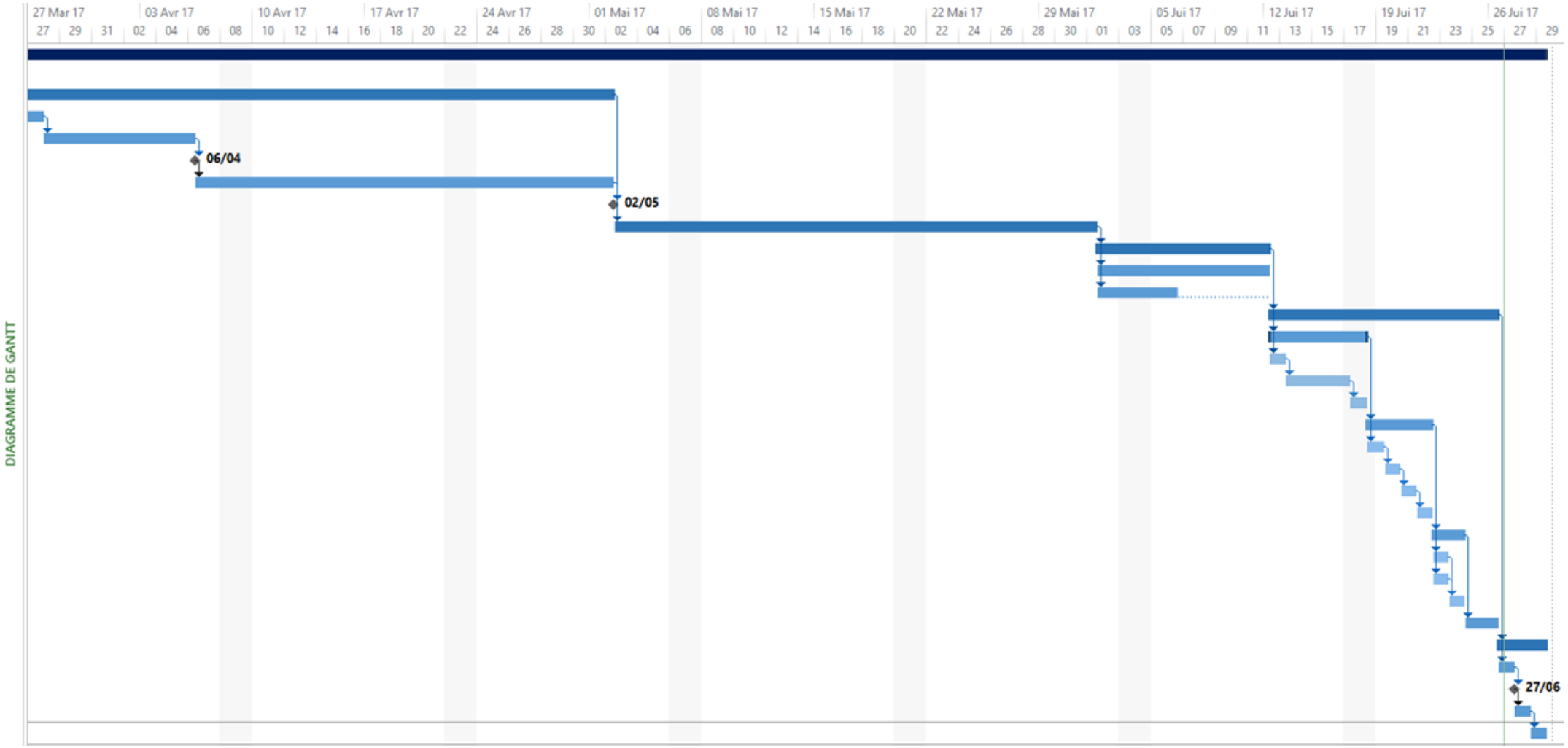
Présentation

Gestion du planning d'actions

Pour cette première partie du projet, nous avons beaucoup utilisé Google Drive afin d'optimiser notre productivité. Cela nous a permis de partager les fichiers plus facilement au sein du groupe, tout en ayant la possibilité de travailler individuellement. De ce fait chacun a pu traiter une partie différente pour le cahier des charges ainsi celui d'analyse.

Lors de la deuxième période, en binôme, Google Drive nous a également été utile pour partager nos différents fichiers (java, lib, class, build …) pour continuer notre travail en cours ainsi qu'à la maison. Tous les matins en arrivant à l'IUT, nous faisions un bref bilan sur l'état du projet (20 à 30 minutes), de façon à être d'autant plus efficace. Nous n'avons pas eu besoin d'exploité Trello ou autre d'outil de gestion de projet similaire n'ayant pas de difficulté à communiquer entre nous. Par ailleurs, notre emploi du temps et nos heures passées à l'IUT (7h55 à 17h15) étaient également propice à l'avancement de notre projet.

Diagramme Gantt



Nom	Durée	Début	Fin
Projet programmation M2107	91 jours	27 Mars 2017 09:00	29 Juin 2017 16:00
Etudes Préalables	35 jours	27 Mars 2017 09:00	02 Mai 2017 12:00
Analyse des besoins	1 jour	27 Mars 2017 09:00	27 Mars 2017 18:00
Rédaction du cahier des charges	9 jours	28 Mars 2017 09:00	06 Avril 2017 11:00
Rendu du cahier des charges	0 jour	06 Avril 2017 11:00	06 Avril 2017 11:00
Rédaction du cahier d'analyse	25 jours	06 Avril 2017 11:00	02 Mai 2017 12:00
Rendu du cahier d'analyse	0 jour	02 Mai 2017 12:00	02 Mai 2017 12:00
Pause dans le projet (Cours et Partiels)	29 jours	02 Mai 2017 14:00	01 Juin 2017 15:00
Recherche Java et SQL	10 jours	01 Juin 2017 15:00	12 Juin 2017 10:00
Recherche JDBC	10 jours	01 Juin 2017 15:00	12 Juin 2017 10:00
Recherche SQL	10 jours	01 Juin 2017 15:00	12 Juin 2017 10:00
Réalisation	14 jours	12 Juin 2017 10:00	26 Juin 2017 16:00
Programmation "Model"	6 jours	12 Juin 2017 10:00	18 Juin 2017 11:00
Accès a une base et Sécurisation de la connexion	1 jour	12 Juin 2017 10:00	13 Juin 2017 10:00
Extraction des données (classe Table, View et Trigger)	4 jours	13 Juin 2017 10:00	17 Juin 2017 10:00
Implémentation des fonctions pour créer, supprimer une Table, colonne et T-uples	1 jour	17 Juin 2017 10:00	18 Juin 2017 11:00
Programmation "View"	4 jours	18 Juin 2017 11:00	22 Juin 2017 12:00
GUI(classe contenant les composants graphiques), MenuBar, ToolBar et Connexion	1 jour	18 Juin 2017 11:00	19 Juin 2017 12:00
LeftPane et RightPane	1 jour	19 Juin 2017 14:00	20 Juin 2017 12:00
JTabbedPane, ButtonTab	1 jour	20 Juin 2017 14:00	21 Juin 2017 12:00
Dialog modification Table	1 jour	21 Juin 2017 14:00	22 Juin 2017 12:00
Programmation Controller	2 jours	22 Juin 2017 14:00	24 Juin 2017 12:00
LeftPane et RightPane	1 jour	22 Juin 2017 14:00	23 Juin 2017 12:00
MenuBar et ToolBar	1 jour	22 Juin 2017 14:00	23 Juin 2017 12:00
Dialog et Connexion	1 jour	23 Juin 2017 14:00	24 Juin 2017 12:00
Test	2 jours	24 Juin 2017 14:00	26 Juin 2017 16:00
Déploiement	3 jours	26 Juin 2017 16:00	29 Juin 2017 16:00
Rédaction manuel d'utilisation	1 jour	26 Juin 2017 16:00	27 Juin 2017 16:00
Rendu du projet	0 jour	27 Juin 2017 16:00	27 Juin 2017 16:00
Préparation de l'Oral	1 jour	27 Juin 2017 16:00	28 Juin 2017 16:00
Oral et fin de projet	1 jour	28 Juin 2017 16:00	29 Juin 2017 16:00

Explications

Nous avons bien respecté les dates et les étapes de la première partie du projet lorsque nous étions à 4 cependant avant la réalisation il nous a fallu une période de documentation plus importante que prévu car la connexion entre java et SQL ainsi que JDBC sont très mal expliquée sur les forums et autres sites. La notion du JDBC nous a été compliqué à cerner et même encore maintenant on a encore du mal à le définir précisément.

En plus de ça, on a modifié les étapes de notre projet afin de dédier du temps au « controller » qui est la partie la plus importante du projet car il fait les interactions entre interface graphique et donnée. On a eu le temps de faire beaucoup de fonctionnalité supplémentaire qui n'était pas dans le cahier des charges de base comme la partie droite de notre interface qui permet d'ajouter ou supprimer une table, une colonne ou une ligne en 3 clic. Ces fonctionnalités impliquent des méthodes additionnelles au package « controller » et « model » afin d'assurer le bon fonctionnement des boutons et autres interactions.

Quelle est l'importance de planifier

Pour nous il est très important de planifier de quelle manière va se dérouler le projet. Sont importances est de 9 points sur 10. Cependant de notre point de vue il est plus intéressant de planifier le déroulement de la journée que le déroulement du projet complet puisque le projet aura du retard ou des inconvénients tandis qu'une journée est planifiable car il s'agit de court terme et que si on sait ce que l'autre personne fait on peut anticiper nos actions à venir et celle qui reste à faire.

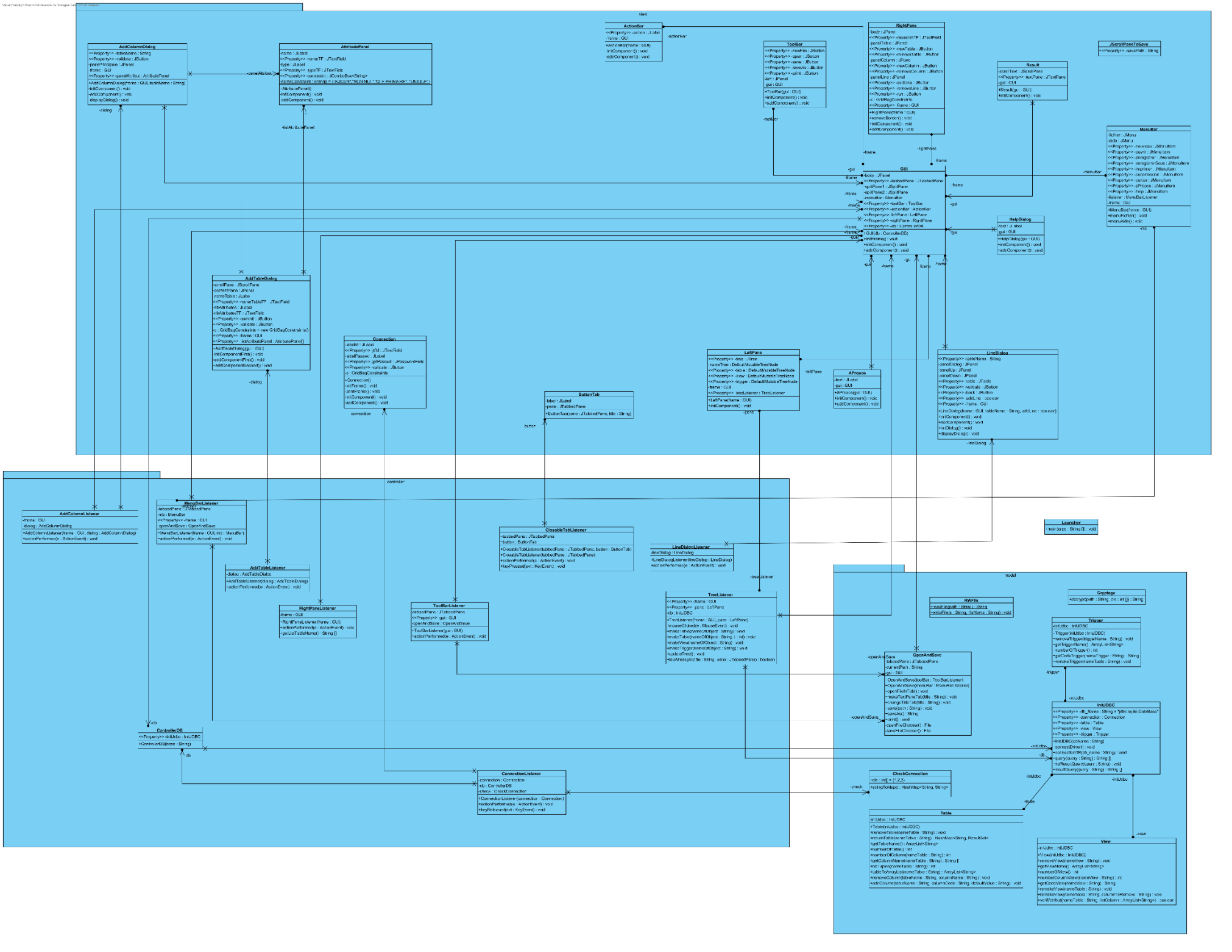
Recommandation

Ce projet de programmation a été très instructif pour avoir une approche du travail en équipe dans le domaine informatique qui est indispensable en entreprise. Ce production en groupe nous permet également de nous sortir de notre isolement et nous apporte de multiples avantages. D'une part, c'est l'occasion de partager ses connaissances avec d'autres, tout en profitant de celles des collaborateurs. Le groupe apporte aussi un soutien et donne confiance en soi. Un travail en équipe permet d'autre part d'élargir ses pistes et d'obtenir de nouvelles idées. Il était rassurant de travailler en groupe, d'autant plus que nous n'avions jamais étudié le JDBC auparavant.

D'autre part, il est plaisant d'exploré un sujet que nous n'avions pas eu l'occasion de voir en cours, car ceci nous pousse à chercher les informations par nous-même. Néanmoins, il aurait été intéressant de disposer de quelques pistes pour certaines parties (installation de Ant avec l'intégration de JUnit).

Nous recommanderions également les 2 semaines qui ont été dédié uniquement à l'élaboration de ce projet ce qui nous a mis dans une ambiance plus professionnelle.
Pour les années à venir nous recommanderions d'aborder les fichiers xml de façon plus approfondie.
Le choix du binôme est important car celui-ci est décisif pour la motivation sur la suite du projet.

Diagramme de classe


view
controller
model
AddColumnDialog
AttributePanel
ActionBar
RightPane
ToolBar
Result
JScrollPaneToSave
MenuBar
GUI
HelpDialog
AddTableDialog
Connection
ButtonTab
LeftPane
APropos
LineDialog
Launcher
AddColumnListener
MenuBarListener
ClosableTabListener
LineDialogListener
TreeListener
RightPaneListener
ToolBarListener
RWFile
Cryptage
Trigger
OpenAndSave
InitJDBC
ControllerDB
ConnectionListener
CheckConnection
Table
View

Choix Technique

Model

On a modifié notre diagramme de classe pour interagir non seulement avec les tables mais aussi avec les triggers et les vues, ces 3 classes sont appelés dans une classe qui initialise la connexion a la base de données.

Il a fallu ajouté au package « model » la classe « cryptage » qui chiffre les identifiants et les décrypte à partir d'un fichier pour sécuriser la connexion à la base de données grâce à la classe CheckConnection. La classe RWFile écrira et lira les fichiers.

Nous avons choisi de ne pas stocker le contenu de la base de données dans des variables. Cela demanderait un stockage trop important alors que l'on peut directement récupérer les informations venant de la base de données, ainsi on récupère le code de création des tables, vues et triggers.

La classe OpenAndSave est placé dans le "model" bien qu'elle interagisse avec l'utilisateur par des éléments graphiques. De notre point de vue, cette classe appartient au "model" puisqu'elle sauvegarde des fichiers sur notre système. Cette classe manipule directement les données contenues dans le système, ce qui est le critère principal du package model.

View

Nous avons fait le choix de séparer notre interface graphique en plusieurs partie :

- La partie droite qui comporte les boutons de toutes les modifications de Table possibles comme ajout ou suppression de Table, Colonne ou Ligne (RightPane)

- La partie gauche qui contient l'arborescence (JTree) de notre base de données où l'on peut voir les Tables, Vues et Triggers présents dans la base de données (LeftPane)

- La partie barre d'outils qui possède les fonctionnalités Nouveau, Ouvrir, Enregistrer, Enregistrer-Sous et Imprimer présentes sur de nombreux programmes (ToolBar)

- La partie Menu permettant de se déconnecter ou de quitter l'application en plus des actions possibles dans la barre d'outils (MenuBar)

- La partie centrale instanciée par un attribut JTabbedPane dans la GUI principale qui affiche le contenu des Tables et des Vues ainsi que le code de création des Triggers via un double-clic dans le JTree sur le nom de l'objet à ouvrir. Cette partie permet également de lancer des requêtes à la base de données et d'écrire des scripts de création (table, vues, triggers) ainsi que des insertions de t-uples

- Une partie hors de la fenêtre principale est dédié à la console (Result) qui affichera le résultat des requêtes envoyées à la base de données. Cette partie étend la classe Java JDialog et est rattaché à la fenêtre principale (GUI)

- 2 autres dédiée aux crédits et à l'aide pour l'utilisateur

Controller

Nous avons jugé utile de créer plusieurs classes de « Controller ». Il y a environ un contrôleur par classes du package vue. Nous voulions respecter le plus possible la méthode Model View Controller (MVC). De ce fait, chaque classe du package « View » comprenant plus de 2 interactions avec le model possède un Listener/Controller associé.

Il y a par exemple le controller pour le LeftPane(JTree) qui vérifie si un élément a été double-cliqué, dans ce cas, s'il s'agit d'une table il affiche celle-ci avec en dessus la requête permettant de ressortir toutes ses données. Le ClosableTabListener quant à lui est un peu particulier car il va servir uniquement à fermer les onglets dans la partie centrale(JTabbedPane), ce Listener étend la classe KeyAdapter et implémente l'interface ActionListener.

Campagne de test

Pour tout ce qui concerne la campagne de test, nous avons choisi de tester toutes les méthodes (donc avec retour) de test du package « Model » avec JUnit nous avons donc fait un dossier de classes test. JUnit étant intégré dans Ant nous avons fait en sorte de le lancer et de créer un répertoire où est stocké un « Report » du lancement de Ant qui génère une page web où sont répertorié tous les résultats.

Pour les méthodes du package « View » nous avons réfléchis au préalable aux éventuelles cas d'erreur possible puis nous faisions tester en fin de journée au binôme. Ce test à la fin du développement de la méthode nous a permis de corriger certains soucis ou d'améliorer les méthodes (demande de confirmation avant de quitter quand on fait CTRL + Q par exemple). Pour toutes les méthodes du controller il n'y avait pas nécessairement besoin de les tester car celles-ci servent seulement à faire le pont entre les 2 autres packages.

Quand notre logiciel à commencer à être opérationnel nous avons donc relancer un test nous faisions manuellement plusieurs scénarios différents afin de voir que tout fonctionnait correctement. Généralement le binôme testait les parties graphiques de l'autre personne car l'esprit critique en est que meilleur en effectuant de la sorte.

Avancement

Fonctionnalité de base

Nous avons accompli toutes les fonctionnalités dites dans le cahier d'analyse à savoir :

- ✓ Accéder à une base de données depuis Java
  - ➢ On accède à la base de données par la classe InitJDBC qui initialise la connexion.

- ✓ Posséder une interface texte
  - ➢ On peut écrire des fichiers textes en SQL à l'aide du bouton nouveau fichier dans la barre d'outils qui ouvre une interface textuelle

- ✓ Posséder une interface graphique
  - ➢ Notre interface graphique ce nomme GUI pour Graphical User Interface, elle appel de nombreuse classe graphique tel que RightPane, LeftPane, MenuBar ou encore ToolBar.

- ✓ Ouvrir, enregistrer, enregistrer sous, nouveau fichier (stocker dans des fichiers où sont enregistré les requêtes et permettre leur lecture etc.)
  - ➢ Les différentes fonctions sont accessibles depuis ToolBar par des boutons ou via le Menu du programme ou encore des raccourcie clavier comme Contrôle + P pour imprimer.

- ✓ Visualiser des informations
  - ➢ Toutes les informations de notre base de données sont accessibles dans le centre de notre application en double cliquant sur l'élément voulu dans la partie gauche de l'application. L'élément centrale étant un JtabbedPane on peut lui ajouter des éléments à tout moment

- ✓ Édition, insertion, suppression d'une table, d'un t-uple et enregistrement Automatique
  - ➢ Le Panel de droite contient 6 boutons permettent l'ajout ou la suppression d'une table, d'une colonne ou d'une ligne de la base de données

- ✓ Sécurité pour la connexion à la base de données (attribuer droits aux utilisateurs)
  - ➢ La classe cryptage nous permet de crypter des identifiants et mots de passe puis de les mettre dans un fichier, les valeurs de ce fichier seront récupérées lors de la connexion pour les comparer dans CheckConnection afin de valider ou non la connexion à la base

- ✓ Posséder un champ de saisie de texte (pour l'écriture des requêtes SQL) → Nouveau fichier
  - ➢ Tout comme l'interface texte c'est le bouton nouveau fichier qui nous permet de faire des requêtes à partir d'une saisie de texte

Fonctionnalité additionnelle

Nous avons même pu faire quelques fonctionnalités en plus :

- ✓ Pouvoir créer une table juste avec un enchaînement de clic sur des boutons et de saisie de nom d'attribut
  - ➢ L'interface graphique que nous avons développée nous permet de créer des tables, colonnes et lignes en 3 clics tout au plus.

- ✓ Onglet aide
  - ➢ L'onglet d'aide nous permet d'accéder à un lien vers un tutoriel SQL

- ✓ Exporter sous un autre format
  - ➢ De par le bouton Sauvegarder-Sous du Menu ou de la barre d'outils nous pouvons enregistrer sous format txt ou sql

- ✓ Imprimer le fichier SQL qui a été chargé → ToolBar bouton imprimer
  - ➢ Le bouton imprimer du Menu et de la barre d'outils nous donne accès à l'impression, on peut également y avoir accès par le raccourcie contrôle + P

Difficulté rencontrée

Nous avons utilisé SQLite comme base de données seulement cette base de données à quelque lacune comme le fait qu'il n'y est pas d'utilisateur, pour combler ce manque de fonction nous avons dû utiliser d'autres voies telles que l'utilisation d'un fichier contenant les pseudos utilisateur et leur mot de passe crypté.

Une des autres fonctionnalités qui manque à SQLite est la modification de table pour l'ajout et la suppression de colonne pour contourner cette méthode inexistante nous avons créé la méthode addColumn et removeColumn dans la classe Table, ces méthodes crée de nouvelle table avec les colonnes de l'ancienne table, puis efface l'ancienne table et renomme la nouvelle table comme l'ancienne.

Une des autres difficultés a été de réfléchir à comment on aurait pu sauvegarder le chemin de sauvegarder d'un fichier et le mémoriser dans l'application tant que ce fichier est ouvert afin de ne pas redemander à l'utilisateur de choisir l'emplacement où stocker ce fichier mais d'écraser directement le précédent.

Bilan personnel

Samuel

Ce projet est réellement intéressant quant à la compréhension de comment fonctionne une base de données et qu'elle est le programme qui se cache derrière lorsque l'on souhaite crée une table ou une vue.

Il m'a permis de mieux comprendre certain programme ou langage qui peuvent être utilisé avec java comme JUnit ou Ant qui sont des fonctionnalités qui devraient nous être présenté avant tant leur utilisation est utile quant à la rigueur qui peut être attendu dans les différents programmes à réaliser dans le futur.

J'ai été fortement étonné de voir que la programmation en MVC augmentait le nombre de classe de façon exponentielle, ainsi pour chaque classe dans le package vue on a un controller qui est parfois très court et ne nécessite pas plus de quelques lignes.

Gurvan

Ce projet m'a beaucoup apporté sur le plan de la gestion d'un projet informatique en groupe. J'ai pu me rendre compte que la communication était vraiment importante pour le bon déroulement de celui-ci. De plus nous avons pu découvrir de nombreux nouveaux outils que nous n'avions jamais vu durant l'année (Ant, JUnit, JDBC, Installation et mise en place d'une base de données, …).

Du côté de l'Interface Homme-Machine, ce projet a été enrichissant car grâce à celui-ci j'ai pu approfondir mes connaissances sur les différents Layouts et en découvrir des nouveaux. Ce projet a également permis de rendre compte de l'intérêt de l'analyse préliminaire car grâce aux différents documents fourni précédemment nous avons pu avancer plus rapidement notamment sur l'IHM