



Institut national des sciences appliquées et de
technologie

Rapport De Projet

Conception et réalisation d'une application à l'aide de la gestion des
ressources au sein d'un institut

Réalisé par : Khemissi Amir, Hammami Chedi et Araar Nour El Houda

Niveau : 2ème année réseau et télécommunication

Professeur : Karoui Kamel

Année universitaire 2019-2020

Sommaire

- [Introduction](#)-----
- [Application client](#) -----
 - Introduction-----
 - Présentation-----
- [Application serveur](#)-----
 - Introduction-----
 - Présentation-----
- [La mise en réseau](#) -----
- [Sécurité informatique](#) -----
- [Développement](#) -----
- [Base de données](#) -----
 - Conception -----

Introduction

L'objectif de notre projet présenté dans ce rapport est la conception et la réalisation d'une application de gestion des différents ressources (étudiants, enseignants et matières) au sein d'un institut.

En fait, ce projet est considéré comme deux applications indépendantes et chacune communique avec l'autre en utilisant des sockets TCP/IP dans un réseau.

De plus, cette communication est garantie à l'aide d'un protocole développé qu'on l'appelle ACN.

Ce projet offre aux administrateurs de notre institut une facilité de gestion des ressources avec un espace sécurisé et bien organisé.

Application Client

● Introduction :



Pour interagir avec la base de données, il faut utiliser une application client qui va communiquer avec le serveur et transmettre aux utilisateurs le résultat que le serveur lui aura donné.

● Présentation :

Cette application se présente aux administrateurs à travers deux fenêtres.


La première consiste à s'identifier à travers le nom d'utilisateur et le mot de passe fourni par l'administrateur de la base de données qui est chargé de créer les comptes pour les utilisateurs en assurant que le mot de passe doit avoir 4-14 caractères et précise aussi un certain privilège pour chaque utilisateur.

Lorsque l'utilisateur est bien connecté l'application s'ouvre et l'utilisateur est face à un accueil qui contient des informations globales sur toute la base de données telle que le nombre total d'étudiants, nombre total des enseignants, nombre de matières et nombre de filières avec un petit texte explicatif de ce projet.

Puis on a un espace étudiant : c'est l'espace à travers lequel l'utilisateur peut manipuler les données des étudiants (modifier, supprimer, ajouter, voir ces matières etc....) ainsi chercher selon un critère choisi parmi les champs présents. D'autre part on peut exporter les données sélectionnées de la liste des étudiants dans Google Sheet  , puis on peut les imprimer ou les envoyer dans des Email et n'oublions pas que chaque étudiant peut avoir une photo d'extension JPG ou PNG  et de taille qui ne dépasse pas 4 Mo sinon elle sera remplacée par une image par défaut.

L'utilisateur aussi a l'avantage d'accéder à un espace enseignant : cet espace comporte les mêmes fonctionnalités que celui d'étudiant mais pour manipuler les enseignants.

Et finalement un espace matière: qui fournit les informations sur les matières que chaque enseignant l'est concerné en gardant les mêmes fonctionnalités que les autres espaces.

Pour assurer une bonne utilisation de l'application tous les espaces ont une fenêtre d'aide  pour expliquer les contraintes de chaque formulaire .



Application serveur

● Introduction :

L'application serveur est chargé de fournir les ressources demandées par l'application client et de réaliser ces tâches désirées en faisant appel à un autre serveur de base de données.

● Présentation :

Avant de pouvoir se connecter au base de données il faut lancer l'application serveur qui va faire l'intermédiaire entre l'utilisateur et la base de donnée.

Au côté du l'utilisateur de l'application serveur , il voit une case à cocher "Online DB" qui offre la possibilité de choisir un serveur base de données en ligne ou un autre locale en écrivant l'adresse ipv4  de la machine contenant le serveur base de données locale et une fenêtre contenant une zone de texte qui décrit tout ce qu'il se passe dès l'appui sur le bouton "Start". Ces événements peuvent être exporter  dans un fichier log .

La mise en réseau

Dans notre projet la communication entre les différentes applications est garantie par l'utilisation des sockets TCP/IP.

- Les sockets : sont des outils inventé dans la programmation réseau qui sont le point de communication par lesquels un thread peut émettre ou recevoir des informations sur un réseau informatique.

Ainsi, les sockets TCP/IP permet la communication “end-to-end” entre les applications à travers le réseau d’une manière fiable garantie par l’utilisation de protocole TCP et la distinction entre les machines par les adresses logiques IP.

- IP : est un protocole de la couche réseau dans le modèle OSI permettant d'échanger des paquets de données appelés datagrammes. Ce protocole ne garantit pas l'arrivée à bon port des messages, cette fonctionnalité peut être implémentée par la couche supérieure qui est la couche Transport en utilisant un protocole comme le TCP ainsi utilisé dans ce projet.
- TCP : est un protocole de transport fiable, en mode connecté, assurant le contrôle du flux par l’association

des numéros de séquences pour chaque segment de donnée.

L'application client et l'application serveur communiquent à travers le réseau selon un protocole créé par nous-mêmes qu'on l'appelle 'ACN'.

- ACN : est un protocole inventé par les développeurs de ce projet, il fait partie de la couche application de modèle OSI et il fonctionne selon le principe suivant :
 - L'application client et l'application serveur envoie et reçoit des mots de codes spécifiques au chaque code définit une tâche/état bien précis.
 - L'application client communique uniquement avec l'application serveur qui est le point de liaison entre elle et le serveur de base de données.
 - L'application client envoie une " request code " et les données nécessaires au application serveur qui va :
 - les recevoir
 - détermine les tâches à faire



- créer un “query” spécifique
- communique avec le serveur de base de données en lui donnant le “query”
- reçoit le résultat de l'exécution de la “query” par le serveur de base de données
- Selon le résultat, l'application serveur détermine l'état de traitement (succès ou échec) et elle envoie une “ response code ” bien précis au application client.
- l'application client reçoit la réponse et détermine le message à afficher pour l'utilisateur.
- L'application client doit avoir des messages clairs à la fin de chaque tâche réalisée donc elle a l'avantage de détecter les erreurs trouvées dans les réponses du serveur (en cas de distorsion par exemple) et d'envoyer des messages informatifs à l'utilisateur.
- Le serveur base de données qu'on a choisir est MySQL server, qui est un SGBD relationnel open-source, qui utilise le langage SQL pour manipuler les bases de données et qui possède un numéro de port agréé '3306' pour son service.
- La communication entre l'application serveur et le MySQL server est effectuée par un intermédiaire qui est le JDBC.

- Le JDBC est l'acronyme de Java Data Base Connectivity et il désigne un API pour permettre un accès aux bases de données dans différent SGBD avec Java.
- La classe de pilote pour le MySQL est **com.mysql.cj.jdbc.Driver**

Les mots de codes de A C N :

Request codes	Response codes
+login +add_student +add_teacher +add_subject +show_all_students +show_all_teachers +show_all_subjects +show_students +show_teachers +show_subjects +remove_student +remove_teacher +remove_subject +edit_student +edit_teacher +edit_subject +get_student_pic +get_teacher_pic +get_count +get_student_subjects +get_teacher_subjects	pic_recieved- ok- no_student_removed- no_teacher_removed- no_subject_removed- no_student_edited- no_teacher_edited- no_subject_edited- IO_err- CNF_err- SQL_err- err- ----- Pic : Picture IO : Input Output CNF : Class Not Found SQL : Structured Query Language

-On a affecter le numéro de port '1234' pour le service de l'application serveur.

- L'application serveur joue à la fois le rôle d'un serveur avec l'application client et une application client avec le serveur base de données.
- L'application serveur donne le choix à l'utilisateur s'il veut travailler avec une base de données en ligne ou locale.
 - Si la base de données est locale , il faut donner à l'application serveur l'adresse ipv4  de la machine qui contient le service de MySQL server au vous avez installé la base de données 'administration' dans votre réseau locale .
- L'utilisateur de l'application client doit écrire l'adresse ipv4 de la machine au l'application serveur est installé  .

Remarque :

- Lors de l'écriture de chaque octet d'une adresse ipv4 dans nos applications, il faut appuyer sur la touche 'Entrée' après chaque changement et ça due au principe de fonctionnement de javax.swing.JSpinner ou l'octet ainsi modifié ne vas pas changer.
- Les machines au l'application serveur et le MySQL server ont installées doit avoir respectivement les port 1234 et 3306 ouverts.

-Il faut désactiver tous sortes de Pare-feu lors de l'utilisation de nos applications pour qu'ils puissent fonctionner proprement et ça due à l'utilisation d'un port non agréé qui est 1234.

Sécurité informatique

On a essayé de créer notre projet en respectant les normes de sécurité qu'on a connu.

Notre projet est protégé contre les attaques de type MySQL-Injection grâce à l'utilisation des "PreparedStatement" et les analyses de trafic réseau.

● PreparedStatement :

-Une instruction préparée ou une instruction paramétrée est une fonctionnalité utilisée pour exécuter de manière répétée les mêmes instructions de base de données ou des instructions similaires avec une grande efficacité.

Généralement utilisée avec des instructions SQL telles que des requêtes ou des mises à jour, l'instruction préparée prend la forme d'un modèle dans lequel certaines valeurs constantes sont substituées lors de chaque exécution.

Le flux de travail typique de l'utilisation d'une instruction préparée est le suivant:

Préparer: dans un premier temps, l'application crée le modèle d'instruction et l'envoie au SGBD. Certaines valeurs ne sont pas spécifiées appelées paramètres.

Ensuite, le SGBD compile le modèle d'instruction et stocke le résultat sans l'exécuter.

Exécuter: ultérieurement, l'application fournit des valeurs pour les paramètres du modèle d'instruction, et le SGBD exécute l'instruction (en renvoyant éventuellement un résultat). L'application peut exécuter l'instruction autant de fois qu'elle le souhaite avec des valeurs différentes.

Les instructions préparées résistent à l'injection SQL car les valeurs transmises ultérieurement à l'aide d'un protocole différent ne sont pas compilées comme le modèle d'instruction. Si le modèle d'instruction n'est pas dérivé d'une entrée externe, l'injection SQL ne peut pas se produire.

- **Analyse de trafic réseau :**

-Les données échangées entre l'application client et l'application serveur sont des objets et ça permet de crypter les informations indirectement dans le réseau.

Autrement dit , un Hacker qui va faire une analyse de trafic va trouver des informations illisibles et pour les décrypter il va falloir posséder la propre implémentation des classes d'objets développées dans ce projet , les seules informations qu'il peut décrypter directement sont les mots de codes et les images , si le Hacker va changer les mots de codes les informations restent intouchées dans la base de données car l'application serveur est la seule qui les manipule selon le protocole ACN , toute sorte de changement des données va créer une erreur au niveau de l'application serveur grâce à l'incohérence entre les étapes de l'ACN .

Développement

- L'application client et l'application serveur sont développer en Java 8.
- L'application client et l'application serveur sont des "Threaded applications" ce qui assure une haute performance et une bonne UX
- L'application serveur est une "MultiThreaded server application", ce qui permettre à plusieurs clients de communiquer quasi-instantanément avec l'application serveur
- Le IDE utilisé est le Netbeans 8.0 avec JavaSE 8u241 (JDK 8u241 et JRE 8u241).
- Les interfaces graphiques sont tous développé par nous-même en utilisant le netbeans GUI Builder pour manipuler les différents composants dans le package Javax.swing qui sont :
 - javax.swing.JFrame
 - javax.swing.JPanel
 - javax.swing.JTextField

- javax.swing.JLabel
- javax.swing.JPasswordField
- javax.swing.JSeparator
- javax.swing.JSpinner
- javax.swing.JButton
- javax.swing.JScrollPane
- javax.swing.JTextArea
- javax.swing.JTable
- javax.swing.JComboBox
- javax.swing.JRadioButton
- javax.swing.JCheckBox

Et les composantes d'un API externe com.toedter.calendar qui sont :

- com.toedter.calendar.JDateChooser
- Les API utilisées dans le projets sont :
 - mysql-connector-java-8.0.19
 - jcalendar-1.4
- Tous les fenêtres sont dans le package [App_Frames](#)
- Tous les fenêtres de dialogue sont dans le package Dialogs
- Les icones des fenêtres sont dans le package [App_Frames.icons](#)

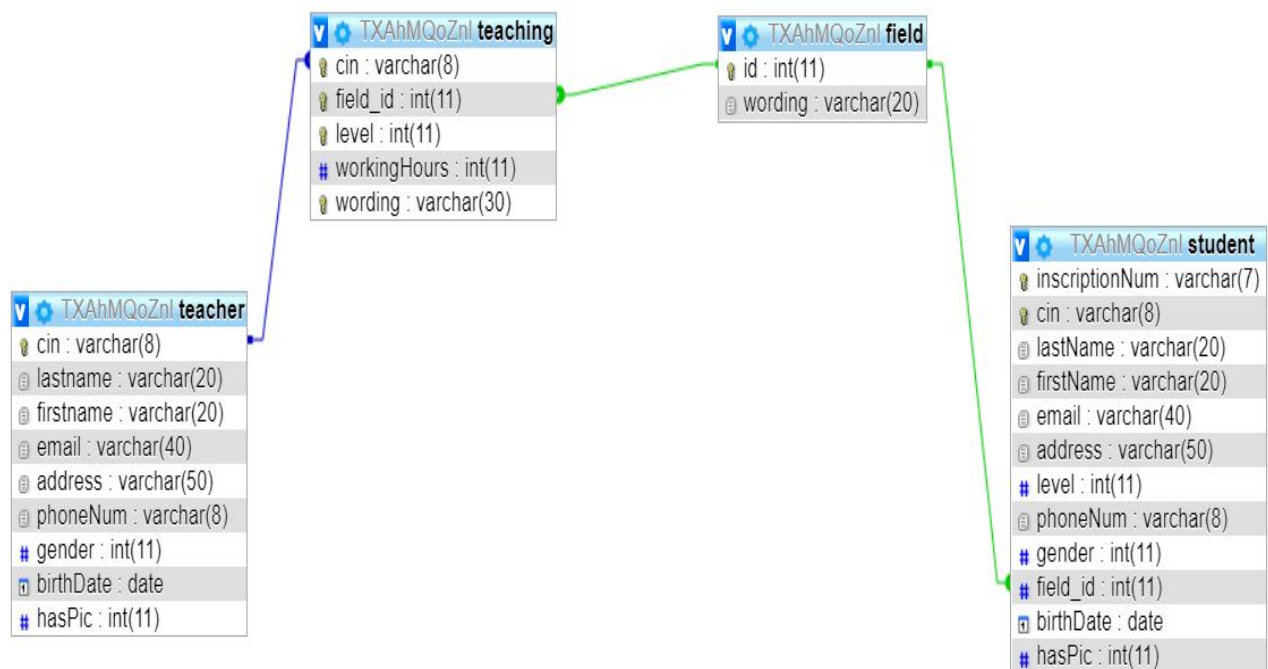
- Les icones des fenêtres de dialogue sont dans le package `Dialogs.icons`
 - L'implémentation de tous les exceptions sont dans le package `App_Exceptions`
- L'implémentation de tous les modèles sont dans le package `Models`
- Tous les fichiers d'extensions 'jar' des API sont dans le package `lib`

Remarque :

- les applications doivent être exécuter avec le JRE 8u241 ou vous devez trouver quelques problèmes d'incompatibilité des composantes de javax.swing , mal résolution des fenêtres, mauvaises qualité des images et des icônes ou un erreur d'incompatibilité lors de l'affichage des images .
- Lors d'un changement de l'implémentation des classes des modèles, il faut mettre à jour le package Models dans l'autre application.

Base de données

- Conception :



Ci-dessus, on ajoute le modèle relationnel de la base de donnée que nous avons créé, pour faciliter la gestion de base de

donnée, on a créé juste trois tables principales qui suffisent pour notre application :

La table “Student” :

Cette table contient les informations nécessaires d’un(e) étudiant(e) et admet le numéro d’inscription comme une clé primaire et quelques informations nécessaires à connaître pour un étudiant .

La table “Field” :

Cette table est constituée seulement de deux attributs, l’un est un clé primaire (id) et l’autre indique le nom de la filière .

Teacher :

Comme l’étudiant, cette table contient les informations nécessaires sur les professeurs et admet le CIN comme une clé primaire.

Relations entre les tables :

Pour réussir à implémenter un modèle relationnel complet, il faut construire des relations entre les tables :

On fait une relation « un à plusieurs » entre “student” et “field” , ça veut dire qu’on peut associer une filière à plusieurs étudiants et un étudiant admet une seule filière.

Donc on a ajouté l’attribut « field_id » dans la table “student” comme une clé étrangère .

On fait aussi une seconde relation qui fait le lien entre la table “field” et la table “teacher” , c’est une relation plusieurs à plusieurs qui sera traduite par une nouvelle table “teaching”.

Cette table contient deux clés étrangères l’une fait la relation avec la table “field” par l’attribut « field_id » et l’autre avec la table “teacher” par l’attribut « cin ».

La table “teaching” admet quatre clés primaires < cin, field_id, wording, level > pour que chaque enseignant peut enseigner plusieurs filières, et pour chaque filière qui l’enseigne, il peut enseigner plusieurs matières de différent niveau.

Remarque :

Vous allez trouver cette implémentation de base de donnée dans un dossier specifié dans le fichier “Read Me” .

Pour utiliser un serveur de base de données locale, vous devez installer un système de gestion de base de donnée MySQL avec l’application de votre choix (phpMyAdmin , MySQL Workbench etc...), et ça veut donner plus de performance et moins de temps d’attente grâce a un accès garantie et locale.

Pour le test seulement en gagnant de temps, on vous recommande d’utiliser la version en ligne pour utiliser les applications directement sans pré requis mais avec moins de performance .