 Application de gestion de notes scolaire de l’ETML

André Gomes / Grégory Poget – Cin2a

EMTL

100 périodes

Gilbert Gruaz – Laurent Deschamps

Table des matières

[1 Spécifications 2](#_Toc308526316)

[1.1 Titre 2](#_Toc308526317)

[1.2 Description 2](#_Toc308526318)

[1.3 Matériel et logiciels à disposition 2](#_Toc308526319)

[1.4 Prérequis 2](#_Toc308526320)

[1.5 Cahier des charges 2](#_Toc308526321)

[1.5.1 Objectifs et portée du projet (objectifs SMART) 2](#_Toc308526322)

[1.5.2 Caractéristiques des utilisateurs et impacts 2](#_Toc308526323)

[1.5.3 Fonctionnalités requises (du point de vue de l’utilisateur) 2](#_Toc308526324)

[1.5.4 Contraintes 2](#_Toc308526325)

[1.5.5 Travail à réaliser par l'apprenti 2](#_Toc308526326)

[1.5.6 Si le temps le permet … 2](#_Toc308526327)

[1.5.7 Méthodes de validation des solutions 2](#_Toc308526328)

[1.6 Les points suivants seront évalués 2](#_Toc308526329)

[1.7 Validation et conditions de réussite 2](#_Toc308526330)

[2 Planification Initiale 2](#_Toc308526331)

[3 Analyse 2](#_Toc308526332)

[3.1 Opportunités 2](#_Toc308526333)

[3.2 Document d’analyse et conception 2](#_Toc308526334)

[3.3 Conception des tests 2](#_Toc308526335)

[3.4 Planification détaillée 2](#_Toc308526336)

[4 Réalisation 2](#_Toc308526337)

[4.1 Dossier de Réalisation 2](#_Toc308526338)

[4.2 Modifications 2](#_Toc308526339)

[5 Tests 2](#_Toc308526340)

[5.1 Dossier des tests 2](#_Toc308526341)

[6 Conclusion 2](#_Toc308526342)

[6.1 Bilan des fonctionnalités demandées 2](#_Toc308526343)

[6.2 Bilan de la planification 2](#_Toc308526344)

[6.3 Bilan personnel 2](#_Toc308526345)

[7 Divers 2](#_Toc308526346)

[7.1 Journal de travail 2](#_Toc308526347)

[7.2 Bibliographie 2](#_Toc308526348)

[7.3 Webographie 2](#_Toc308526349)

[8 Annexes 2](#_Toc308526350)

# Spécifications

## Titre

Gestionnaire de notes ETML

## Description

Pendant le projet de bureautique de première année, les élèves doivent confectionner un fichier Excel avec un tableau qui permet de regrouper les notes scolaires durant l’apprentissage.

Nous avons décidé de ré-implémenté ce fichier au format Android

## Matériel et logiciels à disposition

* PC Etml.
* Téléphone Android.
* Android studio.
* SQLite.
* Outil de Mokeup

## Prérequis

* Modules de programmation
* Module 214

Module 120

A compléter par une description des compétences, des connaissances et de la formation minimum pour être à même de réaliser le projet …

## Cahier des charges

### Objectifs et portée du projet (objectifs SMART)

* Journal de travail.
* Planification et répartition équilibrée des taches.
* Respect des normes ETML.
* Livrer un produit de fonctionnel et de qualité.
* Utilisation du modèle MVC.
* Versioning GitHub.

### Fonctionnalités requises (du point de vue de l’utilisateur)

* Création de la grille de notes
* Mise en forme de base (CFC, Maturité, Formation accélérée)
* Insérer des modules
* Insérer des notes (pondérées)
* Calcule automatique des moyennes
* Affichage des résultats finaux
* Stockage des données en local
* Choix du thème par l’utilisateur (Couleur du fond)
* Initialisation du mot de passe utilisateur

### Contraintes

* Langage Java pas étudié dans les modules
* Prendre en main l’interface Android

### Si le temps le permet …

* Stockage des données en Cloud

### Méthodes de validation des solutions

Comment les tests vont être entrepris, quels tests doivent être entrepris, etc.…

## Les points suivants seront évalués

* Le rapport
* Les planifications (initiale et détaillée)
* Le journal de travail
* Le code et les commentaires
* Les documentations de mise en œuvre et d’utilisation

## Validation et conditions de réussite

* Compréhension du travail
* Possibilité de transmettre le travail à une personne extérieure pour le terminer, le corriger ou le compléter
* Etat de fonctionnement du produit livré

# Planification Initiale

Le projet commence le Lundi 29 mai 2017 le dernier jour pour le rendre avant les présentations est le Lundi 26 Juin 2017

Les périodes de travails sont réparties comme suis :

* Lundi : 8 périodes
* Mardi : 4 périodes
* Mercredi : 9 périodes
* Jeudi : Théorie
* Vendredi : 9 périodes
* Total : 30 périodes par semaines

Il y a un seul jour de congé pendant ce projet, le lundi de pentecôte le 5 juin 2017

Et une semaine du au camp de sport.

# Analyse

## Opportunités

Le projet comporte plein d’opportunité, nous avons la chance de pouvoir découvrir le développement d’application sous Android et un nouveau langage de programmation « Java ». On va aussi pouvoir utiliser comme il le faut GitKraken pour le versionning dans un vrai projet de programmation complet.

Au niveau des difficultés on aura pas mal du fait que nous devons découvrir entièrement le nouveau langage en autonome ce qui risque surement de nous ralentir lors du développement dans des tâches relativement difficiles à mettre en place comme par exemple l’affichage dynamique que nous avons prévus de faire.

Ce paragraphe énumère la liste des difficultés potentielles de tout ordre :

Liste des compétences à acquérir ou approfondir

Liste du matériel à exploiter

Recherche d’informations particulières

Gestion du travail en équipe & collaboration

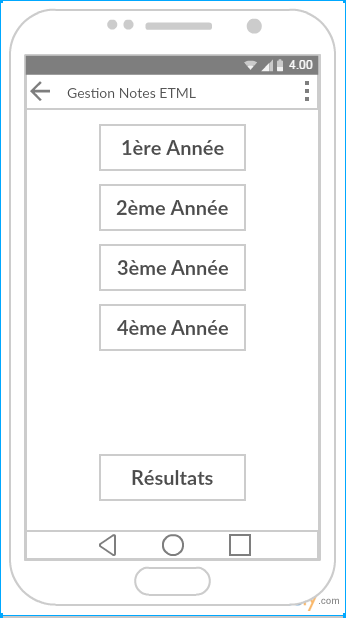
Ainsi que les solutions possibles

Si les spécifications de départ ne laissent pas de doutes sur la manière de réaliser un projet, ce chapitre ne fera que renvoyer le lecteur aux spécifications.

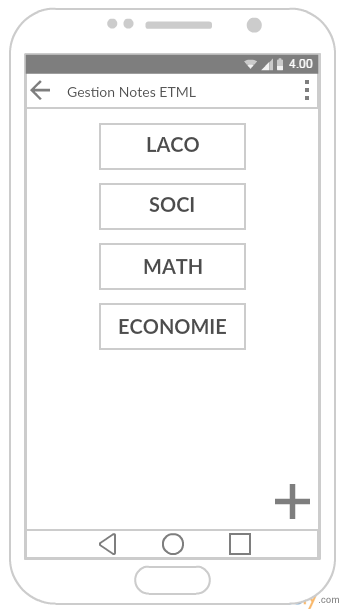
## Document d’analyse et conception

Nous avons décidé de séparer l’application en 4 activités pour pouvoir faire notre application.

Page d’accueil avec les 4 années déjà créer automatiquement avec un bouton qui nous permet d’afficher les résultats finaux de chaque année.



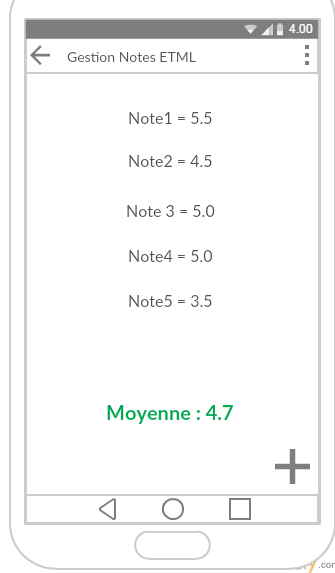
Page qui affiche les branches de chaque année avec dans le menu en haut à droite un onglet de plus qui va permettre d’ajouter des branches.



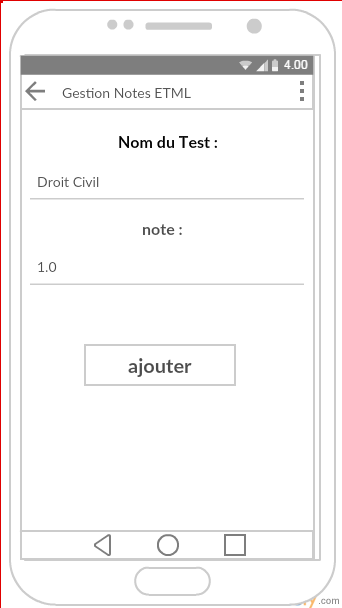
Après avoir cliqué pour ajouter une branche nous aurons un dialogue qui s’ouvre pour nous permettre de créer une branche supplémentaire.



Ensuite nous avons l’affichages des notes de la branche avec la moyenne de la branche tout en bas et dans le menu nous avons la possibilité d’ajouter une note classique et une note pondérée.



Pour l’ajout des notes classiques nous auront un affichage de ce genre-là.



Et voici avec le pourcentage en plus pour les notes pondérées.

Nous pensons aussi créer une activité pour pouvoir modifier certain paramètre dans l’application tel que le thème et le mot de passe de connexion.



Ce paragraphe décrit le fonctionnement de manière détaillée.

Autant que possible de manière graphique, imagée, tableaux, etc.

Tous les cas particuliers devraient y être spécifiés…

Il s’agit d’y présenter les fonctionnalités à développer :

Découpage en étapes, en modules, en fonctionnalités, etc.

Formulaires, interfaces graphiques, pages web, etc.

Schémas de navigation, schémas événementiels, structogramme, pseudocode, etc.

Si le projet inclut une base de données :

Dictionnaire des données

Modèle conceptuel des données, modèles logique des données.

## Conception des tests

Pour la conception des tests vu que nous utilisons Android Studio comme programme nous avons un émulateur de téléphone Android intégrée ce qui nous permet dès l’ajout d’une ligne de code de pouvoir test si le programme ne plante pas et que la ligne fait bien ce que l’on demande. Nous allons aussi faire des tests sur nos téléphones respectifs pour voir si l’affichage et les fonctionnalités ne pose pas de problème.

Tableau ou on va regrouper les tests.

|  |  |  |
| --- | --- | --- |
| Type de test | Réussi ? | Commentaire |
| Lancement de l’application | OUI | Le programme se lance parfois avec une erreur de compilation de l’émulateur |

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.

## Planification détaillée

Fichier ci-dessous « Modele-JDT.xlsm » sous l’onglet Diagramme.

# Réalisation

## Dossier de Réalisation

### Base de donnée

Pour la nécessité du projet nous avons dû créé une base de donnée comportant deux tables. Une pour sauvegarder les informations des boutons créés et une autre pour garder en mémoire les notes de l’utilisateur qu’il rentrera.

Code pour la table Branch :

private String createBranchTable = "CREATE TABLE " + *TABLE\_BRANCH* + "("  
 + *KEY\_ID\_BRANCH* + " INTEGER PRIMARY KEY," + *KEY\_TEXT\_BRANCH* + " TEXT,"  
 + *KEY\_YEAR\_BRANCH*+ " TEXT" + ")";

Code pour la table Note :

private String createMarkTable = "CREATE TABLE " + *TABLE\_MARK* + "("  
 + *KEY\_ID\_MARK* + " INTEGER PRIMARY KEY," + *KEY\_NAME\_NOTE* + " TEXT,"  
 + *KEY\_NOTE\_NOTE* + " TEXT," + *KEY\_YEAR\_NOTE* + " TEXT," + *KEY\_IDBRANCH\_NOTE* + " TEXT" + ")";

Pour pouvoir travailler on a créé plusieurs méthodes pour pouvoir ajouter supprimer et modifier les données ajouter dans la base.

Cette méthode est utile pour pouvoir ajouter les notes dans la bdd

void addMark(MarkClass markClass){  
 SQLiteDatabase db = this.getWritableDatabase();  
  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_NAME\_NOTE*, markClass.getMarName());  
 values.put(*KEY\_NOTE\_NOTE*, markClass.getMarNote());  
 values.put(*KEY\_YEAR\_NOTE*, markClass.getMarYear());  
 values.put(*KEY\_ID\_BRANCH*, markClass.getIdBranch());  
  
 db.insert(*TABLE\_MARK*, null, values);  
 db.close();  
}

En premier on ouvre la base en mode écriture puis on va récupérer les valeurs entrées dans le constructeur de la classe MarkClass pour pouvoir les entrées dans la variable « values » qui va stocker les informations grâce à la méthode insert puis en referme la base.

void addBranch(BranchClass branchClass){  
 SQLiteDatabase db = this.getWritableDatabase();  
  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_ID\_BRANCH*, branchClass.getIdBranch());  
 values.put(*KEY\_TEXT\_BRANCH*, branchClass.getBraText());  
 values.put(*KEY\_YEAR\_BRANCH*, branchClass.getBraYear());  
  
 db.insert(*TABLE\_BRANCH*, null, values);  
 db.close();  
}

Cette méthode fait la même chose que celle présenter au-dessus mais pour la classe Branche qui stock les informations des boutons.

Pour pouvoir afficher les notes ou les branches nous avons dû créé une méthode qui doit récupérer grâce à une requête SQL qui parcours toute la base de donnée et qui stock dans une liste les informations qu’il va récupérer.

public List<MarkClass> getAllMark(String year, String branch) {  
 List<MarkClass> markList = new ArrayList<>();  
  
 String selectQuery = "SELECT \* FROM " + *TABLE\_MARK* + " WHERE " + *KEY\_YEAR\_NOTE* + " = " + year + " AND " + *KEY\_IDBRANCH\_NOTE* + " = " + branch;  
  
 SQLiteDatabase db = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, null);  
  
 if(cursor.moveToFirst()) {  
 do {  
 MarkClass markClass = new MarkClass();  
 markClass.setIdMark(Integer.*parseInt*(cursor.getString(0)));  
 markClass.setMarName(cursor.getString(1));  
 markClass.setMarNote(cursor.getString(2));  
 markClass.setMarYear(cursor.getString(3));  
 markClass.setIdBranch(cursor.getString(4));  
  
 markList.add(markClass);  
 }while (cursor.moveToNext());  
 }  
 return markList;  
}

Dans notre cas nous avons fait en sorte que la requête SQL trie selon les l’année et l’id de la branche que nous récupérons quand nous passons d’activité en activité pour pouvoir seulement sélectionner les notes que nous voulons.

Pour pouvoir gérer les notes nous avons fait une méthode qui permet de supprimer une note comme ça on peut supprimer des notes ou des boutons si l’utilisateur se trompe lors de l’insertion.

public void deleteMark(MarkClass markClass){  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(*TABLE\_MARK*,*KEY\_ID\_MARK* + "=?", new String[] {String.*valueOf*(markClass.getIdMark())});  
  
 db.close();  
}

Le code récupère juste l’intégralité de l’entité et va supprimer l’entité en se basant sur son ID.

### Option

### Login

### Accueil

### Branche

### Note

### Moyenne

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.

Arborescences des documents produits.

Il faut décrire le parcours de réalisation et justifier les choix.

## Modifications

Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.

Date, raison, description, etc.

# Tests

## 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 obtenus et 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.

# Conclusion

## 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.

## 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 différence entre les planifications initiales et détaillées avec le journal de travail.

## 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 que ce projet m’a appris ?

Suite à donner, améliorations souhaitables, …

Remerciements, signature, etc.

# Divers

## 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.

## Bibliographie

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

## Webographie

Références des sites Internet consultés durant le projet.

# Annexes

[H:\DemoMot\AGS-GPT-DemoMot\Modele-JDT.xlsm](file:///H:\DemoMot\AGS-GPT-DemoMot\Modele-JDT.xlsm)

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.