GILLIOEN François
GORLIN Tristan
GUIBERT Nicolas

Rapport de projet
Evaluations/Elèves/Professeurs

GILLIOEN François
GORLIN Tristan
GUIBERT Nicolas

Table des matières

Introduction

Dans le cadre du module de Java 2, nous devons réaliser un projet pour mettre en pratique toutes les méthodes que nous avons apprises. Le projet Evaluations/Elèves/Professeurs est un projet divisé en quatre grandes parties ou versions permettant d'avoir un rendu de plus en plus abouti.

La première version contenant les fonctions de base du projet, et la seconde version présente l'utilisation d'un fichier csv afin de sauvegarder les opérations enregistrées durant les phases de tests. La troisième version nécessite l'utilisation de librairies externes pour pouvoir créer des bulletins au format PDF ainsi que des statistiques sur les notes. Enfin la dernière version reprend toutes les fonctions précédentes mais avec une interface graphique.

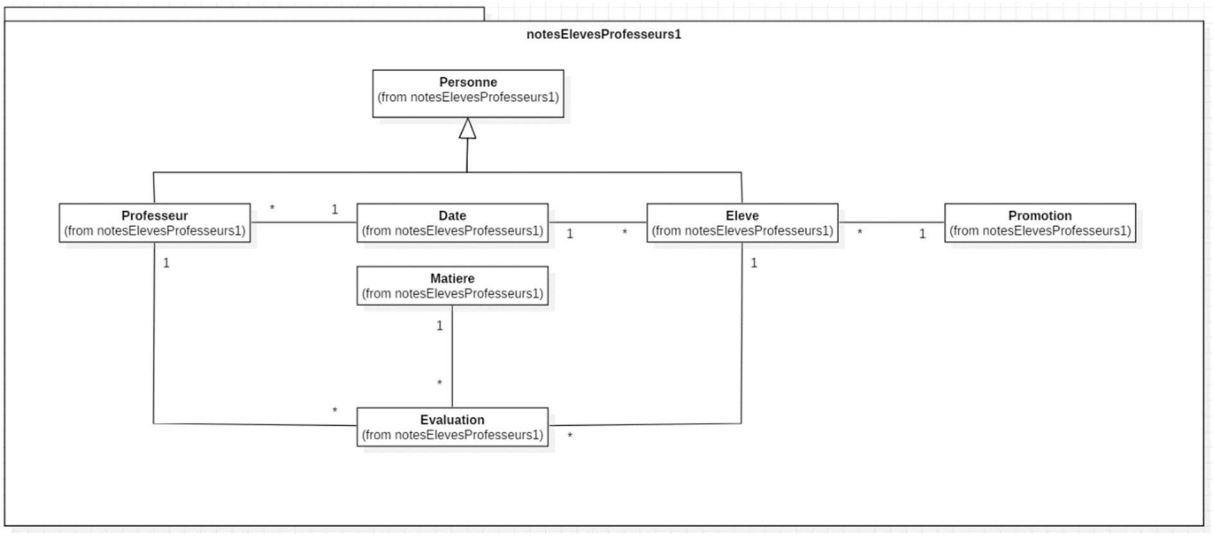
Ce projet nous a permis de soulever différentes problématiques. Nous avons notamment dû nous organiser pour pouvoir travailler en équipe de trois et gérer les différentes versions du code. Afin de mener à terme ce projet, nous avons utilisé différents outils pour simplifier la gestion du projet et sa conception. Dans ce rapport nous allons tenter de retranscrire au mieux la façon dont nous avons avancé dans ce projet ainsi que les difficultés rencontrées.

GILLIOEN François
GORLIN Tristan
GUIBERT Nicolas

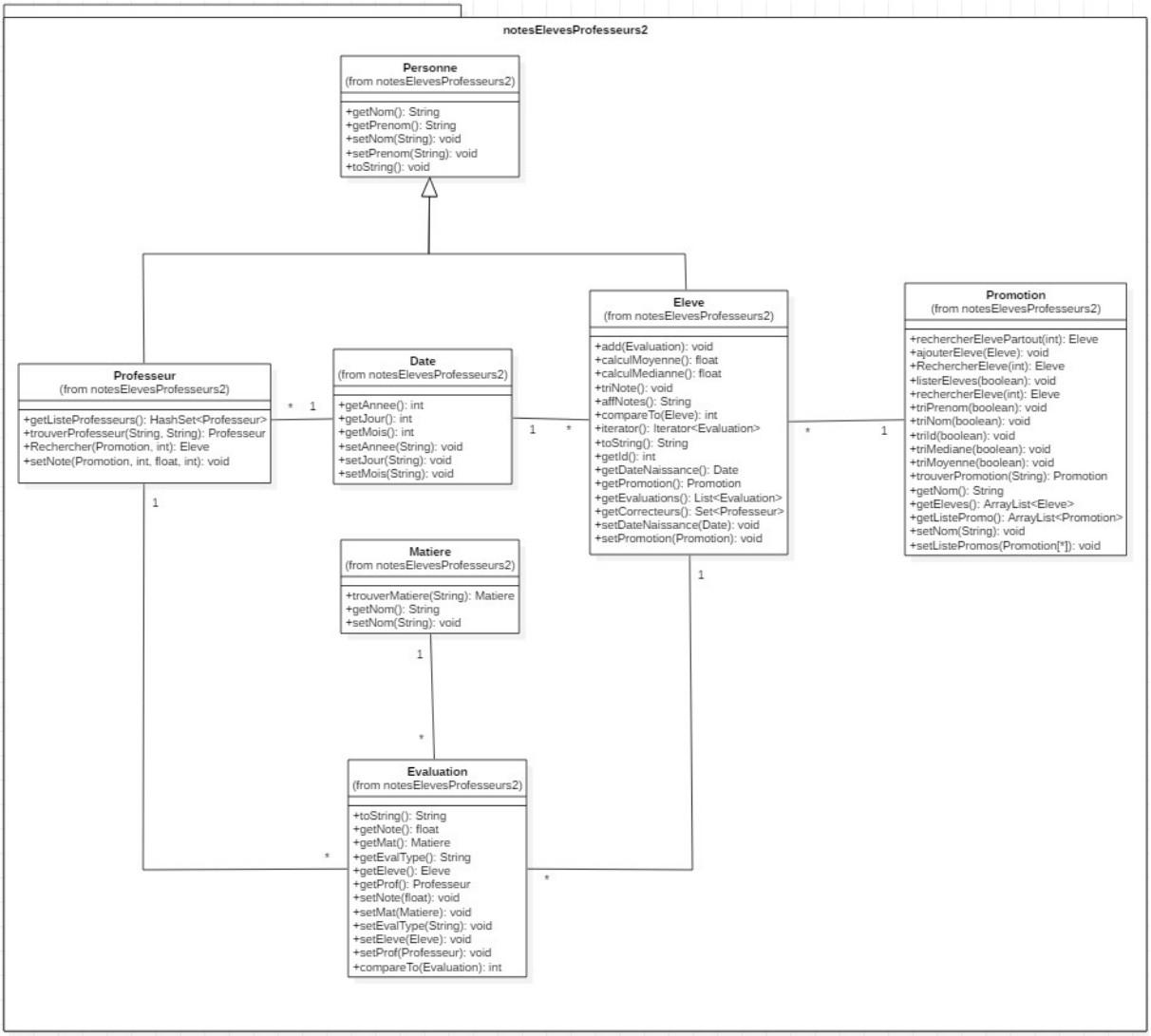
I. Diagrammes de classe

Afin d'avoir un bon suivi de notre projet nous avons utilisé le logiciel starUml pour créer nos diagrammes de classe. Ces diagrammes seront axés sur nos classes principales (élève, professeur, évaluation, etc…). Nous avons choisi de ne pas faire figurer les classes pour l'affichage en console ou graphique dans un souci de clarté. Chaque version du diagramme de classe sera de plus en plus précise.

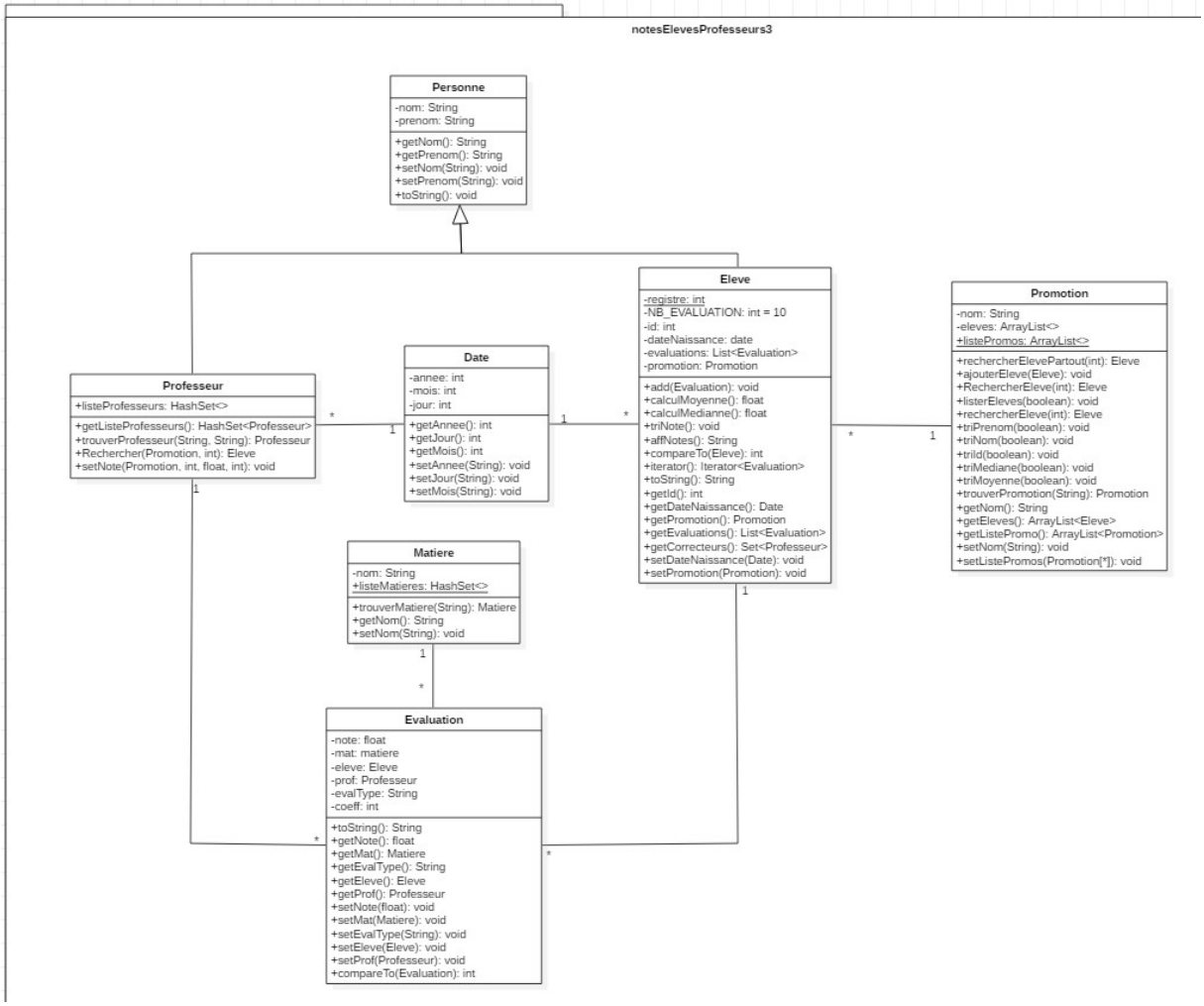
Le premier diagramme représente les relations entre les classes principales. D'après les consignes de l'énoncé, nous avons nos deux classes Professeur et Eleve qui héritent de la classe plus générale Personne. Ces deux classes sont aussi liées à une classe Date afin de gérer les dates de naissance. De plus la classe Professeur a une relation avec la classe Evaluation pour qu'un professeur puisse attribuer une note à un élève. D'autres part la classe Eleve est aussi liée à la classe Evaluation afin d'attribuer une note à un élève. Une Evaluation se voit attribuer une Matière tandis qu'une Matière peut être reliée à plusieurs Evaluation. Enfin pour gérer le système de promotions nous avons créé une classe Promotion qui peut contenir 0 ou plusieurs Eleve.



Le second diagramme de classe nous permet d'avoir une idée un peu plus précise de notre projet. En effet, dans celui-ci nous rajoutons les principales fonctions publiques de chaque classe. Les fonctions les plus intéressantes autres que les accesseurs, mutateurs et constructeurs se trouvent notamment dans la classe Promotion. Elles permettent de trier les élèves selon différents critères des Comparator<>. Nous effectuons aussi ce tri dans la classe Eleve pour classer les notes. Cela nous permet de calculer la médiane de manière plus simple.



Enfin dans le dernier diagramme nous ajoutons les attributs de chaque classe.



II. Analyse Fonctionnelle

a. Analyse fonctionnelle générale

On peut diviser notre programme en trois grandes parties :

- Une première contenant toutes les classes principales avec les fonctions nécessaires au bon fonctionnement du projet. Ce module peut se décomposer en 4 sous parties : la gestion des élèves (création, suppression, modification). Les évaluations qui varient selon la note, la matière, le professeur qui les corrigent. Les options de tri de promotion permettant de trier les élèves selon leurs notes, nom, identifiants, etc… Et enfin les professeurs qui peuvent s'occuper de gérer les promotions et les notes. De plus nous pouvons ajouter dans cette partie la gestion des fichiers CSV permettant de sauvegarder les données lorsqu'on ferme l'application afin ne pas perdre les informations déjà enregistrées. Nous avons donc créé une classe contenant toutes les fonctions pour gérer nos deux fichiers CSV : 'Elèves.csv' et 'Résultat Elève.csv'. Le premier contenant les informations sur les élèves, le second celles sur les notes et les enseignants.

Grâce à ce premier module nous avons pu commencer à développer le second axé sur l'affichage. Pour cela nous avons créé un nouveau package contenant tous les fichiers de l'affichage graphique. Nous avons utilisé java swing pour créer notre interface. Pour simplifier la création de fenêtres nous nous sommes servis du module de création graphique de NetBeans. Ce module permet de générer automatiquement le code lorsqu'on place un élément (par exemple JTable,JTextField, etc…) dans notre frame. Dans notre dossier GUI, nous pouvons distinguer deux grandes catégories de fichiers : une première servant à l'affichage des fenêtres avec la mise en forme, les ActionListener, etc… et une seconde catégorie correspondant aux fichiers Operation. Ces fichiers contiennent toutes les fonctions pour traiter les données et les renvoyer sous la forme souhaitée dans notre affichage. Ce procédé nous a permis de séparer le traitement des données de l'affichage et ainsi d'avoir un code plus compréhensible.

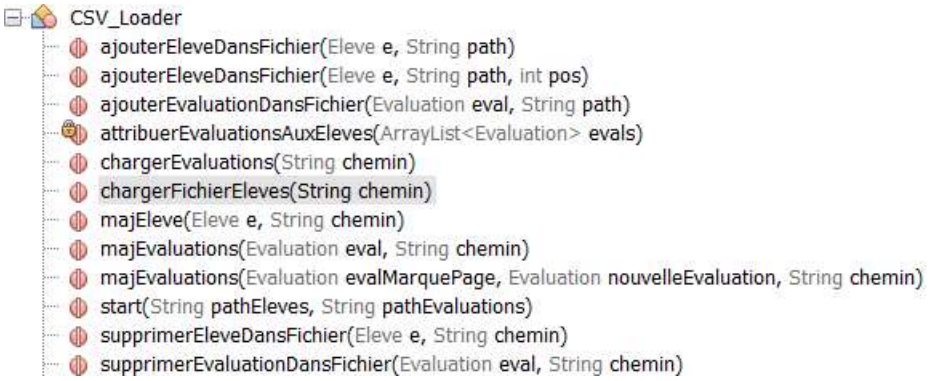
Enfin, la troisième grande partie est spécifique aux graphes et bulletins. En effet, il nous est demandé de créer un fichier Bulletin.pdf pour chaque élève. Pour se faire nous avons dû utiliser des librairies externes comme JFreeChart et Itext. Nous avons décidé de le mettre en tant que nouveau module car même s'il est lié au reste du projet il n'en reste pas moins assez spécifique. Il était donc plus simple pour nous de le voir comme une troisième partie à implémenter au projet. Nous avons également décidé d'utiliser Maven, un outil très pratique dans le développement en Java lorsque certaines de nos librairies ont de nombreuses dépendances. En effet, grâce à Maven, nous devons juste modifier un fichier .xml pour indiquer quelles librairies et quelles versions nous utilisons, ainsi le téléchargement se fait automatiquement. Cela est surtout utile pour Itext, la bibliothèque que nous utilisons pour générer les bulletins en PDF, qui nécessite de nombreux autres packages pour fonctionner.

b. Analyse Fonctionnelle détaillée

i. Utilisation d'un fichier CSV

Fonctions CSV :

Afin de répondre à tous les besoins possibles en termes de création, modification et suppression de données, nous avons créé 12 méthodes réalisant ces opérations.



Comme nous pouvons le voir, chaque fonction correspond à une opération.

Conception des algorithmes :

Nous considérons les algorithmes CSV comme « puissants » d'après nos observations ci-dessus, nos méthodes prennent uniquement des objets en argument au lieu de modifier une ligne en particulier à l'aide d'un index. Cela procure selon nous deux avantages :

1) L'impossibilité de se tromper sur la ligne à remplacer

2) La facilité d'utilisation des méthodes pour le développeur

Structure des fichiers CSV :

Nous avons 2 fichiers CSV : « élèves.csv » et « Résultats élèves.csv » situés dans un dossier intitulé data.

Colonnes du fichier « élèves.csv » :

Identifiant	Nom	Prénom	Promotion	Date de naissance

Colonnes du fichier « Résultats élèves.csv » :

Identifiant	Note	Matière	Correcteur	Type	Coeff

Les deux fichiers renvoient au même identifiant qui est celui de l'élève. Ils ont été conçus de sorte à être au maximum « user-friendly » pour des utilisateurs voulant remplir à la main ces fichiers. En effet, à la différence des bases de données en SQL où il faudrait relier de nombreuses tables entre elles par des identifiants, nous avons créé un minimum de tables afin de rendre plus simple pour l'utilisateur.

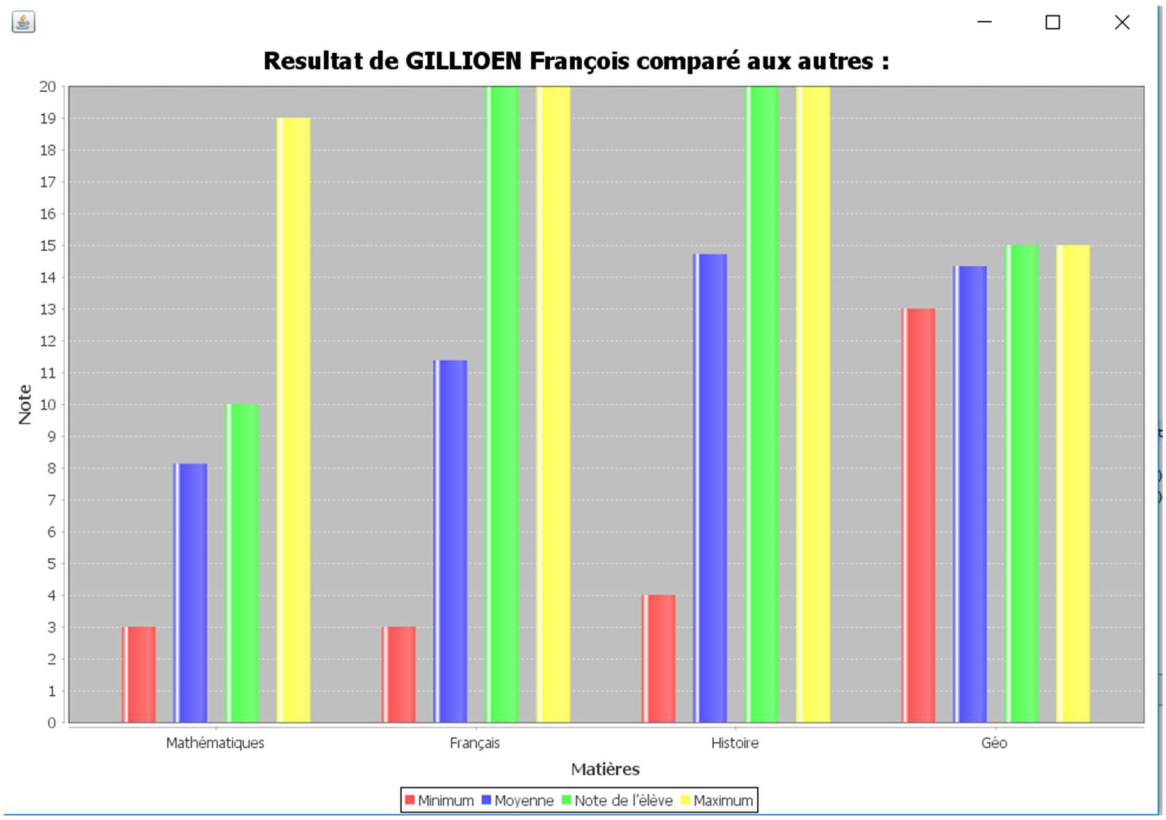
De ce fait, au lieu d'avoir un fichier Matière et un fichier Professeur, les professeurs et les matières sont créés automatiquement au fur et à mesure que nous parcourons les évaluations du fichier « Résultats élèves.csv ».

La fonction majEvaluation (Evaluation evalMarquePage, Evaluation nouvelleEvaluation, String chemin) a été plus difficile à programmer. En effet, aucune des évaluations ne possèdent d'identifiant. Dès lors, la modification d'une évaluation semble plus compliquée car il faut pouvoir trouver l'évaluation d'origine à partir de l'évaluation modifiée, leur comparaison doit être différente.

Pour cela, nous comparons par rapport à une première évaluation evalMarquePage quelle évaluation doit être modifiée (c'est une copie de l'évaluation juste avant sa modification dans l'application). Puis, une fois que la bonne évaluation est trouvée dans le fichier CSV, nous pouvons la remplacer avec l'évaluation nouvelleEvaluation.

ii. Jfreechart et son fonctionnement

JFreechart est une API en java qui nous permet de générer des graphiques ou des diagrammes. Cette API s'utilise parallèlement avec une bibliothèque graphique, dans notre cas Swing. L'API est Open-source et comporte une excellente documentation sur le site. Il est donc aisé de générer plusieurs types de diagrammes, dont en voici des exemples :



Le fonctionnement de JFreeChart est assez simple :

On crée une variable de type DefaultCategoryDataset auquelle on ajoute des données avec une méthode “addValue”. Ensuite, il nous reste juste à générer ce

que l'on désire à l'aide la méthode " ChartFactory.create ", et à ajouter l'élément créé au JPanel de notre fenêtre.

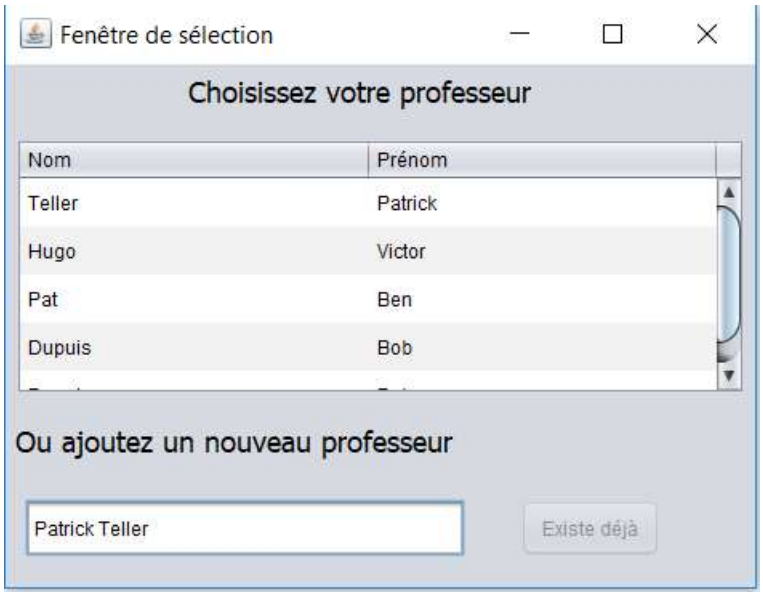
Cela nous permet de réaliser assez facilement différents types de diagrammes, et nous avons ainsi créé une classe par graphique que nous souhaitons. Afin de générer les diagrammes, nous avons juste à instancier la classe en passant en paramètre ce que la classe demande pour fonctionner. Cela s'intègre parfaitement avec l'interface graphique également, car il suffit de juste lier un bouton à la classe pour la faire fonctionner.

iii. Itext, générer des fichiers en PDF très facilement

Pour créer nos Bulletins, nous utilisons une bibliothèque nommée Itext. Cependant, lors de l'installation d'Itext, nous avons rapidement compris que cela causerait des problèmes au niveau des dépendances. Après différentes recherches, nous avons décidé d'utiliser Maven. En effet, en modifiant le fichier Pom.xml (Project Object Model), nous définissons ce qu'il faut télécharger pour que le code fonctionne.

iv. Autres fonctionnalités

Nous avons utilisé des expressions régulières pour vérifier les dates de naissance des élèves ainsi que des fonctions de saisie sécurisé comme SecureIntInput() qui demande une saisie console et redemande la saisie tant que la saisie n'est pas un entier. Nous avons ajouté des fonctionnalités de sécurité telles que le selectionneur de matières et de correcteurs pour une évaluation, cela permet de créer un professeur ou une matière tout en laissant le choix d'en choisir un existant



Comme nous pouvons le voir si dessus, nous pouvons ajouter des nouveaux professeurs à la liste, et si le professeur à créer est déjà présent le bouton se verrouillera et il aura pour texte « existe déjà »

François

GILLIOEN

Promotion 2021

BULLETIN DE NOTE

Résultats						
Matière	Note	Moyenne Promo	Mediane Promo	Minimum Promo	Maximum Promo	Correcteur
Mathématiques CE	10.0	6.6666665	7.0	3.0	10.0	Teller
Français CE	20.0	11.5	11.5	3.0	20.0	Hugo
Histoire CE	20.0	15.25	18.5	4.0	20.0	Dupuis
Géo CE	15.0	15.0	15.0	15.0	15.0	Pat

Moyenne générale : 16.25 Validé mention très bien

Moyenne de la Promotion : 4.9791665

Minimum de la Promotion : 3.0

Maximum de la Promotion : 20.0

III. Difficultés rencontrées

Tout au long de notre projet nous avons rencontré différents obstacles. L'un des plus importants que nous avons eu est celui de la compatibilité. En effet nous n'utilisons pas le même IDE (netBeans ou Intelij) pour programmer et cela peut provoquer quelques erreurs de compatibilité. Il était donc parfois compliqué de mettre à jour nos projets ou d'implémenter les nouvelles librairies.

Nous avons aussi rencontré des erreurs d'encodage selon les IDE ce qui avait pour conséquence de changer le contenu de nos phrases. Cela pouvait même provoquer un grand nombre d'erreurs.

Enfin, certains membres du trinôme n'étaient pas très à l'aise avec l'outil Git. Or celui-ci est maintenant quasi indispensable au bon déroulement d'un projet de programmation. Nous avons donc dû faire une remise à niveau sur les connaissances de base de Git afin de pouvoir travailler dans les meilleures conditions.

IV. Conclusion

Finalement ce projet nous a permis de mettre en pratique toutes nos connaissances sur le langage JAVA et la programmation orientée objet. Nous avons dû apprendre à bien répartir les tâches afin d'optimiser notre travail. Cela nous a aussi permis d'améliorer nos compétences avec Git.

Pour nous repartir le travail nous avons décidé de travailler sur les classes principales ensemble pour avoir des bases solides. Puis nous nous sommes réparti le travail en fonction des différentes versions demandées.

Ce projet fût très intéressant pour nous car deux des trois membres du groupe souhaitent intégrés la majeur *Software Engineering* lors de notre quatrième année à l'EFREI.

Il nous a aussi permis d'améliorer nos compétences en Java, notamment sur la gestion de fichier CSV ou PDF, l'affichage graphique ou encore la création de graphes ce qu'il l'a rendu d'autant plus stimulant

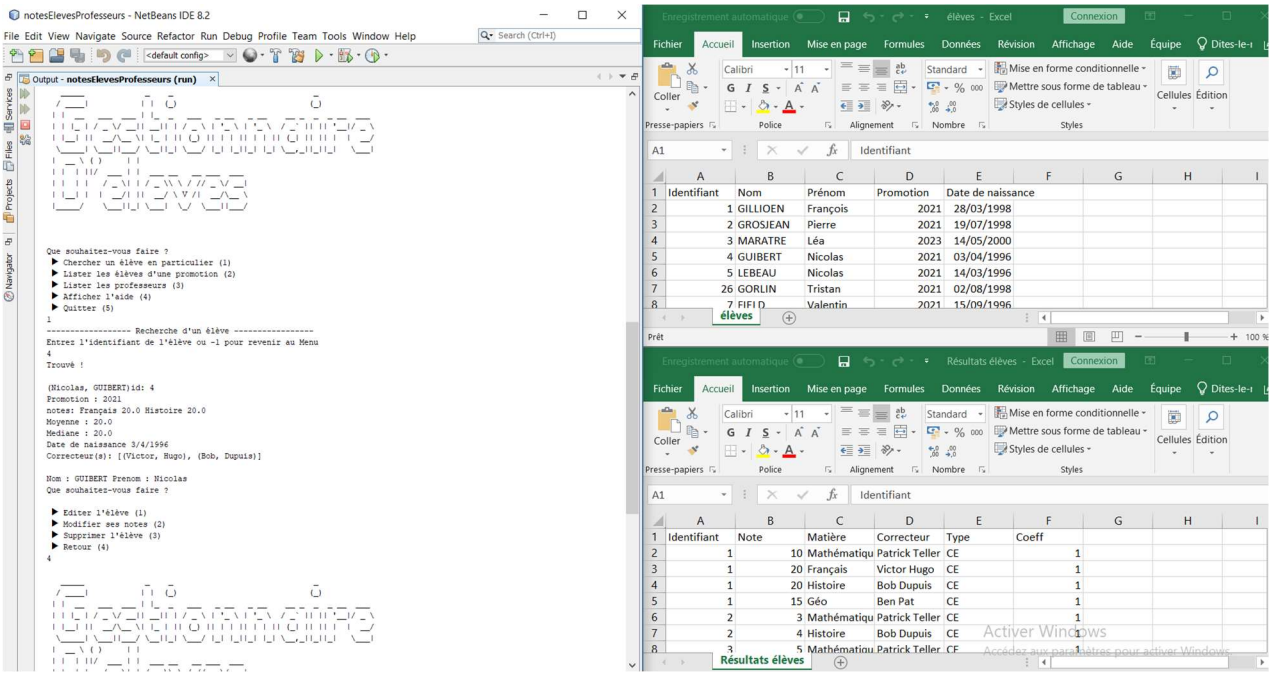
V. Annexes

Captures d'écrans de ce que l'on peut obtenir à travers les différentes versions :

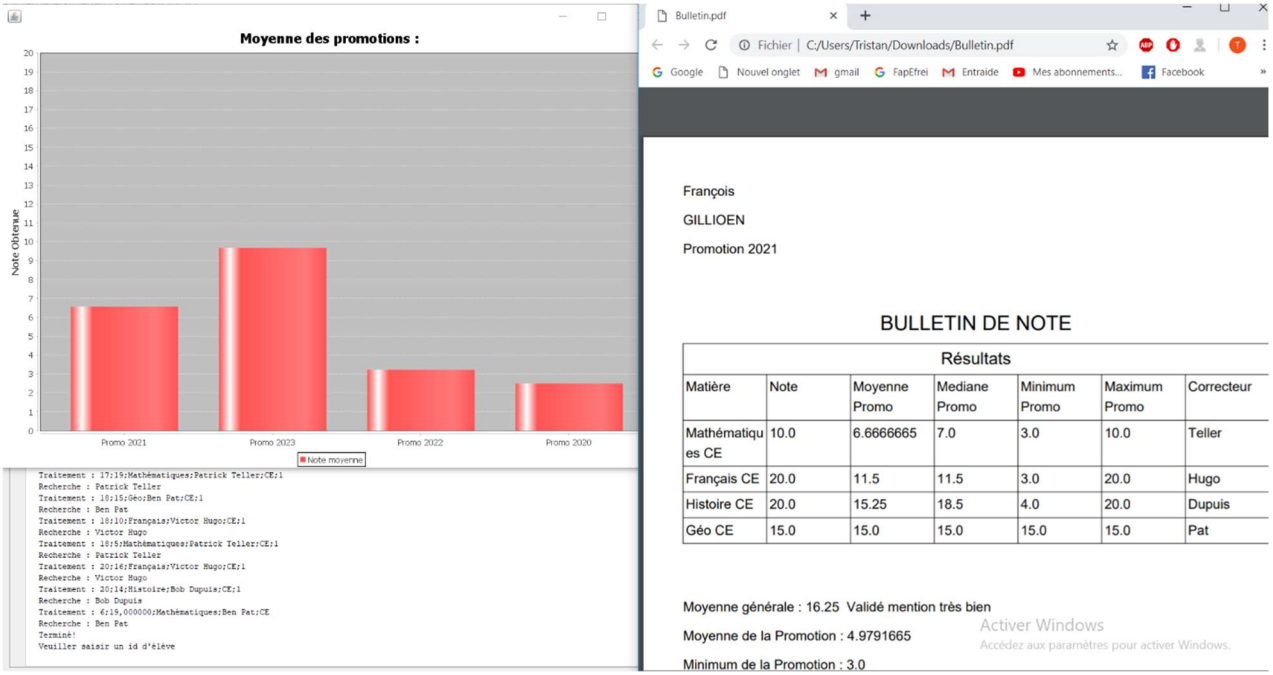
Version 1 :

```
run:
Version 1
(Guibert, Nicolas)id: 4
Promotion : 2021
notes: maths 20.0 histoire 19.0
Moyenne : 19.5
Mediane : 19.5
Date de naissance 2/4/1998
Correcteur(s): [(Historien, ), (Patrick, Teller)]

Affichage des élèves pour la promotion : 2021
------------------------------
ID : 2|Gillioen|François
----------------------------------
ID : 3|Gorlin|Tristan
----------------------------------
ID : 4|Nicolas|Guibert
----------------------------------
Affichage des élèves pour la promotion : 2023
------------------------------
ID : 1|Maratre|Lea
----------------------------------
(Guibert, Nicolas)id: 4
Promotion : 2021
notes: histoire 19.0 maths 20.0
Moyenne : 19.5
Mediane : 19.5
Date de naissance 2/4/1998
Correcteur(s): [(Historien, ), (Patrick, Teller)]

------------------- Promotion 2021 -------------------
Tri croissant moyenne
Affichage des élèves pour la promotion : 2021
------------------------------
(François, Gillioen)id: 2
Promotion : 2021
notes: maths 9.0 maths 11.0
Moyenne : 10.0
Mediane : 10.0
Date de naissance 28/3/1998
Correcteur(s): [(Patrick, Teller)]
```

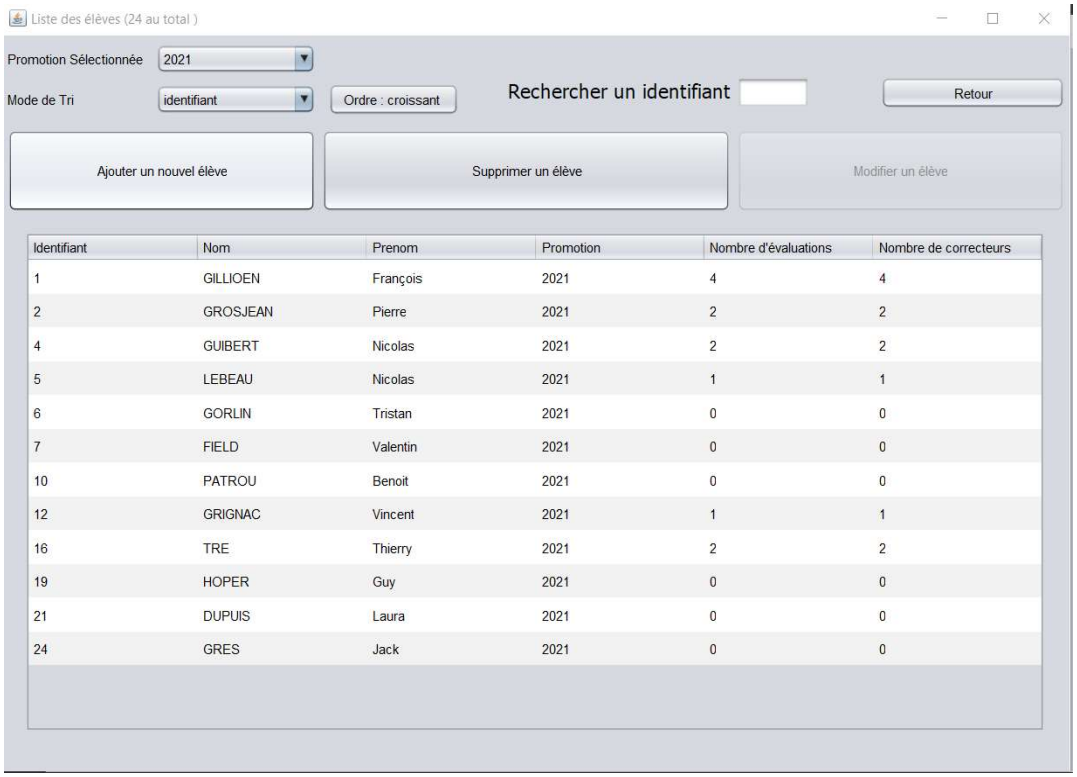
Version 2 :

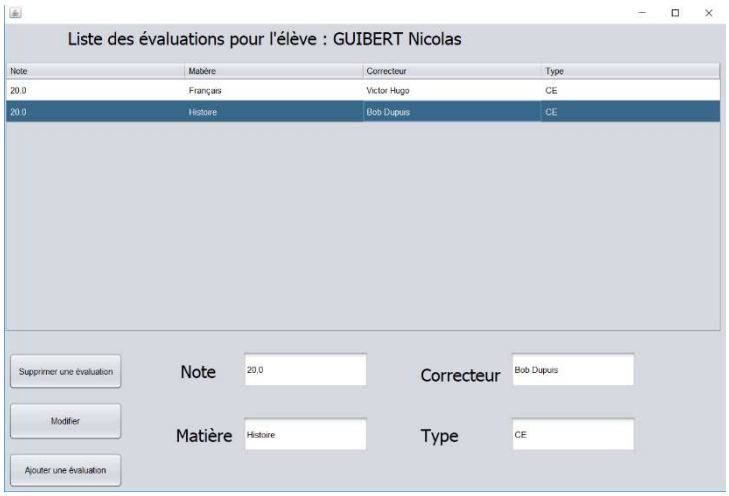


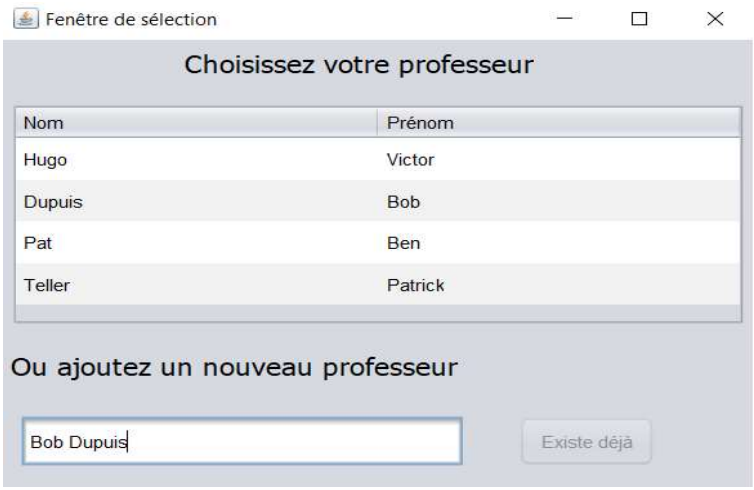
Version 3 :


Moyenne des promotions :
Note Obtenue
Promo 2021
Promo 2023
Promo 2022
Promo 2020
Note moyenne
Traitement : 17;19;Mathématiques;Patrick Teller;CE;1
Recherche : Patrick Teller
Traitement : 18;15;Géo;Ben Pat;CE;1
Recherche : Ben Pat
Traitement : 18;10;Français;Victor Hugo;CE;1
Recherche : Victor Hugo
Traitement : 18;5;Mathématiques;Patrick Teller;CE;1
Recherche : Patrick Teller
Traitement : 20;16;Français;Victor Hugo;CE;1
Recherche : Victor Hugo
Traitement : 20;14;Histoire;Bob Dupuis;CE;1
Recherche : Bob Dupuis
Traitement : 6;19,000000;Mathématiques;Ben Pat;CE
Recherche : Ben Pat
Terminé!
Veuiller saisir un id d'élève
Bulletin.pdf
Fichier | C:/Users/Tristan/Downloads/Bulletin.pdf
Google
Nouvel onglet
gmail
FapEfrei
Entraide
Mes abonnements...
Facebook
François
GILLIOEN
Promotion 2021
BULLETIN DE NOTE
Résultats
Matière | Note | Moyenne Promo | Mediane Promo | Minimum Promo | Maximum Promo | Correcteur
Mathématiques CE | 10.0 | 6.6666665 | 7.0 | 3.0 | 10.0 | Teller
Français CE | 20.0 | 11.5 | 11.5 | 3.0 | 20.0 | Hugo
Histoire CE | 20.0 | 15.25 | 18.5 | 4.0 | 20.0 | Dupuis
Géo CE | 15.0 | 15.0 | 15.0 | 15.0 | 15.0 | Pat
Moyenne générale : 16.25 Validé mention très bien
Moyenne de la Promotion : 4.9791665
Minimum de la Promotion : 3.0
Activer Windows
Accédez aux paramètres pour activer Windows.

Version 4 :


Liste des élèves (24 au total)
Promotion Sélectionnée
2021
Mode de Tri
identifiant
Ordre : croissant
Rechercher un identifiant
Retour
Ajouter un nouvel élève
Supprimer un élève
Modifier un élève
Identifiant | Nom | Prenom | Promotion | Nombre d'évaluations | Nombre de correcteurs
1 | GILLIOEN | François | 2021 | 4 | 4
2 | GROSJEAN | Pierre | 2021 | 2 | 2
4 | GUIBERT | Nicolas | 2021 | 2 | 2
5 | LEBEAU | Nicolas | 2021 | 1 | 1
6 | GORLIN | Tristan | 2021 | 0 | 0
7 | FIELD | Valentin | 2021 | 0 | 0
10 | PATROU | Benoit | 2021 | 0 | 0
12 | GRIGNAC | Vincent | 2021 | 1 | 1
16 | TRE | Thierry | 2021 | 2 | 2
19 | HOPER | Guy | 2021 | 0 | 0
21 | DUPUIS | Laura | 2021 | 0 | 0
24 | GRES | Jack | 2021 | 0 | 0





Sources utilisées :

https://itextpdf.com/fr

https://www.supinfo.com/articles/single/1187-creer-pdf-avec-itext

http://www.jfree.org/jfreechart/

https://thierry-leriche-dessirier.developpez.com/tutoriels/java/afficher-graphe-jfreechart-5-min/

https://docs.oracle.com/javase/tutorial/uiswing/

https://www.geeksforgeeks.org/comparable-vs-comparator-in-java/

Ressources Efrei Moodle

Open Classroom