TASK MANAGEMENT

ANAS ICHMAWIN
IMANI MOURAD
MOHAMED-AMINE FATIH
SOHAIL MOUSSALIM

JAVA PROJECT

ENCADRANT:
M.IMADE HAFIDI

INTRODUCTION

La gestion efficace des tâches et des projets académiques représente un défi majeur pour les professeurs. Avec la multiplicité des responsabilités, allant de l'enseignement à l'encadrement des projets de fin d'études et des thèses, il devient crucial de disposer d'un outil performant pour organiser et suivre ces activités. C'est dans ce contexte que s'inscrit notre projet : développer une application de suivi des projets et tâches académiques.

Cette application a pour vocation d'aider les professeurs à mieux gérer leurs différentes tâches en les structurant sous forme de projets et de listes de tâches. Chaque projet peut inclure diverses activités telles que des cours, des thèses, des PFE (Projets de Fin d'Études), ou encore des tâches administratives. L'application permet non seulement de planifier et de suivre ces projets, mais aussi d'importer des événements depuis Google Calendrier, offrant ainsi une intégration fluide avec les outils existants.

Les fonctionnalités principales de l'application incluent la gestion des projets et des tâches, la consultation et la modification des informations associées, la gestion des séances de travail, et la capacité de filtrer et de rechercher des projets et des tâches par différents critères.

Techniquement, cette application a été développée en Java, utilisant JavaFX pour l'interface graphique, et MongoDB pour le stockage des données. Tous les paramètres de configuration sont stockés dans un fichier de propriétés, assurant ainsi une flexibilité et une facilité d'adaptation aux besoins spécifiques des utilisateurs. L'application est conçue pour être multiplateforme, avec des fichiers d'installation disponibles pour Windows, Mac, et Linux.

En conclusion, cette application se veut être un outil complet et intuitif, capable de répondre aux besoins de gestion des tâches académiques des professeurs, tout en leur offrant des fonctionnalités avancées pour optimiser leur temps et leur efficacité.

Les fonctionnalités de l'application

Le logiciel fonctionnera de la manière suivante :

1. Il permettra à l'utilisateur de se connecter avec une adresse e-mail Google.
2. Il autorisera l'utilisateur à naviguer entre les projets existants et à les ordonner/filtrer par catégorie ou type.
3. Il offrira à l'utilisateur l'option de recherche en utilisant les mots-clés existants dans la description.
4. Le logiciel offrira la possibilité :
    a. d'ajouter un projet.
    b. de clôturer un projet.
    c. de cloner un projet.
    d. de modifier les informations d'un projet.
5. Il permettra à l'utilisateur de naviguer entre les différentes tâches.
6. Il offrira l'option de :
    a. création d'une tâche.
    b. modification de la description.
    c. ajout d'une tâche existante.
    d. suppression d'une tâche :
7. si l'utilisateur supprime une tâche, la tâche supprimée de la liste restera toujours existante.
8. si l'utilisateur supprime une liste de tâches, les tâches dans cette liste ne seront pas supprimées.
9. L'utilisateur peut consulter les tâches ordonnées par :
    a. Ordre alphabetique
    b. Date (DD/MM/AAAA)
10. Il va permettre à l'utilisateur l'option de recherche sur les tâches en utilisant les mots-clés existants dans la description.
11. Il va permettre à l'utilisateur d'ajouter, cloner ou modifier les informations des tâches.
12. Il va permettre à l'utilisateur d'importer les tâches de son Google calendrier. Les tâches importées peuvent être attribuées à un projet, et cela s'applique également aux séances. De plus, les tâches peuvent être créées avec les séances.
13. Le logiciel va permettre la gestion des séances en offrant les options suivantes:
    a. Consultation des séances.
    b. Ajout des séances.
    c. Modification des séances (modifier les informations telles que la date, la description, insérer des documents).
    d. Suppression des séances.

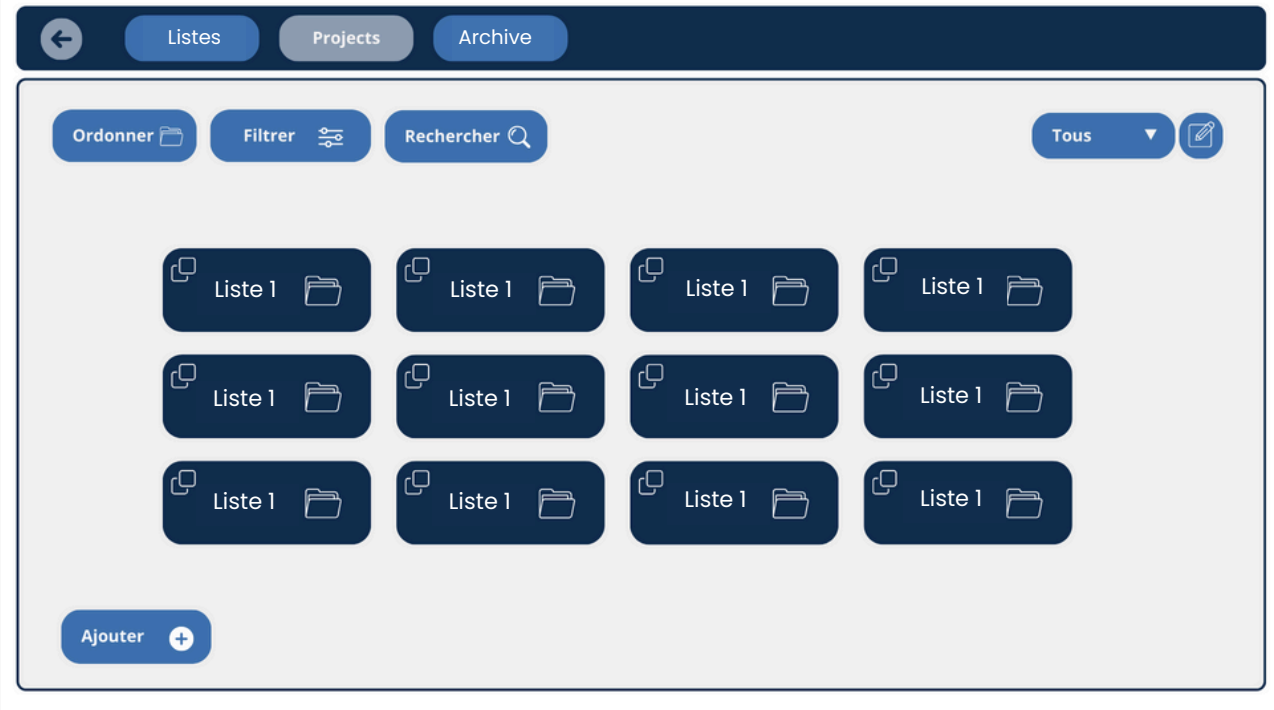
Les fonctionnalités de l'application

14. Le logiciel offre l'option de recherche par les mots-clés existants dans les documents attachés aux projets.
15. Le logiciel contient un tableau de bord affichant les différentes statistiques de l'utilisateur, telles que le nombre d'heures de travail dans un projet.
16. Le logiciel peut contenir des options avancées telles que l'insertion des notes vocales et les attacher à des séances, et en plus, l'option de recherche sur le contenu des documents.

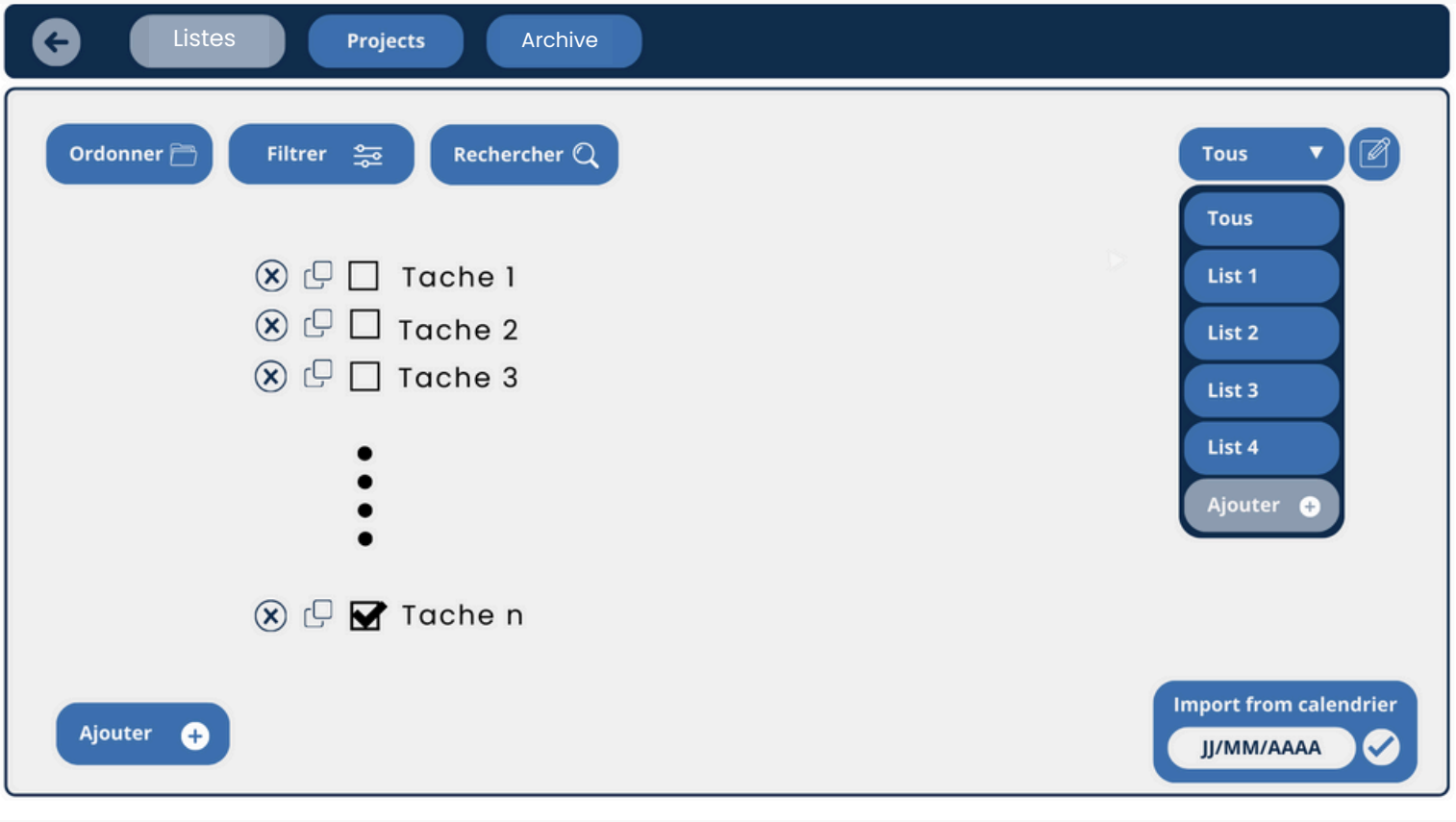
Les maquettes :

Page de connexion :

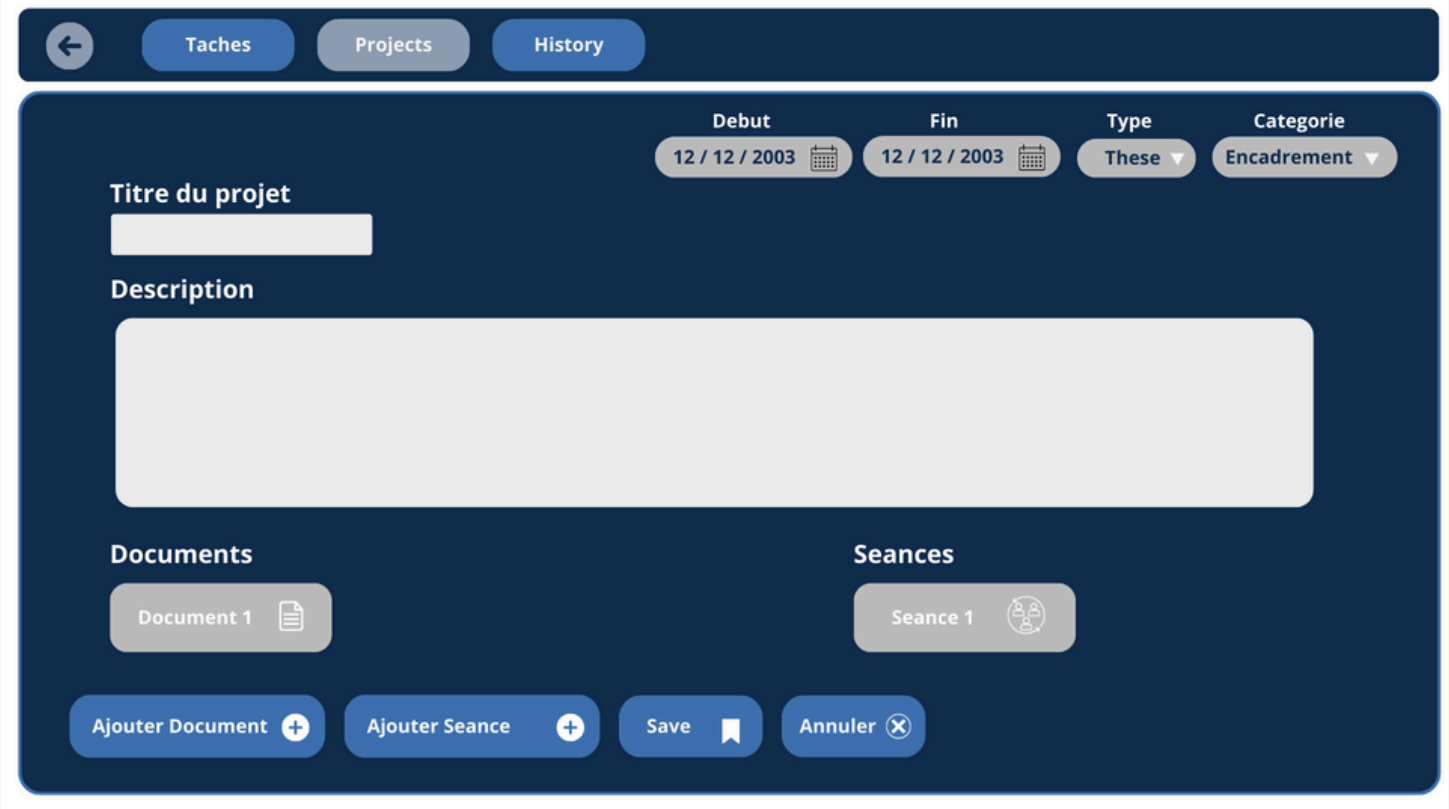
Page des listes des taches



Liste detail en cliquant sur l'une des listes



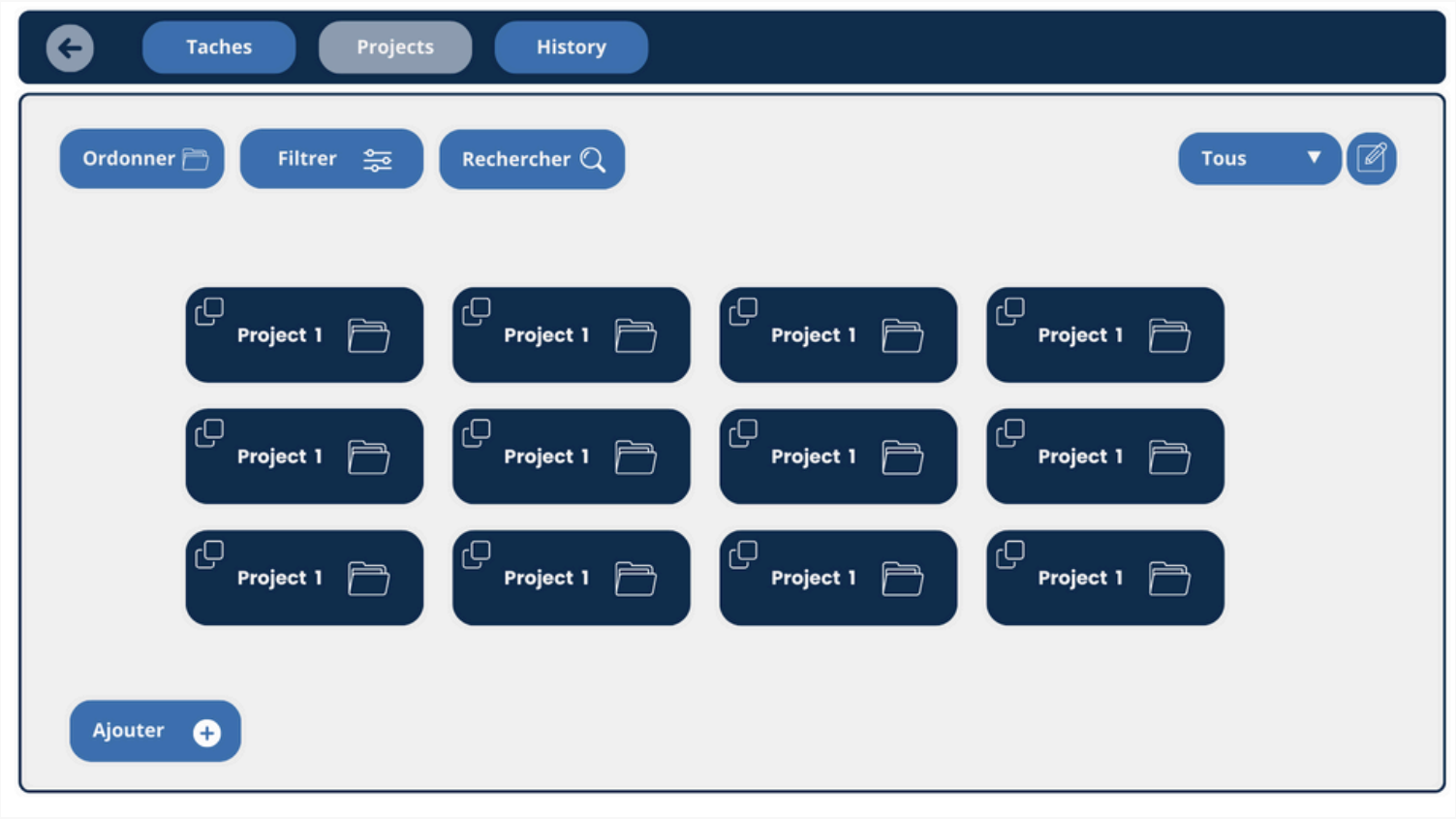
Formulaire de liste



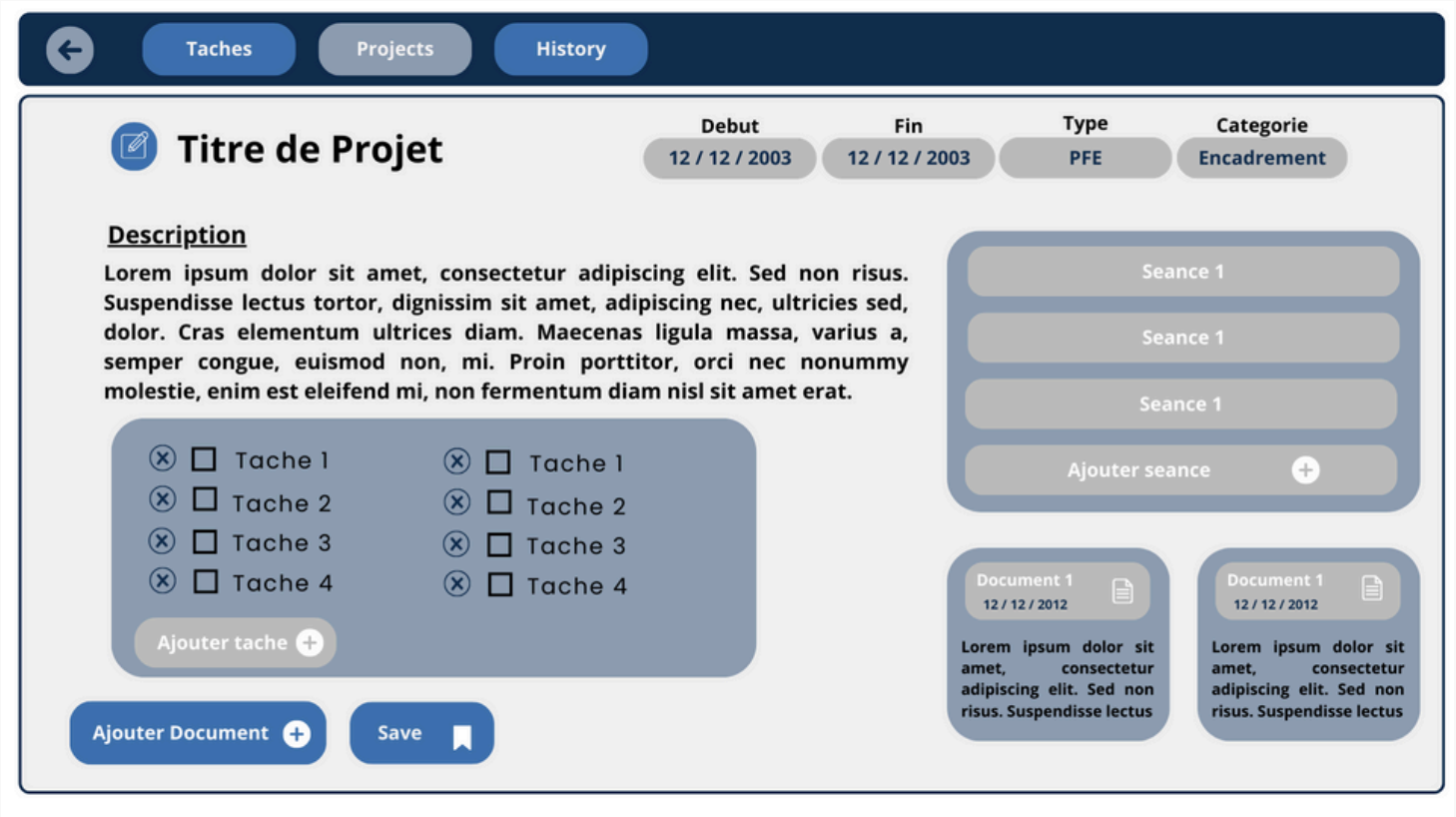
Formulaire de tache

Formulaire document

Page des projets

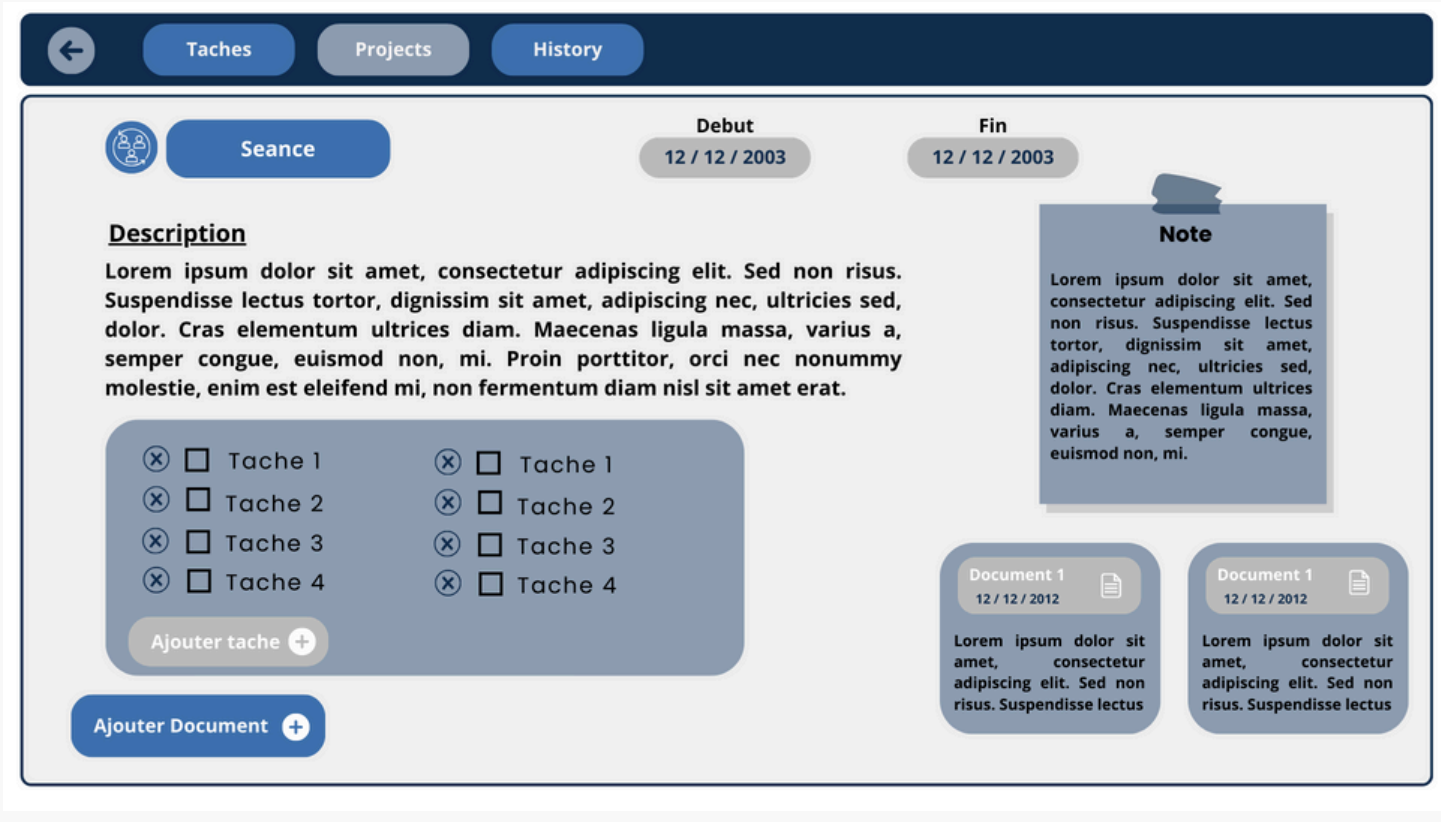


Projet detail

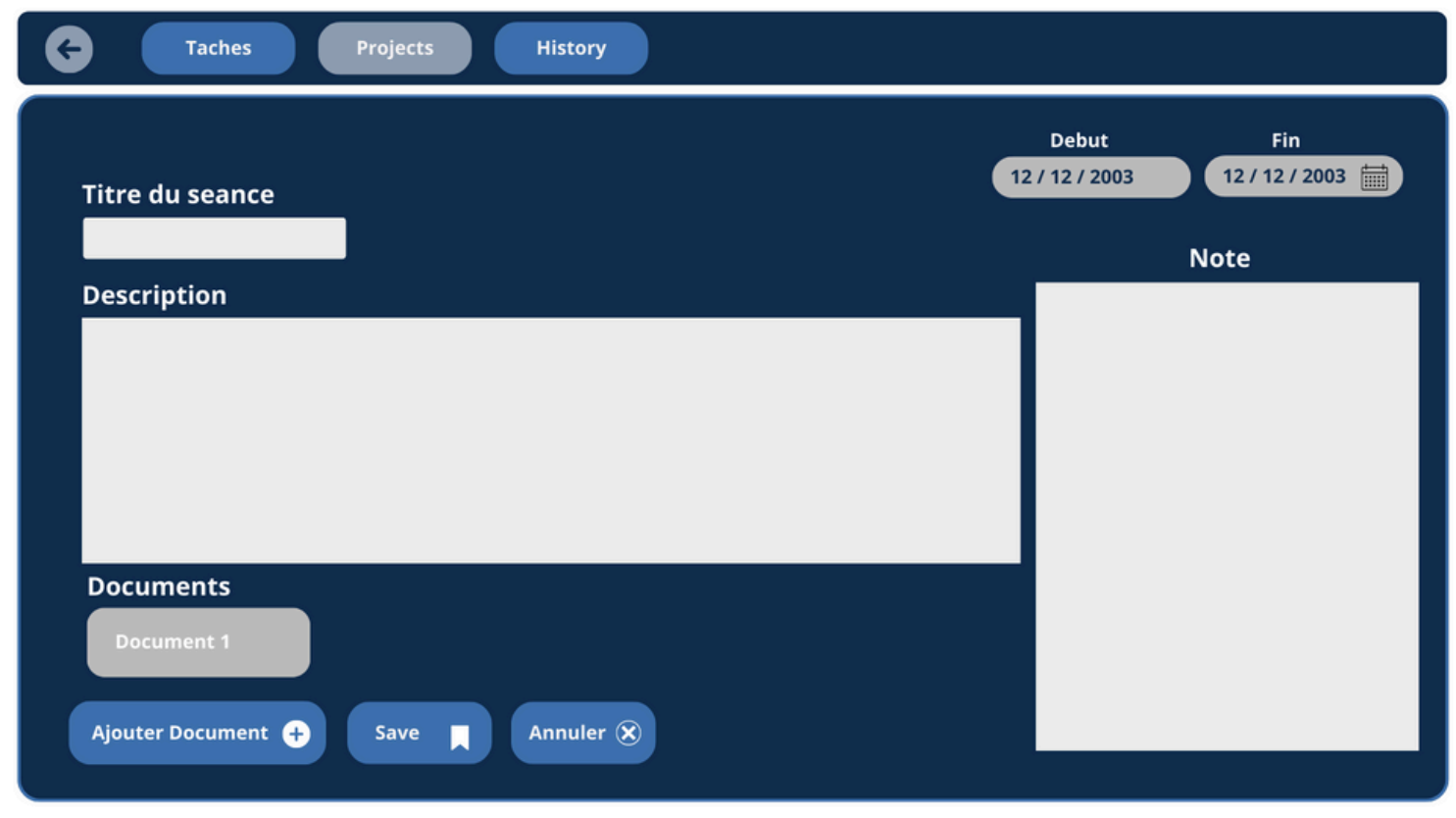


Tache details

Seance detail

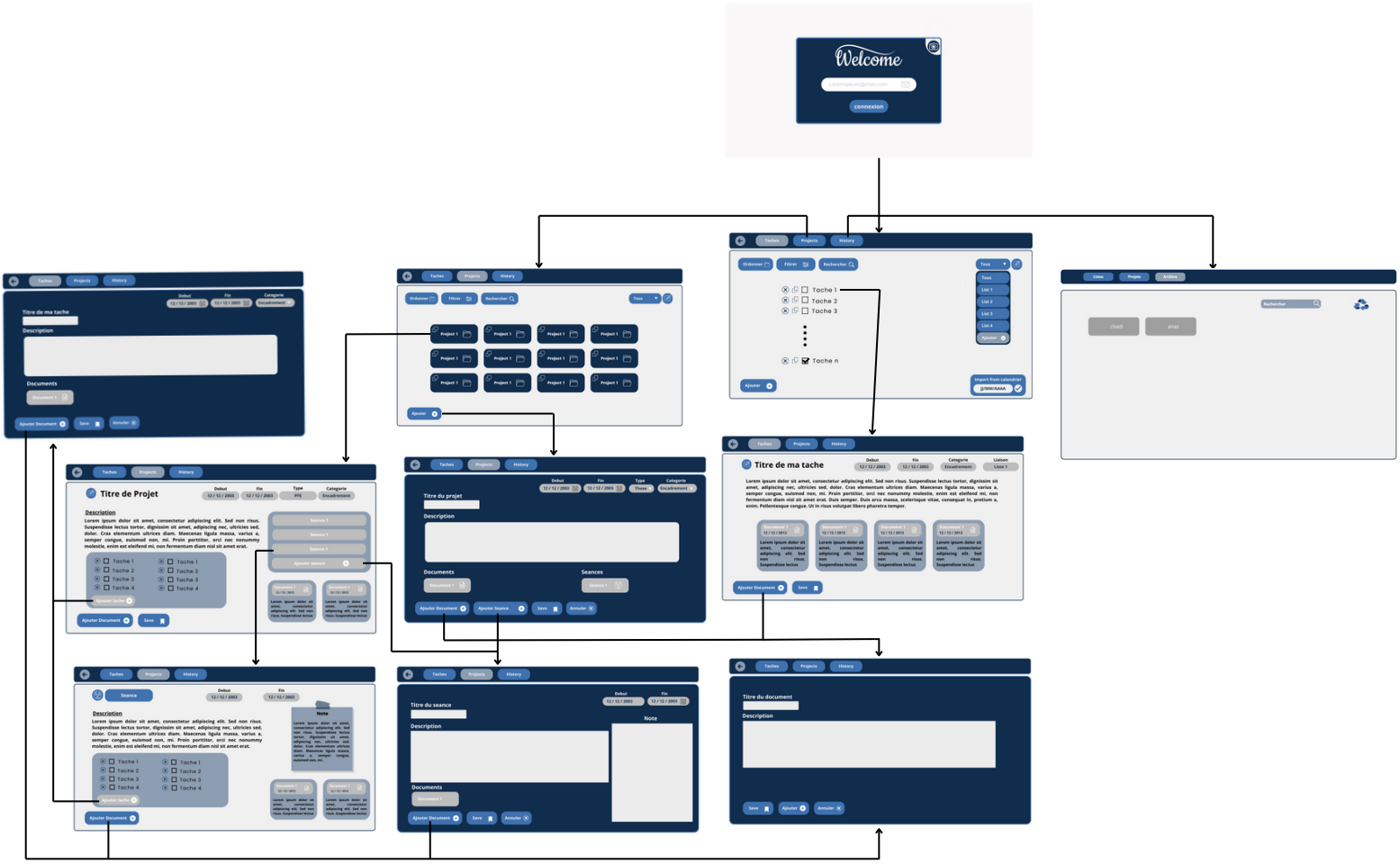


Formulaire Seance



Archive des projet

Diagramme de navigation



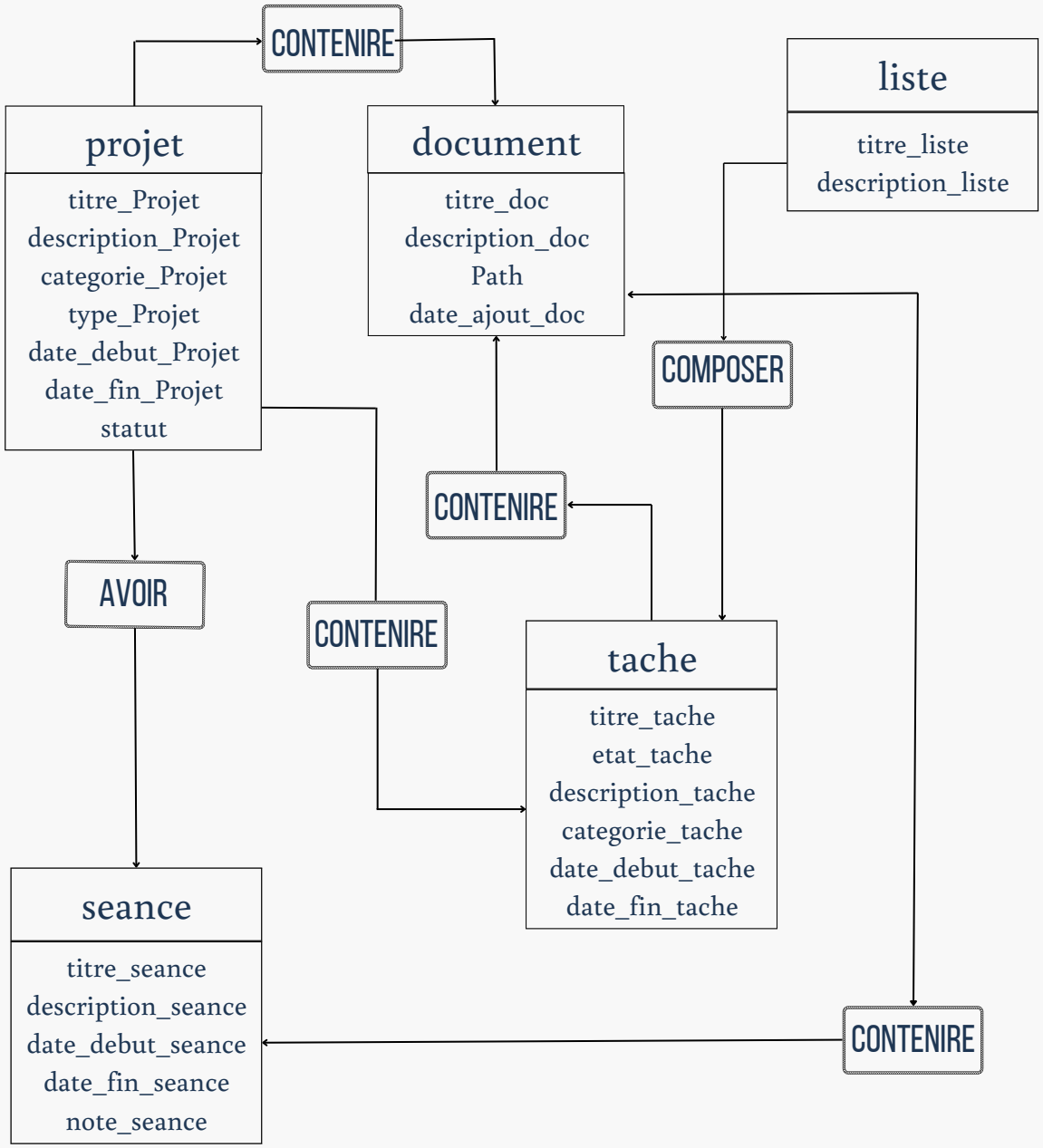
dictionnaire de données

Données Retenues	Type de Données	Commentaires
titre_doc	Alphanumérique	Texte
description_doc	Alphanumérique	Texte
Path	Alphanumérique	Chemin du fichier dans le device
date_ajout_doc	Date/Heure	
titre_Projet	Alphanumérique	Texte
description_Projet	Alphanumérique	Texte
categorie_Projet	Alphanumérique	Enseignement, Encadrement, Autre
type_Projet	Alphanumérique	Thèse, PFE, PFA, Cours, Examen, Autre
date_debut_Projet	Date/Heure	
date_fin_Projet	Date/Heure	
statut	Booléen	0 si le projet est clos et 1 si non
titre_tache	Alphanumérique	Texte
etat_tache	Booléen	0 si la tache est clos et 1 si non
description_tache	Alphanumérique	Texte
categorie_tache	Alphanumérique	Enseignement, Encadrement, Autre
date_debut_tache	Date/Heure	
date_fin_tache	Date/Heure	
titre_liste	Alphanumérique	Texte
description_liste	Alphanumérique	Texte

dictionnaire de données

titre_seance	Alphanumérique	Texte
description_seance	Alphanumérique	Texte
date_debut_seance	Date/Heure	
date_fin_seance	Date/Heure	
note_seance	Alphanumérique	Texte

MCD (modele conceptuel des donnees)



La conception générale

Nous avons réalisé une application en trois couches :

1. Couche persistance :

- Cette couche était responsable de la communication avec la base de données.
- Chaque objet dans la base de données était représenté par une classe DAO.
- Chaque table DAO offrait les services CRUD au minimum pour l'objet qu'elle représentait.
- Nous avons ajouté une classe responsable de la connexion, de la lecture et de l'écriture avec la base de données en utilisant le pattern singleton.
- Un objet était équivalent à une entité dans le modèle conceptuel de données.

2. Couche Métier :

- Cette couche était divisée en trois packages : POJO, Gestion et Service.
- POJO : Les classes métier étaient des objets BD avec des getters et des setters.
- Gestion : Les classes géraient les opérations CRUD et la logique métier des POJO (collections de POJO + méthodes CRUD).
- Service : Les classes géraient les appels aux API extérieures.
- Seules les classes Gestion communiquaient avec la couche persistance.

3. Couche Présentation :

- Chaque écran dans la maquette était représenté par une classe.
- Nous avons ajouté une classe modèle pour chaque écran (cette classe contenait les données de l'écran avec des getters et des setters).
- Une classe contrôleur gérait l'ensemble des écrans et actions d'un objet.
- Nous avons respecté le pattern MVC pour les classes précédentes.
- Seules les classes contrôleurs avaient le droit de communiquer avec la couche métier (classes Gestion), un contrôleur pouvait appeler un autre contrôleur.

Nous avons appliqué ce modèle pour réaliser notre application en suivant ces règles et en respectant le pattern MVC.

Documentation des API utilisées et leurs interactions avec le logiciel

Google Tasks

Notre application utilise l'API Google Tasks pour intégrer la gestion des tâches dans notre système. L'API Google Tasks permet aux utilisateurs d'afficher, de créer, de modifier et de supprimer des listes de tâches et des tâches individuelles. Cette intégration permet de synchroniser les tâches de l'application avec celles de Google Tasks, offrant une gestion unifiée et centralisée des tâches.

Bibliothèques et API utilisées:

- Google API Client Library for Java: Cette bibliothèque permet d'interagir avec les API de Google. Nous utilisons principalement les classes et méthodes fournies par cette bibliothèque pour authentifier les utilisateurs et accéder à leurs données Google Tasks.

Structure du package "mygroup.metier.Service":

Le package "mygroup.metier.Service" contient la classe `TaskQuickstart` qui gère l'interaction avec l'API Google Tasks. Voici une description détaillée des principales fonctionnalités :

1. Authentification et autorisation:
   - `getCredentials`:
     Cette méthode gère l'authentification de l'utilisateur et la récupération des informations d'identification nécessaires pour accéder à l'API Google Tasks. Elle utilise OAuth 2.0 pour autoriser l'application à accéder aux données de l'utilisateur.
   - Lit les secrets du client à partir du fichier `credentials.json`.
   - Initialise le flux de l'autorisation OAuth 2.0.
   - Utilise un serveur local pour gérer le processus d'autorisation.
   - Rafraîchit le token si nécessaire ou redirige l'utilisateur pour renouveler les autorisations.

- `hasPermissions`:
Vérifie si l'utilisateur a les permissions nécessaires pour accéder aux données Google Tasks. Si le token est expiré ou invalide, il tente de le rafraîchir.

2. Récupération des tâches:

- `getTasks`:
Cette méthode récupère les tâches de l'utilisateur à partir de Google Tasks pour une date spécifique. Elle :
- Vérifie les permissions.
- Récupère les listes de tâches de l'utilisateur.
- Pour chaque liste de tâches, récupère les tâches et les filtre par date due.
- Formate les informations des tâches et les ajoute à une liste observable qui peut être utilisée dans l'interface utilisateur.

3. Formatage et nettoyage des données:

- `getFormattedDateTime`:
Formate la date et l'heure des tâches récupérées pour les afficher de manière conviviale.

- `clearTokenContent`:
Supprime les fichiers de token pour forcer une nouvelle autorisation lors de la prochaine connexion. Cela peut être utile en cas de problème avec les permissions ou pour réinitialiser l'état de l'authentification.

Exemple d'interaction avec l'application:

1. Authentification:
Lorsqu'un utilisateur ouvre l'application pour la première fois ou lorsque le token est expiré, l'application le redirige vers un navigateur pour se connecter avec son compte Google et autoriser l'accès à Google Tasks.

2. Récupération des tâches:
Après authentification, l'application appelle `getTasks` pour récupérer les tâches du jour spécifié. Les tâches sont ensuite affichées dans l'interface utilisateur, permettant à l'utilisateur de voir ses tâches et leurs détails (titre, description, date due).

3. Mise à jour et affichage des tâches:
Les tâches sont affichées dans une interface JavaFX sous forme de liste observable. L'utilisateur peut interagir avec cette liste pour afficher les détails des tâches, marquer des tâches comme complétées ou en créer de nouvelles (fonctionnalités supplémentaires potentielles).

4. Gestion des erreurs:
En cas d'erreur d'authentification ou de problème de connexion à l'API, des messages d'erreur appropriés sont affichés, et des actions correctives comme le rafraîchissement du token ou la redirection pour une nouvelle autorisation sont entreprises.

Hunter

Notre application utilise l'API Hunter pour vérifier la validité des adresses e-mail. Hunter fournit un service de vérification d'e-mails qui permet de déterminer si une adresse e-mail est livrable, c'est-à-dire si elle est valide et peut recevoir des e-mails. Cette vérification est essentielle pour garantir la qualité des données des utilisateurs et éviter les erreurs lors de la communication par e-mail.

Bibliothèques et API utilisées:

-Apache HttpClient: Utilisé pour envoyer des requêtes HTTP à l'API Hunter et recevoir les réponses.
-JSON.simple: Utilisé pour analyser et traiter les réponses JSON de l'API Hunter.
-Java I/O: Utilisé pour lire les fichiers de configuration.

Structure du package `mygroup.metier.Service`:

Le package `mygroup.metier.Service` contient la classe `LoginService` qui gère l'interaction avec l'API Hunter pour vérifier les adresses e-mail. Voici une description détaillée des principales fonctionnalités :

1. Vérification de l'adresse e-mail

- `isValidEmailAddress`:
  Cette méthode vérifie si une adresse e-mail est valide en utilisant l'API Hunter. Elle :

  - Récupère la clé API à partir du fichier de configuration.
  - Exécute une requête de vérification d'adresse e-mail à l'API Hunter.
  - Analyse le résultat de la requête pour déterminer si l'adresse e-mail est livrable.

2. Récupération de la clé API

- `getAPIKeyFromConfig`:
  Cette méthode lit la clé API à partir d'un fichier de configuration JSON. Elle :

  - Lit le contenu du fichier JSON.
  - Analyse le JSON pour extraire la clé API.

3. Exécution de la requête de vérification

- `executeEmailVerificationRequest`:
  Cette méthode envoie une requête HTTP à l'API Hunter pour vérifier une adresse e-mail. Elle :

  - Crée une requête HTTP GET avec l'adresse e-mail et la clé API.
  - Envoie la requête et reçoit la réponse.
  - Convertit la réponse en une chaîne de caractères.

4. Analyse du résultat de la vérification

- `parseEmailVerificationResult`:
  Cette méthode analyse la réponse JSON de l'API Hunter pour déterminer si l'adresse e-mail est livrable. Elle :

  - Analyse le JSON de la réponse.
  - Extrait et évalue le résultat de la vérification.

5. Lecture du fichier JSON

- `getJSONFromFile`:
  Cette méthode lit le contenu d'un fichier JSON. Elle :

  - Ouvre et lit le fichier ligne par ligne.
  - Construit une chaîne de caractères contenant le contenu du fichier.

Exemple d'interaction avec l'application

1. Vérification de l'adresse e-mail:
   Lorsqu'un utilisateur saisit une adresse e-mail, l'application appelle la méthode `isValidEmailAddress` pour vérifier sa validité. Cette méthode :

   - Lit la clé API à partir du fichier de configuration en appelant `getAPIKeyFromConfig`.
   - Envoie une requête à l'API Hunter avec l'adresse e-mail et la clé API en appelant `executeEmailVerificationRequest`.
   - Analyse la réponse de l'API pour déterminer si l'adresse e-mail est livrable en appelant `parseEmailVerificationResult`.

2. Gestion des erreurs:
   Si une erreur survient lors de la lecture de la clé API ou de la requête à l'API Hunter, une exception est capturée et l'erreur est affichée dans la console. L'application retourne `false` pour indiquer que l'adresse e-mail n'est pas vérifiable.

3. Utilisation des fichiers de configuration:
La clé API est stockée dans un fichier JSON (`config.json`) qui est lu par la méthode `getAPIKeyFromConfig`. Cette méthode utilise `getJSONFromFile` pour lire le fichier et extraire la clé API.

Conclusion

L'intégration de l'API Hunter dans notre application permet de vérifier efficacement la validité des adresses e-mail, améliorant ainsi la qualité des données utilisateur et réduisant les erreurs de communication. Grâce à l'utilisation des bibliothèques Apache HttpClient et JSON.simple, nous avons pu implémenter une solution robuste et fiable pour interagir avec l'API Hunter.

Google Calendar

Dans notre application, nous utilisons l'API Google Calendar pour récupérer les événements du calendrier de l'utilisateur. Cette intégration permet d'afficher les événements planifiés dans l'application, offrant ainsi une vue consolidée de l'emploi du temps de l'utilisateur.

Bibliothèques et API utilisées

- Google API Client Library for Java: Cette bibliothèque permet d'interagir avec les API de Google. Nous utilisons principalement les classes et méthodes fournies par cette bibliothèque pour authentifier les utilisateurs et accéder à leurs données de calendrier.

Structure du package `mygroup.metier.Service`

Le package `mygroup.metier.Service` contient la classe `CalendarQuickstart` qui gère l'interaction avec l'API Google Calendar pour récupérer les événements du calendrier de l'utilisateur. Voici une description détaillée des principales fonctionnalités :

1. Authentification et autorisation

- `getCredentials`:
Cette méthode gère l'authentification de l'utilisateur et la récupération des

informations d'identification nécessaires pour accéder à l'API Google Calendar. Elle utilise OAuth 2.0 pour autoriser l'application à accéder aux données de calendrier de l'utilisateur.

- Lit les secrets du client à partir du fichier `credentials.json`.
- Initialise le flux de l'autorisation OAuth 2.0.
- Utilise un serveur local pour gérer le processus d'autorisation.
- Rafraîchit le token si nécessaire ou redirige l'utilisateur pour renouveler les autorisations.

- `hasPermissions`:
Vérifie si l'utilisateur a les permissions nécessaires pour accéder aux données du calendrier. Si le token est expiré ou invalide, il tente de le rafraîchir.

2. Récupération des événements du calendrier

- `getEvents`:
Cette méthode récupère les événements du calendrier de l'utilisateur à partir de Google Calendar. Elle :

- Vérifie les permissions.
- Récupère les événements du calendrier de l'utilisateur pour la journée spécifiée.
- Formate les informations des événements et les ajoute à une liste observable qui peut être utilisée dans l'interface utilisateur.

3. Formatage et nettoyage des données

- `getFormattedDateTime`:
Formate la date et l'heure des événements récupérés pour les afficher de manière conviviale.

- `clearTokenContent`:
Supprime les fichiers de token pour forcer une nouvelle autorisation lors de la prochaine connexion. Cela peut être utile en cas de problème avec les permissions ou pour réinitialiser l'état de l'authentification.

Exemple d'interaction avec l'application

1. Authentification:
   Lorsqu'un utilisateur ouvre l'application pour la première fois ou lorsque le token est expiré, l'application le redirige vers un navigateur pour se connecter avec son compte Google et autoriser l'accès à Google Calendar.

2. Récupération des événements du calendrier:
   Après authentification, l'application appelle `getEvents` pour récupérer les événements du calendrier de l'utilisateur pour la journée spécifiée. Les événements sont ensuite affichés dans l'interface utilisateur, permettant à l'utilisateur de voir ses rendez-vous et activités planifiées.

3. Gestion des erreurs:
   En cas d'erreur d'authentification ou de problème de connexion à l'API, des messages d'erreur appropriés sont affichés, et des actions correctives comme le rafraîchissement du token ou la redirection pour une nouvelle autorisation sont entreprises.

Plan de Tests Fonctionnels

1. Test de Connexion par Email : Vérifier que l'utilisateur peut se connecter à l'application en utilisant son email.
2. Test des Boutons de la Barre de Navigation : S'assurer que les boutons de la barre de navigation (Listes, Projets, Archive) fonctionnent correctement.
3. Test du Bouton "Ordonner" dans l'Interface des Listes : Confirmer que le bouton "Ordonner" permet de trier les listes comme prévu.
4. Test de la Fonction de Recherche par Mot-Clé : Vérifier que la recherche par mot-clé retourne les résultats appropriés.
5. Test d'Ajout d'une Nouvelle Tâche : Valider que l'utilisateur peut ajouter une nouvelle tâche correctement.
6. Test d'Importation des Tâches depuis Google Calendrier : S'assurer que les tâches peuvent être importées depuis Google Calendrier sans problème.
7. Test de Sélection d'une Liste : Vérifier que l'utilisateur peut sélectionner et afficher le contenu d'une liste.
8. Test de Création d'une Liste : Confirmer que l'utilisateur peut créer une nouvelle liste.
9. Test de Détails d'une Tâche : S'assurer que les détails d'une tâche s'affichent correctement.
10. Test d'Ajout d'un Document à une Tâche : Valider que l'utilisateur peut ajouter des documents aux tâches.
11. Test d'Importation d'une Tâche dans l'Interface Ajouter Tâche : Vérifier que l'importation d'une tâche dans l'interface fonctionne correctement.
12. Test de Sauvegarde des Modifications d'une Tâche : S'assurer que les modifications apportées à une tâche sont bien sauvegardées.
13. Test de Création d'un Projet : Confirmer que l'utilisateur peut créer un nouveau projet.
14. Test d'Ajout d'un Document à un Projet : Vérifier que des documents peuvent être ajoutés à un projet.
15. Test d'Ajout d'une Séance à un Projet : Valider que l'utilisateur peut ajouter des séances à un projet.
16. Test de Sauvegarde d'un Projet : S'assurer que les projets peuvent être sauvegardés correctement.

17. Test du Bouton "Annuler" : Vérifier que le bouton "Annuler" fonctionne comme prévu.
18. Test d'Affichage des Éléments Attachés à un Projet : Confirmer que les séances, tâches et documents attachés à un projet s'affichent correctement.
19. Test d'Ajout d'une Séance : Valider que l'utilisateur peut ajouter une nouvelle séance.
20. Test d'Affichage d'une Séance : Vérifier que les détails d'une séance s'affichent correctement.
21. Test d'Ajout des Projets Clôturés aux Archives : S'assurer que les projets clôturés sont bien ajoutés à l'archive.
22. Test de Recherche dans l'Archive : Valider que la recherche dans l'archive fonctionne correctement et retourne les résultats attendus.
23. Test de Suppression d'une Tâche : Vérifier que l'utilisateur peut supprimer une tâche et que celle-ci disparaît de l'interface appropriée sans erreurs.
24. Test de Modification des Informations d'un Projet : S'assurer que les informations d'un projet peuvent être modifiées et que les modifications sont sauvegardées correctement.
25. Test de Clonage d'unE tache : Valider que l'utilisateur peut cloner un projet et que toutes les informations du projet original sont correctement dupliquées.
26. Test de Clôture d'un Projet : Vérifier que l'utilisateur peut clôturer un projet et que le projet passe bien dans l'état "clôturé".
27. Test de Filtrage des Projets par Catégorie ou Type : S'assurer que les projets peuvent être filtrés par catégorie (Enseignement, Encadrement, Autre) ou type (Thèse, PFE, etc.) et que les résultats sont corrects.
28. Test de Recherche des Projets par Mot-Clé : Vérifier que la recherche de projets par mot-clé fonctionne correctement et retourne les résultats pertinents.
29. Test de Consultation des Statistiques : Valider que les statistiques, telles que le nombre d'heures de travail et le nombre de documents par projet, s'affichent correctement.
30. Test de Performance : Vérifier que l'application fonctionne de manière fluide et efficace, même avec un grand nombre de projets, tâches et documents.

32. Test d'Importation et d'Exportation de Données : Vérifier que les données peuvent être importées et exportées sans perte ni corruption, facilitant ainsi les sauvegardes et les restaurations.
33. Test de Gestion des Séances Importées depuis Google Calendrier : Valider que les séances importées depuis Google Calendrier s'affichent correctement et peuvent être éditées ou associées à des projets existants.
Ces tests permettent de garantir que l'application est fonctionnelle, robuste, et répond aux besoins de l'utilisateur tout en offrant une expérience utilisateur optimale.

Maintenance

Les Fonctionnalités à Ajouter

1. Gestion de la Participation des Étudiants : La gestion de la participation des étudiants dans les séances pourrait être une fonctionnalité précieuse. En stockant les notes de participation des étudiants pour chaque séance, les professeurs pourraient suivre de manière plus précise et organisée l'engagement de leurs étudiants. Cette modification est relativement facile à implémenter puisqu'il s'agit principalement d'ajouter une interface utilisateur permettant d'ajouter et de consulter les notes de participation. En intégrant cette fonctionnalité, l'application offrirait une vue d'ensemble plus complète de chaque séance, facilitant ainsi l'évaluation continue des étudiants.

2. Notifications et Alertes : Ajouter des notifications et des alertes pour rappeler les échéances et les tâches importantes est une autre fonctionnalité clé à intégrer. Cela peut être réalisé en utilisant des services de notification comme Firebase Cloud Messaging pour envoyer des notifications push. Cette tâche est relativement facile à accomplir car les services de notification sont bien documentés et leur intégration est généralement simple. Cependant, il sera crucial de gérer minutieusement les préférences des utilisateurs pour éviter les notifications excessives et garantir que les rappels restent utiles et non intrusifs.

3. Développement d'une Version Mobile : Développer une version mobile de l'application permettrait aux professeurs de gérer leurs tâches et projets en déplacement, augmentant ainsi leur flexibilité et leur efficacité. Cela pourrait être réalisé en créant des applications natives pour iOS et Android, ou en utilisant des frameworks multiplateformes comme React Native ou Flutter. Cette tâche est plus difficile, car le développement d'applications mobiles nécessite des compétences spécifiques, notamment en matière de conception d'interfaces utilisateur adaptées aux petits écrans et de gestion de la synchronisation des données entre les appareils.

Conclusion

Le développement de cette application de gestion des tâches et des projets académiques a été une expérience enrichissante et formatrice.
Le projet a été conçu spécifiquement pour répondre aux besoins de notre professeur de classe, offrant une solution pratique pour organiser et suivre ses diverses activités académiques.

Avis sur le Projet

Le projet a été particulièrement intéressant à réaliser, car il s'agissait de créer un outil personnalisé et adapté aux exigences spécifiques de notre professeur.
L'application offre une interface conviviale et des fonctionnalités robustes pour la gestion des projets et des tâches, permettant ainsi une organisation efficace du travail.
Le choix des technologies, comme Java avec Swing ou JavaFX pour l'interface graphique, et JSON pour le stockage des données, a permis de développer une application fiable et performante.
Le développement a été axé sur la simplicité d'utilisation et la clarté de présentation, assurant ainsi que notre professeur puisse utiliser l'application sans difficulté.
La documentation et les commentaires dans le code source ont été soigneusement préparés pour faciliter la maintenance future et les éventuelles modifications.

Résultats Produits

Les résultats obtenus sont conformes aux attentes initiales. L'application répond pleinement aux besoins de gestion des projets et des tâches académiques, en offrant des fonctionnalités comme la création, la modification, la clôture, et la consultation de projets et de tâches. L'intégration avec Google Calendrier pour l'importation des séances de travail a ajouté une dimension pratique, permettant une synchronisation facile avec les événements existants.
Les tests effectués ont montré que l'application fonctionne de manière stable et fiable, offrant une performance fluide même avec un volume important de données. Les retours de notre professeur ont été très positifs, soulignant l'utilité et la facilité d'utilisation de l'outil.

Perspectives Éventuelles

Pour l'avenir, plusieurs perspectives d'amélioration peuvent être envisagées :

1. Amélioration de l'Interface Utilisateur : Bien que l'interface soit déjà conviviale, des améliorations peuvent toujours être apportées pour rendre l'expérience utilisateur encore plus agréable, comme l'ajout de thèmes personnalisables ou l'optimisation de la disposition des éléments.

2. Fonctionnalités Avancées : Intégrer des fonctionnalités supplémentaires telles que des rappels automatiques pour les échéances des tâches, ou des statistiques plus détaillées sur le temps de travail et l'avancement des projets.

3. Optimisation des Performances : Continuer à optimiser l'application pour assurer une performance optimale, même avec une augmentation du nombre de projets et de tâches.

4. Maintenance et Mise à Jour : Assurer une maintenance régulière de l'application pour corriger d'éventuels bugs et ajouter de nouvelles fonctionnalités en fonction des retours de notre professeur.

En conclusion, ce projet a permis de créer une application utile et efficace pour la gestion des tâches académiques de notre professeur. L'expérience acquise lors de ce développement est précieuse, et les perspectives d'amélioration offrent de nombreuses opportunités pour continuer à perfectionner l'outil. Nous sommes satisfaits des résultats obtenus et confiants que cette application aidera notre professeur à mieux organiser et gérer ses activités académiques.