

Epreuve 5 : Projet n°2

Occicards



Mittou Dorian - BTS SIO SLAM - Lycée Paul Sabatier

Épreuve E5 - Conception et développement d'applications (option SLAM)

ANNEXE 7-1-B : Fiche descriptive de réalisation professionnelle (recto)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE		N° réalisation : 2
Nom, prénom : Mittou, Dorian	N° candidat : 02150196575	
Épreuve ponctuelle <input type="checkbox"/> Contrôle en cours de formation <input checked="" type="checkbox"/>	Date : 08/04/2025	
<p>Organisation support de la réalisation professionnelle Lycée Paul Sabatier</p> <p>Intitulé de la réalisation professionnelle Création d'un client lourd de gestion de flashcards</p> <p>Période de réalisation : 09/03/2025 au 18/04/2025 Lieu : Carcassonne</p> <p>Modalité : <input checked="" type="checkbox"/> Seul(e) <input type="checkbox"/> En équipe</p> <p>Compétences travaillées</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Concevoir et développer une solution applicative <input checked="" type="checkbox"/> Assurer la maintenance corrective ou évolutive d'une solution applicative <input checked="" type="checkbox"/> Gérer les données <p>Conditions de réalisation¹ (ressources fournies, résultats attendus) Ressources fournies : Occicards est une solution applicative multiplateforme (Windows, MacOs X, Linux) qui permet la gestion de flashcards.</p> <p>Résultats attendus : l'utilisateur doit pouvoir appliquer les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer) sur les decks, ainsi qu'importer et exporter ces derniers. De plus, il doit être capable d'effectuer les mêmes opérations CRUD sur les cartes individuelles au sein des decks. Enfin, l'utilisateur doit pouvoir visualiser ses decks afin de réviser les cartes qu'ils contiennent.</p> <p>Description des ressources documentaires, matérielles et logicielles utilisées² Ressources documentaires : Le Chat Mistral, documentation de Java, documentation de Java FX, Stack Overflow, vidéo YouTube. Ressources matérielles : ordinateur portable, clé USB. Ressources logicielles :</p> <ul style="list-style-type: none"> • Logiciels: Intelij IDEA édition communautaire • Langages : Java 21 • Gestion des dépendances : Maven • Gestion de version : git • Langage de données : JSON, FXML • Design pattern : Modèle, Vue, Controller • Librairies : JavaFX, org.json, Junit 5 <p>Modalités d'accès aux productions³ et à leur documentation⁴ Portfolio (rubrique projet) : https://dorian-mittou.website</p>		

¹ En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO.

² Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

³ Conformément au référentiel du BTS SIO « *Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l'épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d'organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l'épreuve.* ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d'un espace de stockage et de la présentation de l'organisation du stockage.

⁴ Lien vers la documentation complète, précisant et décrivant, si cela n'a été fait au verso de la fiche, la réalisation professionnelle, par exemples service fourni par la réalisation, interfaces utilisateurs, description des classes ou de la base de données.

Épreuve E5 - Conception et développement d'applications (option SLAM)

ANNEXE 7-1-B : Fiche descriptive de réalisation professionnelle
(verso, éventuellement pages suivantes)**Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs**Contexte

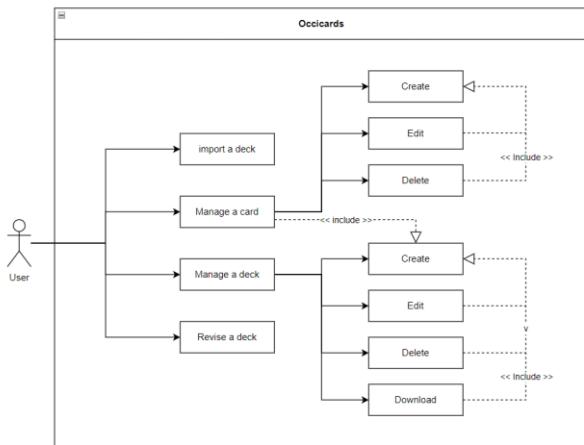
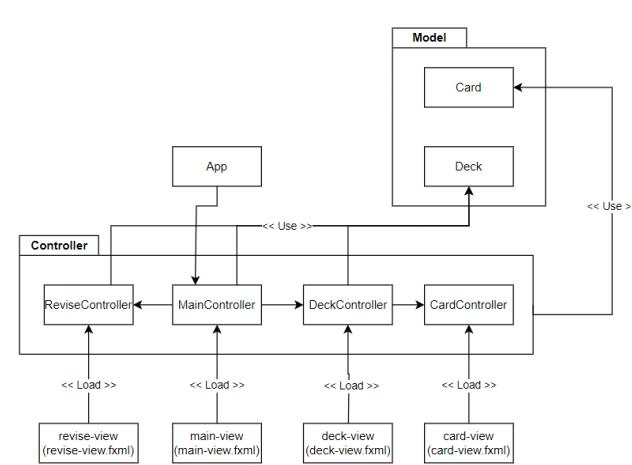
Développement d'une application de bureau de flashcards pour Sabatech, une agence digitale spécialisée dans les solutions numériques sur mesure, basée en région Occitanie.

Besoins

- Problématiques:** Les méthodes d'apprentissage traditionnelles (fiches de révision, bachotage, relecture passive) sont inefficaces pour la rétention à long terme.
- Science de l'apprentissage:** Utilisation de la répétition espacée et du Système Leitner pour améliorer la mémorisation et la rétention des informations.

Solutions

Développement d'OcciCards, une application de bureau en Java utilisant JavaFX, permettant un apprentissage interactif et personnalisé basé sur des flashcards multiplateformes

Diagramme de cas d'utilisationDiagramme de classe

Sommaire

Introduction.....	5
A) Présentation de l'organisation.....	5
B) Problématiques.....	5
B1) Problématiques.....	5
B2) Science de l'apprentissage.....	5
C) Les solutions proposés.....	6
1) Analyse conceptuelle.....	7
A) UML.....	7
A1) Diagramme de cas d'utilisation.....	7
I - Version web.....	7
II - Client lourd.....	8
A2) Diagramme de classe.....	9
I - Package dm.occicards.....	9
II - Package dm.occicards.model.....	9
III - Package dm.occicards.utils.....	10
IV - Package dm.occicards.controller.....	10
V- Liens.....	11
2) Analyse technique.....	12
A) Généralités.....	12
A1) Présentation de l'environnement technique.....	12
A2) Structure du projet.....	12
A3) Modèle Vue Contrôleur.....	13
B) Application.....	14
B1) Vue principale.....	14
I - Menus.....	14
II - Collection des decks.....	15
B2) Vue du deck.....	16
B3) Vue de carte.....	17
B4) Vue de révision.....	18
C) Tests unitaires.....	19
C1) Utilité.....	19
C2) Exemple de test unitaire réalisé sur Card.java.....	19
C3) Résultats.....	20
D) Documentation technique.....	20
Conclusion.....	20

Introduction

A) Présentation de l'organisation

Sabatech est une agence digitale spécialisée dans le développement de solutions numériques sur mesure, basée en région Occitanie. Forte de plusieurs années d'expérience dans le développement d'applications web et logicielles, elle accompagne les entreprises et les institutions publiques dans leur transformation numérique. Sabatech accorde une attention particulière à la sécurité, à la souveraineté des données et à l'utilisation de technologies open source afin d'assurer la pérennité et la conformité de ses solutions.

B) Problématiques

B1) Problématiques

Les méthodes d'apprentissage traditionnelles, telles que les fiches de révision, le bachotage et la relecture passive des notes, présentent plusieurs inconvénients :

- **Fiches de révision** : Elles sont souvent statiques et ne s'adaptent pas au niveau de maîtrise de l'apprenant, ce qui peut entraîner une révision inefficace.
- **Bachotage** : Cette méthode consiste à réviser intensivement juste avant un examen, ce qui ne favorise pas la rétention à long terme et peut entraîner une surcharge cognitive.
- **Relecture passive** : Relire simplement les notes ou le matériel d'apprentissage sans interaction active ne stimule pas suffisamment la mémoire.

B2) Science de l'apprentissage

La **répétition espacée** est basée sur l'effet d'espacement décrit par Hermann Ebbinghaus. Cette technique consiste à revoir le matériel d'apprentissage à des intervalles de temps croissants. Ebbinghaus a découvert que cette méthode réduit significativement la courbe de l'oubli, améliorant ainsi la rétention à long terme. Courbe de l'oubli Des études récentes continuent de soutenir cette théorie, montrant que la répétition espacée optimise l'encodage des informations dans la mémoire à long terme (Cepeda et al., 2006).

Le **Système Leitner**, conçu par Sebastian Leitner, est une méthode de révision utilisant des cartes flash. Les cartes sont organisées dans plusieurs boîtes en fonction de la maîtrise de

l'apprenant sur chaque sujet. Les cartes mal mémorisées sont revues plus fréquemment, tandis que celles mieux maîtrisées progressent dans des boîtes de révision moins fréquentes. Cette méthode assure un focus sur les zones de faiblesse, améliorant ainsi l'efficacité de l'apprentissage. Des recherches ont démontré que le Système Leitner renforce la mémoire active et la rétention à long terme. Il est particulièrement efficace pour les apprentissages nécessitant une mémorisation détaillée, comme les langues étrangères ou les terminologies spécifiques (Kornell, 2009).

C) Les solutions proposés

Sabatech répond à l'appel d'offres de la région Occitanie en proposant une application de bureau. Celle-ci permet aux utilisateurs d'apprendre de manière ludique et personnalisée, grâce aux principes de la répétition espacée et de la gamification, en s'appuyant sur le Système Leitner. Développée en Java, l'application peut être compilée sur les trois principaux systèmes d'exploitation pour ordinateur : Windows, macOS et Linux.

Pour répondre à ces enjeux, Sabatech propose **OcciCards**, une plateforme d'apprentissage basée sur un système de flashcards interactives, déclinée en deux versions :

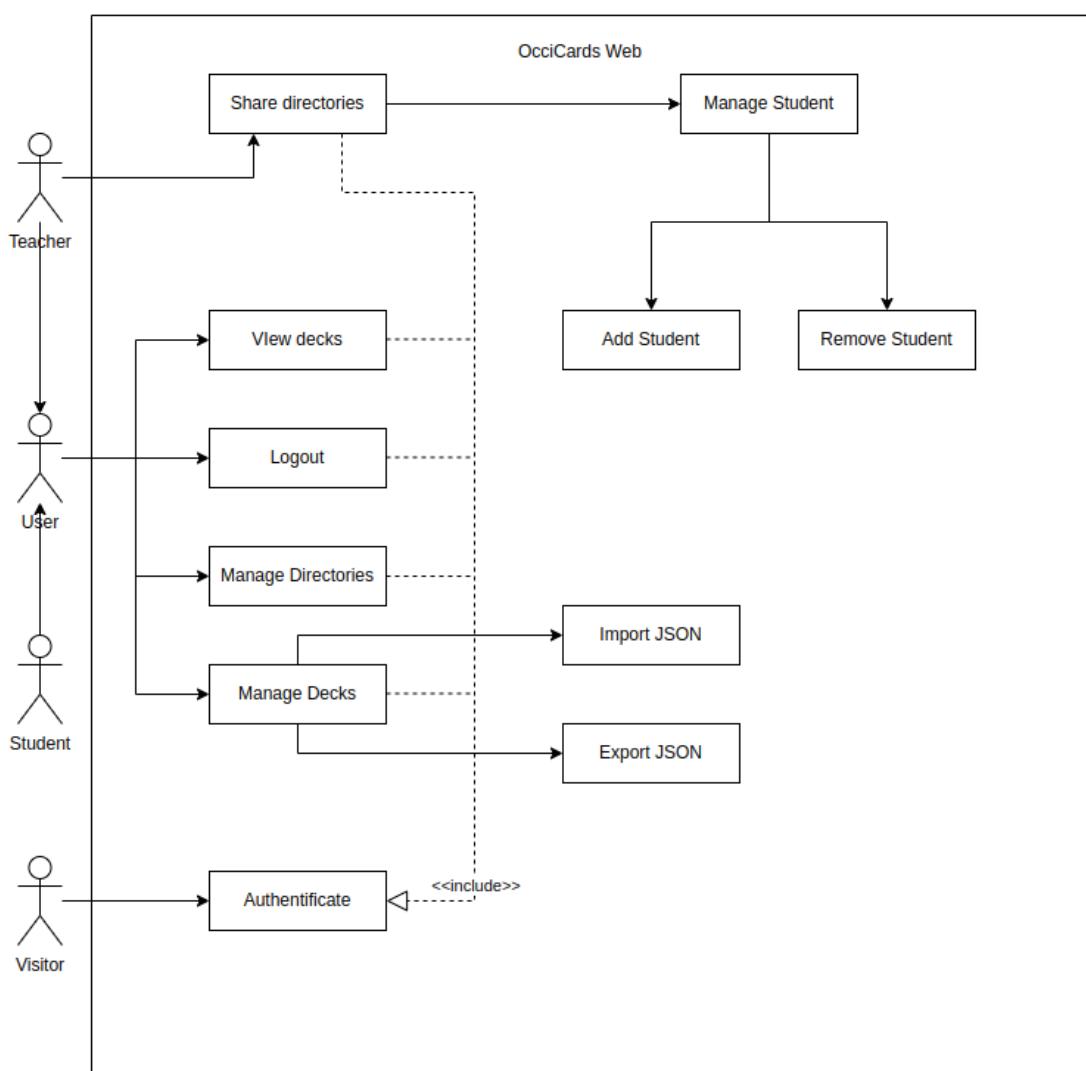
1. **Une application lourde** développée en **Java** pour une utilisation hors connexion directement sur un ordinateur.
2. **Une application légère en PHP et JavaScript** accessible via un navigateur web pour un accès rapide et flexible aux flashcards depuis n'importe quel appareil connecté.
(non réalisée)

1) Analyse conceptuelle

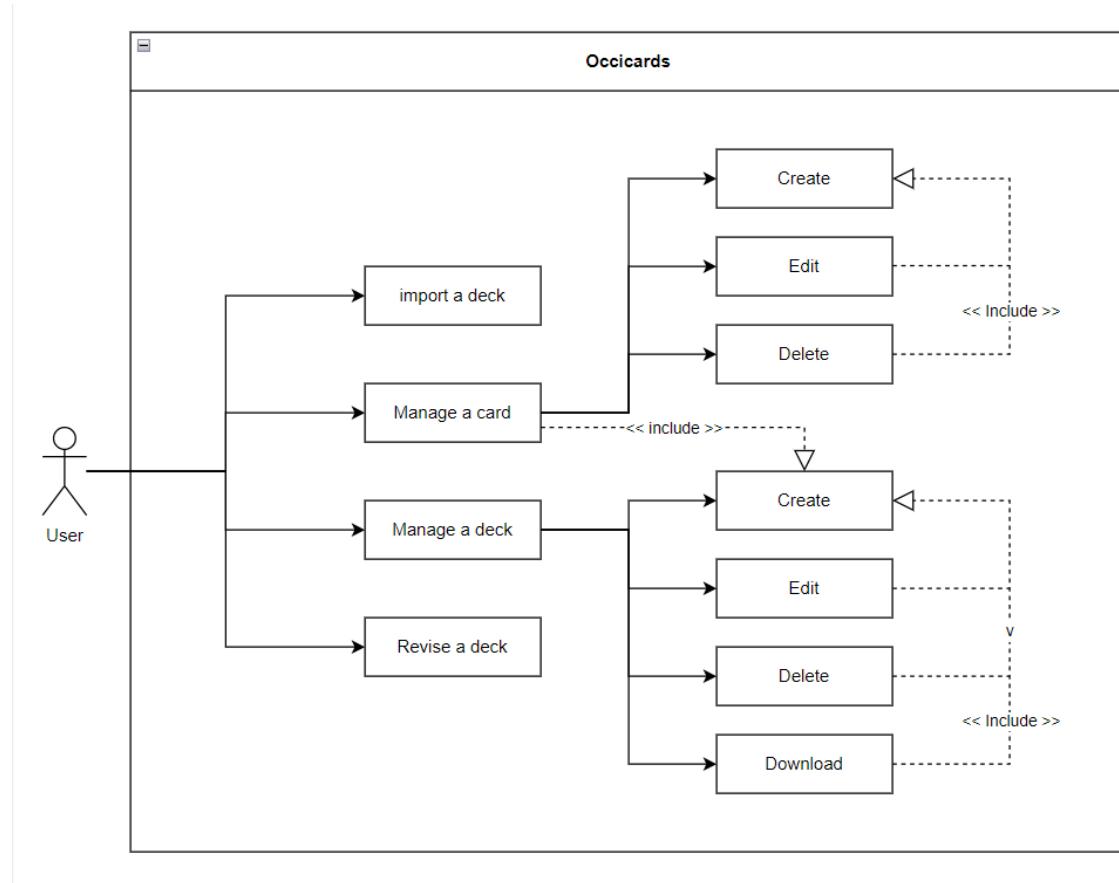
A) UML

A1) Diagramme de cas d'utilisation

I - Version web

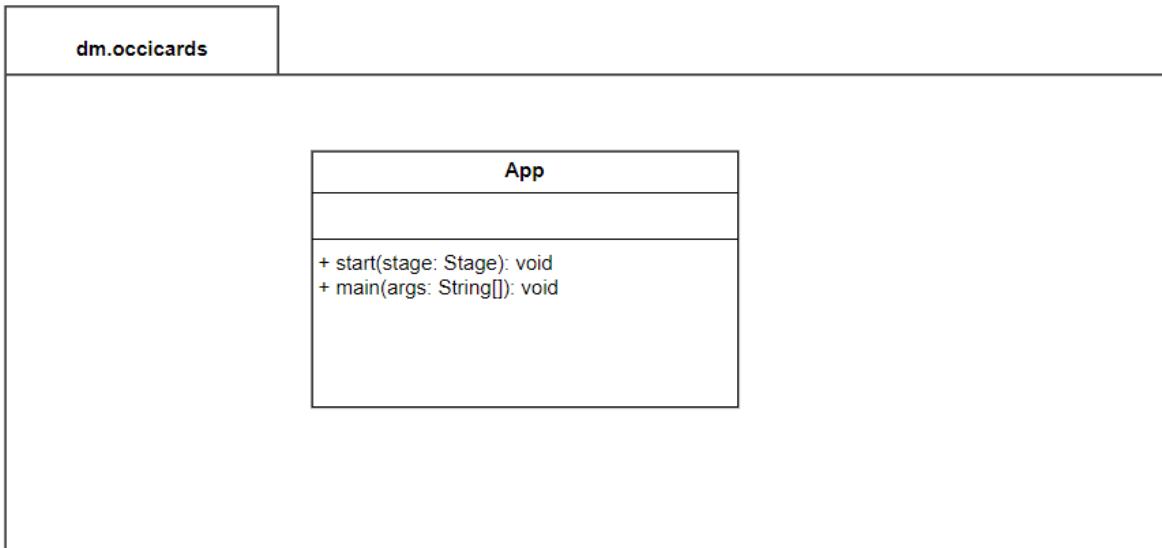


II - Client lourd

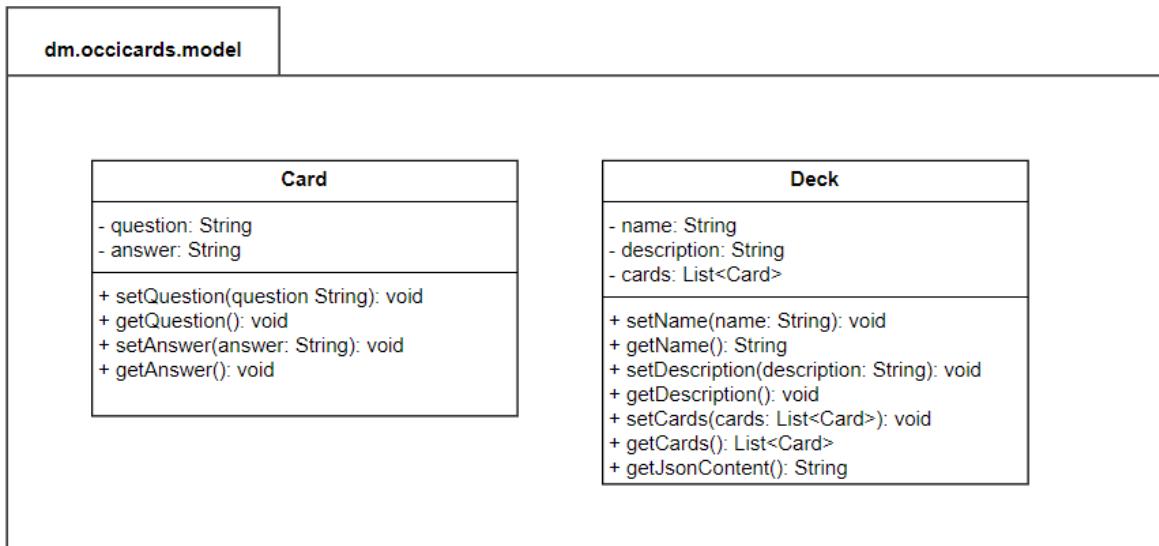


A2) Diagramme de classe

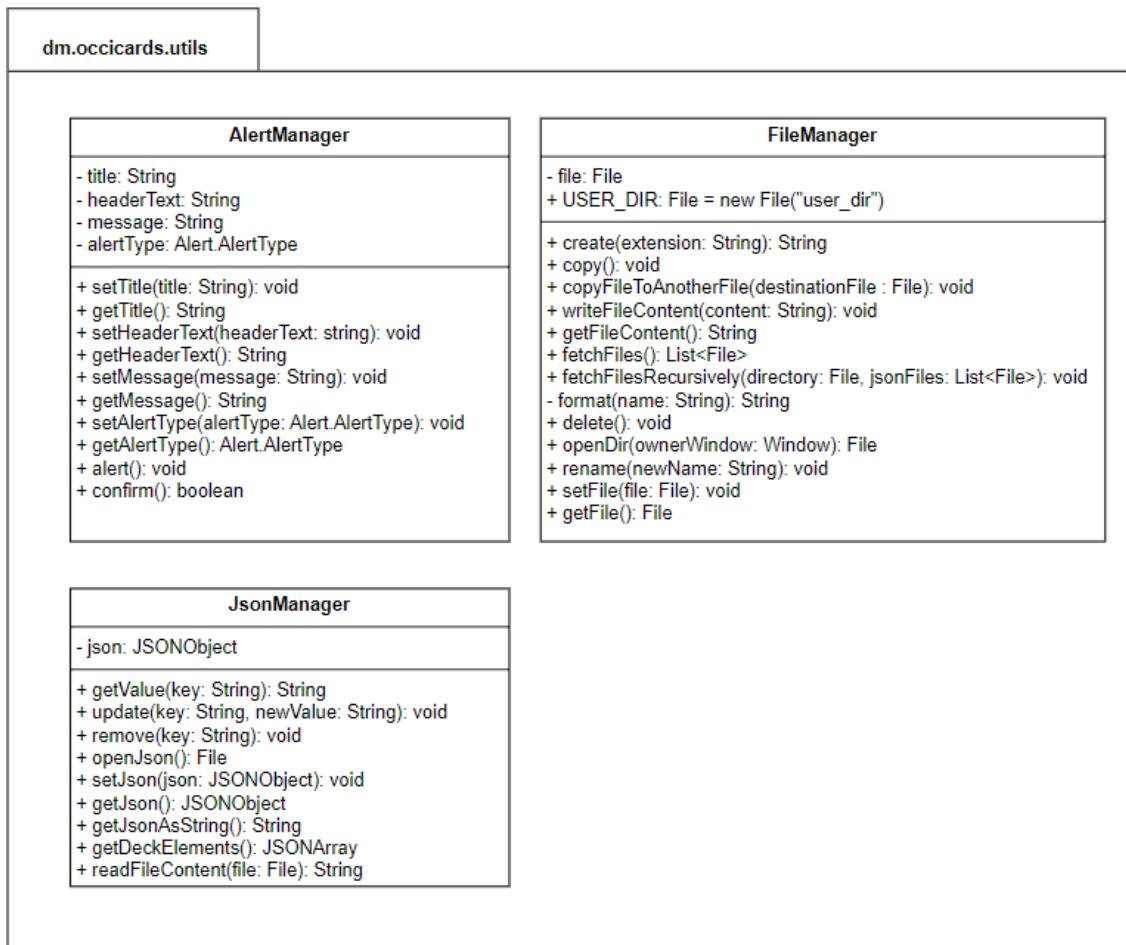
I - Package dm.occicards



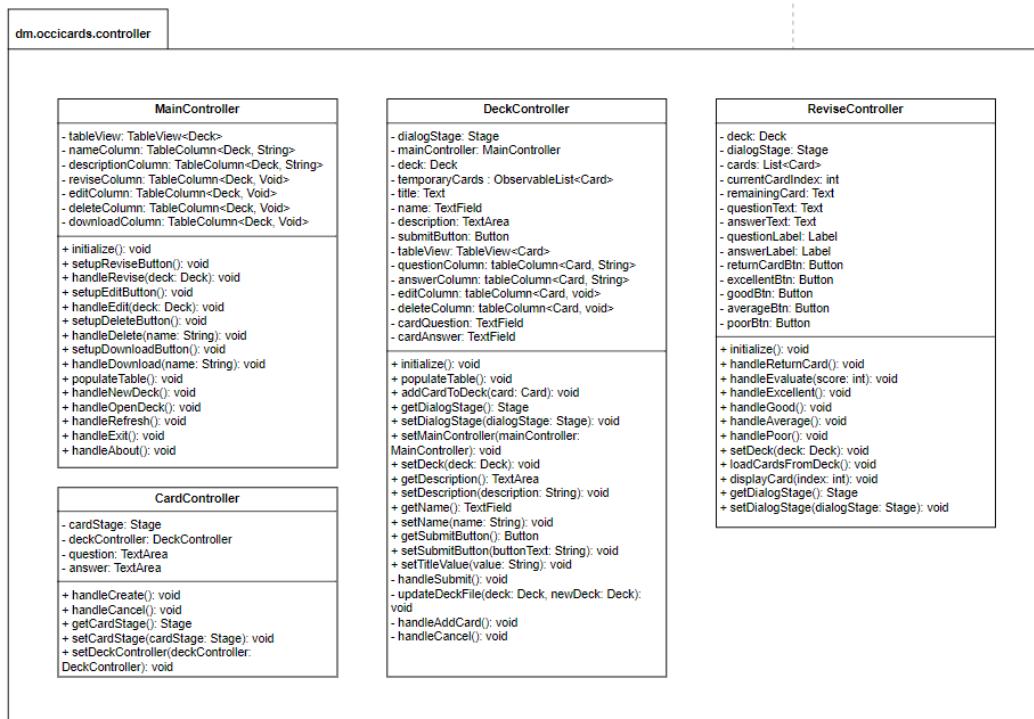
II - Package dm.occicards.model



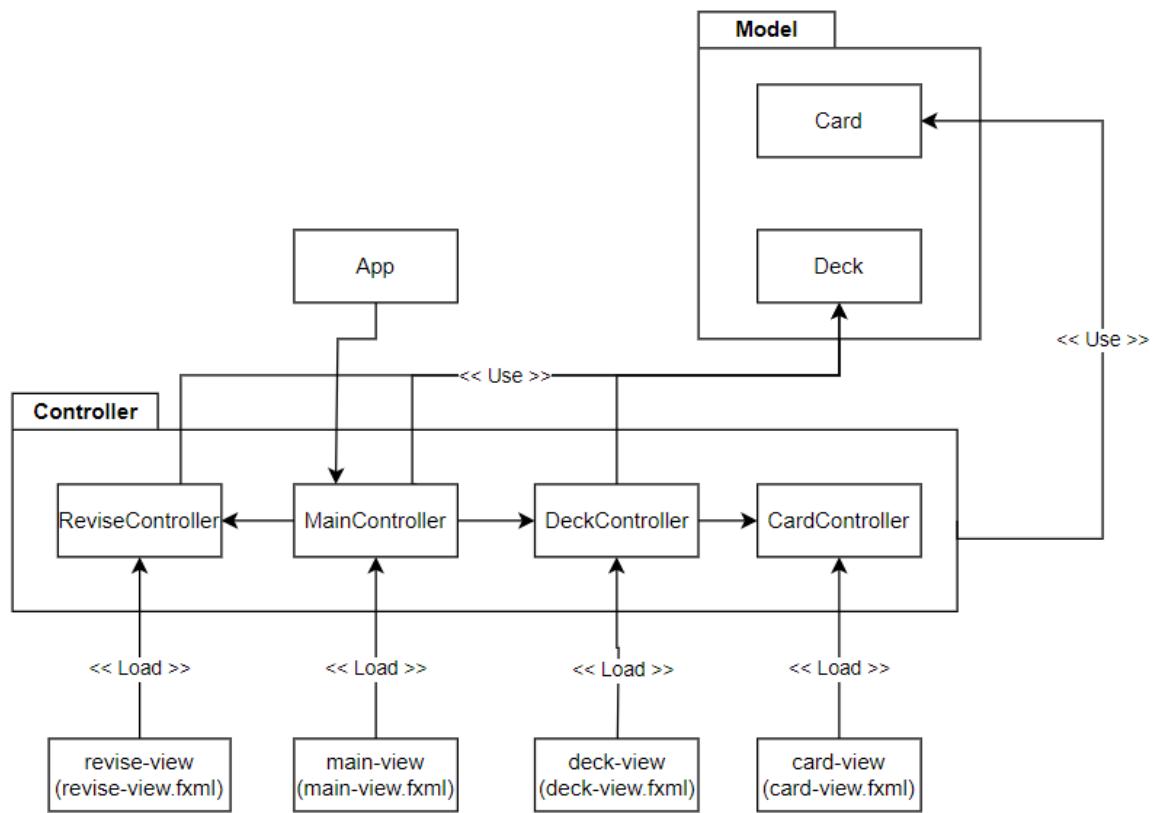
III - Package dm.occicards.utils



IV - Package dm.occicards.controller



V- Liens



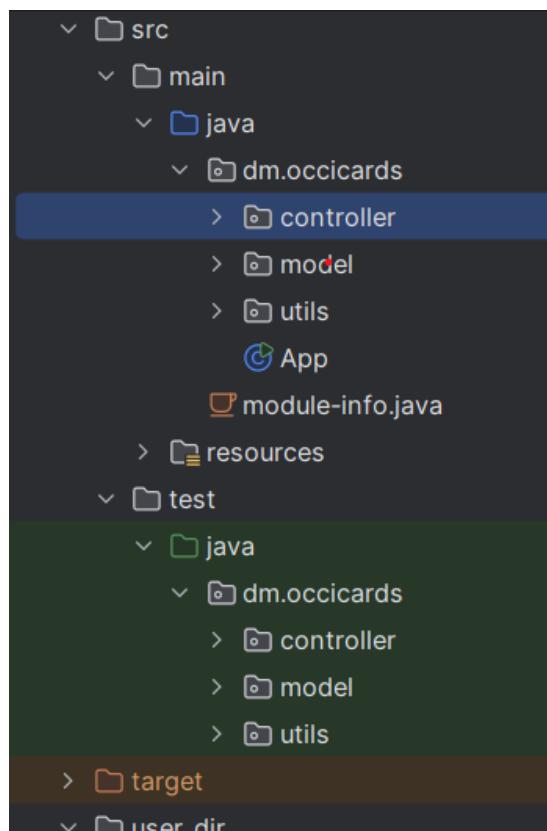
2) Analyse technique

A) Généralités

A1) Présentation de l'environnement technique

- Langage de programmation : JAVA JDK Temurin 21.0.6
- Librairie d'interface graphique : JavaFX
- Gestion des dépendance : Apache Maven
- Langage de donnée : JSON, FXML
- Environnement de développement : Intelij IDEA community
- Versioning : Git
- Système d'exploitation : Windows 11
- Langue de programmation : Anglaise

A2) Structure du projet



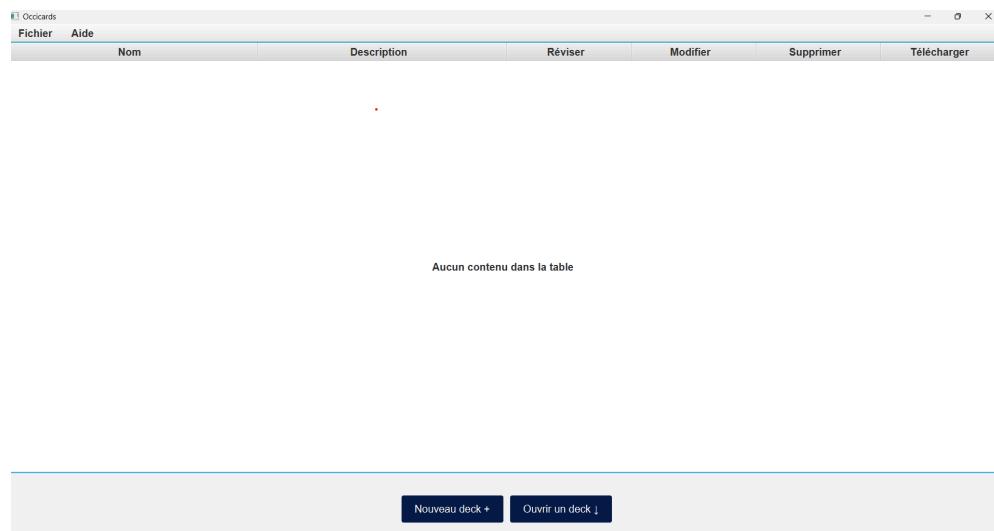
A3) Modèle Vue Contrôleur

- Le **modèle** représente la couche de données et la logique métier de l'application. Il gère les données.
- La **vue** gère l'affichage des données pour l'utilisateur, définissant ainsi l'interface utilisateur (UI) et la manière dont les informations sont présentées. Dans une application JavaFX, la vue est souvent créée avec le langage de balisage FXML, qui offre plusieurs avantages :
 - **Séparation du Code** : FXML permet de séparer la définition de l'UI du code Java, rendant le projet plus organisé.
 - **Organisation Simple** de l'UI : FXML facilite l'organisation des composants de l'interface.
 - **Stylisation avec CSS** : FXML permet d'appliquer des feuilles de style CSS pour personnaliser l'apparence.
 - **Liaison avec le Contrôleur** : Les éléments FXML doivent être liés à un contrôleur pour gérer les interactions utilisateur.
- Le **contrôleur** représente la couche logique de l'application, servant de pont entre les modèles et les vues. Il orchestre les interactions en capturant les actions de l'utilisateur dans l'interface graphique et en exécutant les méthodes nécessaires pour traiter ces actions

B) Application

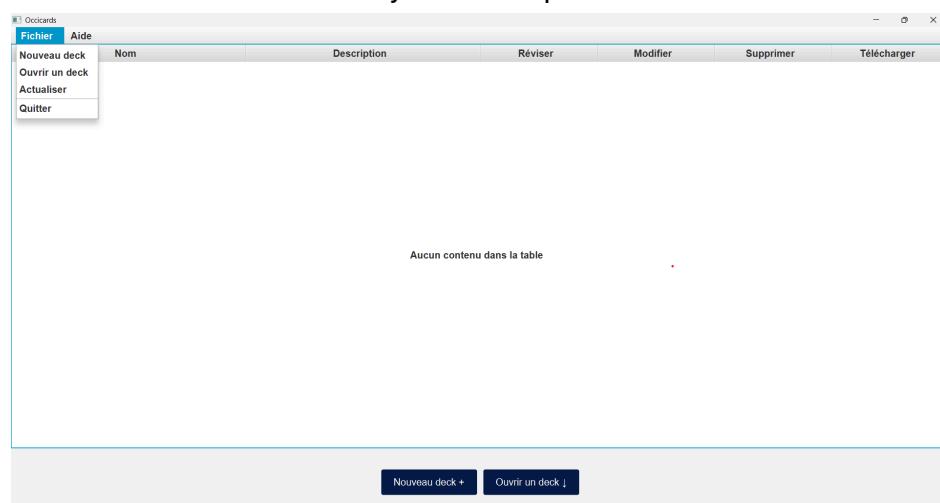
B1) Vue principale

Le "MainController" initialise la scène principale et l'associe à la fenêtre via les composants définis dans le fichier main-view.fxml. Cette fenêtre constitue l'interface principale de l'application.

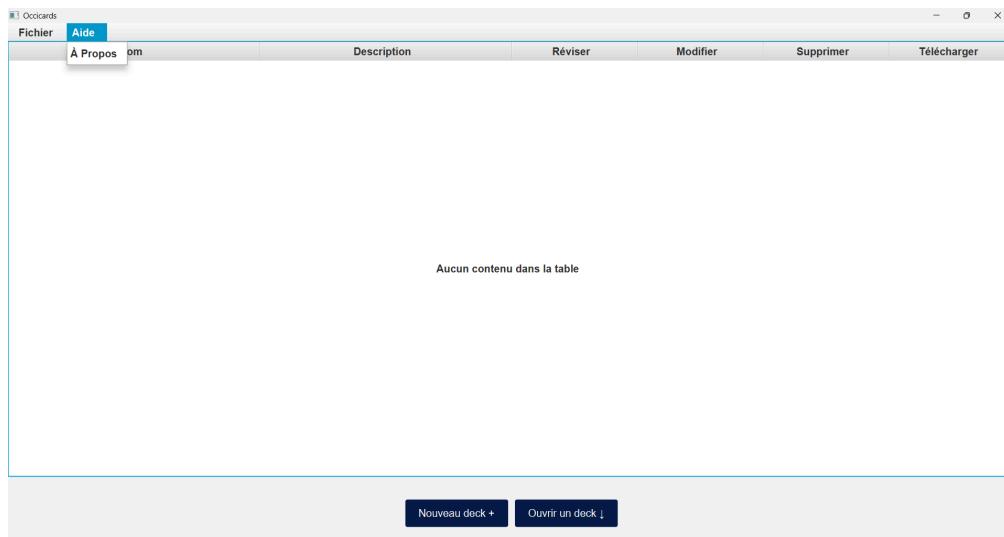


I - Menus

La scène comprend un menu principal, lui-même divisé en plusieurs sous-menus. Le sous-menu "Fichier" permet d'accéder à diverses fonctionnalités : ouvrir la fenêtre de création de deck, charger un fichier JSON existant, actualiser les composants de la scène, et quitter le programme. En outre, l'application peut être fermée en cliquant sur le bouton de fermeture de la fenêtre de votre système d'exploitation.



Le sous-menu "Aide" permet d'afficher une alerte contenant des informations générales sur l'application.



II - Collection des decks

Le deuxième grand composant de la scène est la **collection**, qui permet d'afficher et de gérer les decks via des fonctionnalités CRUD. Cette collection présente les informations principales de chaque deck, telles que le **nom** et la **description**, accompagnées de plusieurs boutons d'action.

Parmi ces boutons, le premier permet de **visualiser le deck** et de **lancer une session d'apprentissage** à l'aide des flashcards créées au préalable. Viennent ensuite les boutons **Modifier** et **Supprimer**, correspondant aux opérations classiques du CRUD. Enfin, un bouton de **téléchargement** permet d'exporter le deck au format **JSON** dans le répertoire de votre choix. Cette fonctionnalité facilite le **partage de decks entre élèves**, ou de la part d'un **enseignant vers ses étudiants**.

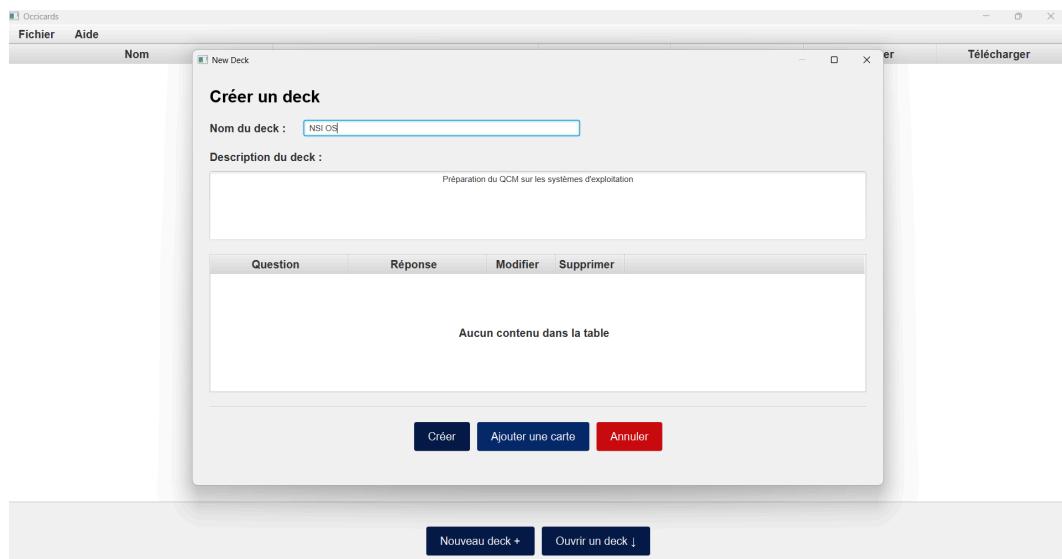
Enfin, deux boutons permettent d'ouvrir un deck existant ou de créer un nouveau deck.

B2) Vue du deck

Lorsque l'utilisateur crée un nouveau deck, le DeckController charge la vue deck-view.fxml. Un deck se compose de trois éléments : un nom, une description et une collection de cartes (Card).

Les champs "nom" et "description" doivent respecter certaines règles de validité :

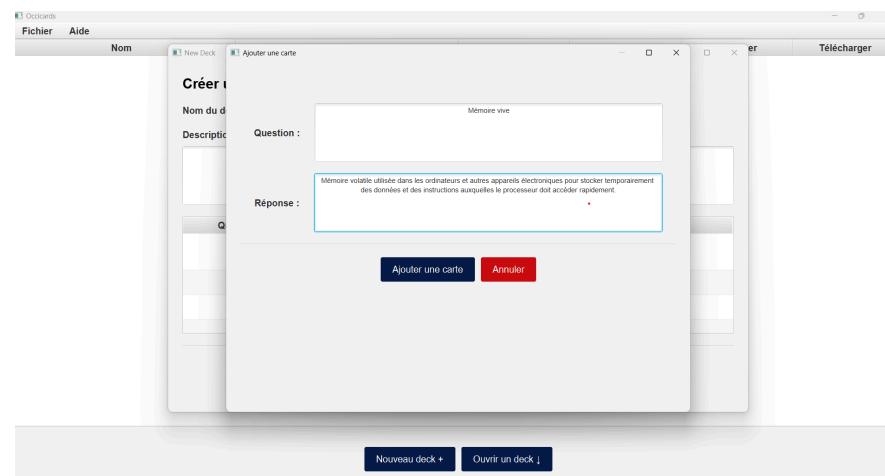
- **Nom** : obligatoire, sans caractères spéciaux (à l'exception de l'espace), et limité à 50 caractères.
- **Description** : obligatoire et limitée à 300 caractères.

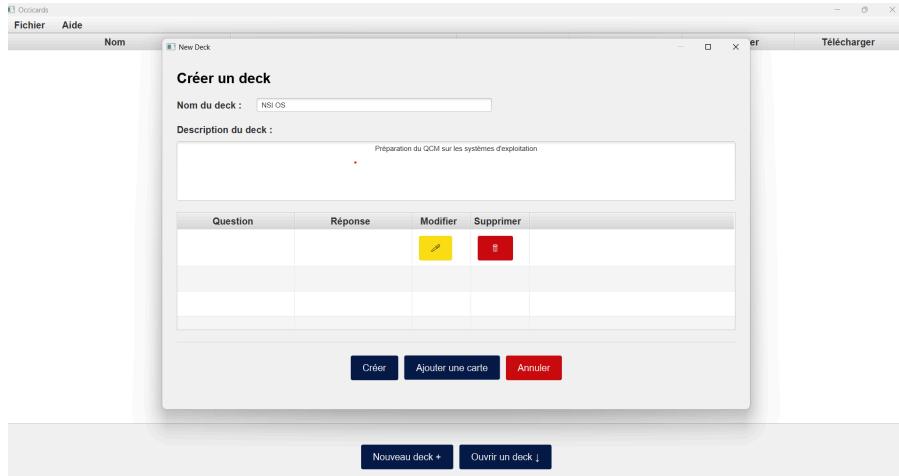


L'utilisateur peut ajouter des cartes à son deck. Un deck sans carte n'est pas valide ; il est donc conseillé d'ajouter au moins deux cartes. Une fois les cartes ajoutées, l'utilisateur pourra créer son deck.

B3) Vue de carte

Le DeckController invoque le CardController, qui à son tour ouvre la vue card-view dans une nouvelle fenêtre. Une carte est constituée de deux éléments principaux : une question et une réponse. Une fois créée, cette carte est ajoutée à une collection temporaire. Le tableau de la vue deck-view se met alors à jour automatiquement pour refléter cette nouvelle entrée, permettant à l'utilisateur de visualiser immédiatement les modifications apportées à son deck.

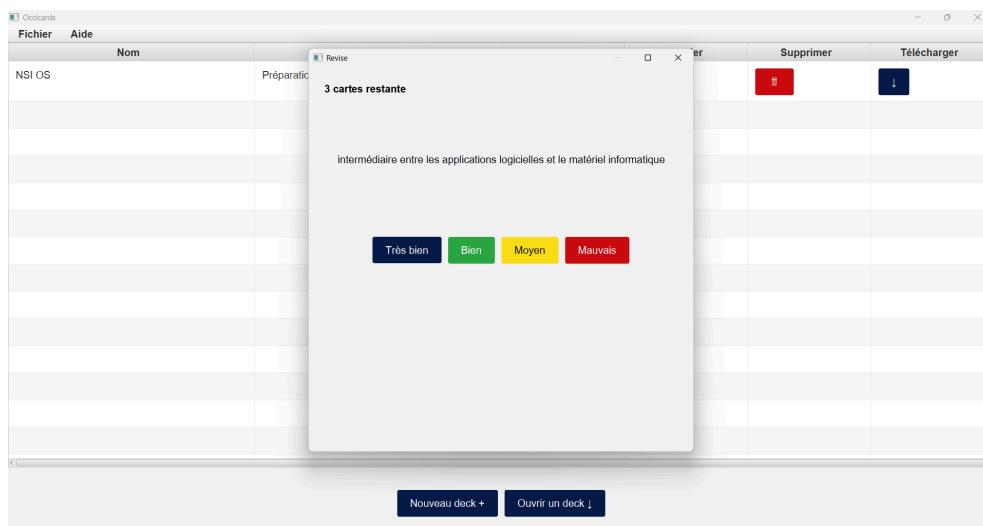


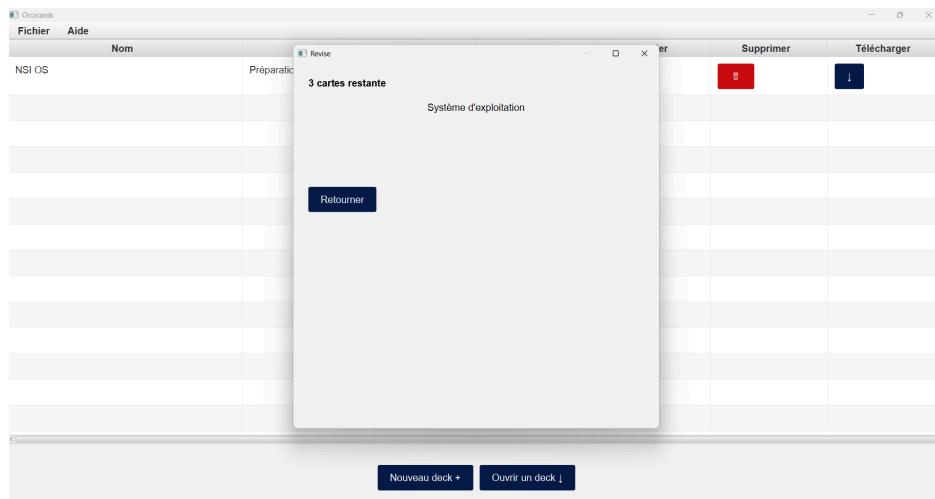


B4) Vue de révision

Le MainController utilise sa vue main-view.fxml pour générer un bouton de révision pour chaque deck. Ce bouton permet d'invoquer le ReviseController, qui lance ensuite la vue revise-view.fxml. Le système de révision est simple et intuitif : une question s'affiche à l'écran, et l'utilisateur peut retourner la carte pour révéler la réponse. Ensuite, il peut s'auto-évaluer en fonction de sa compréhension.

Si l'utilisateur sélectionne "mauvais", la carte est déplacée à l'index suivant de la collection, lui permettant de la revoir rapidement. Si la réponse est jugée "bien", la carte est déplacée de deux index, réduisant ainsi la fréquence de ses apparitions. Enfin, si l'évaluation est "très bien", la carte est retirée du deck, indiquant une maîtrise totale du contenu. Le processus se poursuit jusqu'à ce que toutes les cartes soient parfaitement maîtrisées. Un indicateur affiche en permanence le nombre de cartes restantes, permettant à l'utilisateur de suivre sa progression en temps réel.





C) Tests unitaires

C1) Utilité

Les **tests unitaires** vérifient que chaque composant d'un programme fonctionne correctement de manière isolée, permettant de détecter rapidement les bugs et de faciliter la maintenance. Ils servent aussi de documentation vivante, aidant à comprendre le comportement attendu et à prévenir les régressions.

En Java, les tests unitaires peuvent être réalisés sans bibliothèque tierce. Cependant, **JUnit**, une bibliothèque open source, facilite ce processus avec des annotations, des assertions enrichies et des extensions, rendant les tests plus efficaces et intuitifs.

J'ai réalisé des test unitaire pour les classes suivantes : MainController.java, DeckController.java, CardController.java, ReviseController.java, Deck.java, Card.java, AlertManager.java, FileManager.java, JsonManager.java

C2) Exemple de test unitaire réalisé sur Card.java

```
package dm.occicards.model;

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.*;

class CardTest {
    private Card card;

    @BeforeEach
    void setUp() { this.card = new Card("Question 1", "Réponse 1"); }

    @Test
    void getQuestion() { assertEquals("Question 1",
this.card.getQuestion()); }

    @Test
    void setQuestion() {
        this.card.setQuestion("Question 2");
        assertEquals("Question 2", this.card.getQuestion());
    }

    @Test
    void getAnswer() { assertEquals("Réponse 1", this.card.getAnswer());
}

    @Test
    void setAnswer() {
        this.card.setAnswer("Réponse 2");
        assertEquals("Réponse 2", this.card.getAnswer());
    }
}
```

C3) Résultats

✓	✓ <default package>	213 ms
✓	✓ FileManagerTest	165 ms
✓	✓ testCopy()	109 ms
✓	✓ testCopyFileToAnotherFi	7 ms
✓	✓ testGetFile()	6 ms
✓	✓ testFetchFiles()	8 ms
✓	✓ testCreate()	13 ms
✓	✓ testDelete()	4 ms
✓	✓ testSetFile()	4 ms
✓	✓ testRename()	5 ms
✓	✓ testWriteFileContent()	9 ms
>	✓ JsonManagerTest	43 ms
>	✓ CardTest	1 ms
>	✓ DeckTest	3 ms
✓	✓ AlertManagerTest	1 ms
✓	✓ setHeaderText()	
✓	✓ getAlertType()	
✓	✓ setMessage()	
✓	✓ setAlertType()	
✓	✓ setTitle()	
✓	✓ getHeaderText()	
✓	✓ getTitle()	
✓	✓ getMessage()	1 ms

D) Documentation technique

La documentation technique est générée à l'aide de l'outil Javadoc, un puissant générateur de documentation intégré à l'écosystème Java. Cet outil analyse le code source afin de produire une documentation complète et lisible, permettant aux développeurs de mieux comprendre les fonctionnalités, les classes, les méthodes et les attributs de l'application. Grâce à Javadoc, la structure du projet est clairement mise en évidence et chaque élément est soigneusement documenté, ce qui facilite la maintenance et l'évolution du code. Pour accéder à la documentation générée, ouvrez le fichier 'index.html' situé dans le répertoire '/occicards/target/site/apidocs'.

The screenshot shows a Javadoc-generated index.html page. At the top, there's a navigation bar with tabs: OVERVIEW, PACKAGE, CLASS (which is highlighted in orange), USE, TREE, INDEX, and HELP. Below the navigation bar, there are links for SUMMARY, NESTED, FIELD, CONSTR, METHOD and DETAIL: FIELD, CONSTR, METHOD. On the right, there's a search bar with a magnifying glass icon and the word 'Search'. The main content starts with the package declaration: 'Package dm.occicards.utils'. Then it defines the 'Class AlertManager' which extends 'java.lang.Object'. A brief description follows: 'Utility class for managing and displaying alerts in the application.' Below this is a 'Constructor Summary' section with a table for 'Constructors'. The table has columns for 'Constructor' and 'Description'. The constructor is defined as: 'AlertManager(String title, String headerText, String message, javafx.scene.control.Alert.AlertType alertType)'. The description for this constructor is: 'Constructs a new AlertManager with the specified title, header text, message, and alert type.' Finally, there's a 'Method Summary' section with a table for 'Concrete Methods'. The table has columns for 'Modifier and Type', 'Method', and 'Description'. It lists three methods: 'void alert()', 'boolean confirm()', and 'javafx.scene.control.Alert.AlertType getAlertType()'. The descriptions for these methods are: 'Displays an alert dialog with the specified properties.', 'Displays a confirmation dialog and returns true if the user confirms the action.', and 'Gets the type of the alert.' respectively.

Conclusion

Le projet OcciCards offre une solution efficace pour un apprentissage interactif et personnalisé, grâce à la répétition espacée et au Système Leitner. Développée en Java avec JavaFX et Maven, l'application est robuste et bien structurée. Les tests unitaires avec JUnit assurent sa fiabilité et facilitent les mises à jour futures.

Malgré les difficultés rencontrées, telles que le manque de temps et l'apprentissage IntelliJ IDEA, la gestion des dépendances avec Maven, la génération de la documentation technique et de JavaFX, je suis arrivé à bout du projet.

OcciCards améliore la rétention des connaissances et démontre la capacité de Sabatech à créer des outils numériques performants pour l'éducation.