

PROJET PC3R - S2-24

---

## Site Web de Mémorisation de Vocabulaire

---

Professeur :  
M. Romain Demangeon

*Haotian XUE 21305395*  
*Hejun CAO 21304526*

Mai 2024

M1 - INFO - STL

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Présentation des Fonctionnalités</b>	<b>2</b>
<b>3</b>	<b>Processus Pratique</b>	<b>2</b>
3.1	Maven et Tomcat	2
3.2	Gestion de Serveur (Back-end)	2
3.3	Gestion de Client(Front-end)	4
3.3.1	Structure du Projet	4
3.3.2	Fichiers JSP	4
3.3.3	Problèmes Rencontrés et Solutions	5
3.3.4	Implémentation des Fonctionnalités Front-end	5
3.4	API extension (pour traduire)	6
<b>4</b>	<b>Fonctionnalités du Projet</b>	<b>7</b>
4.1	Gestion des Utilisateurs	7
4.2	Gestion des Groupes de Vocabulaire	7
4.3	Gestion du Vocabulaire	7
4.4	Apprentissage et Test	8
4.5	Analyse des Résultats des Tests	8
<b>5</b>	<b>Autres Problèmes et Solutions</b>	<b>8</b>
5.1	Problème de Connexion à la Base de Données	8
5.2	Problème de Synchronisation des Données Front-end/Back-end	8
5.3	Problème de Perte des Données du Formulaire	9
5.4	Problème de Pagination des Requêtes	9
<b>6</b>	<b>Conclusion</b>	<b>9</b>
<b>7</b>	<b>Affichage du travail du projet</b>	<b>10</b>

# 1 Introduction

Ce rapport décrit le processus de développement d'une plateforme en ligne dédiée à la mémorisation de vocabulaire. En tant qu'étudiant, on a participé à la conception et à la réalisation de ce projet. Grâce à ce projet, on a non seulement amélioré nos compétences en programmation, mais on a également appris à appliquer ces compétences dans un projet concret.

## 2 Présentation des Fonctionnalités

Ce site web de mémorisation de vocabulaire inclut principalement les fonctionnalités suivantes :

- Gestion des utilisateurs : les utilisateurs peuvent s'inscrire, se connecter, modifier leurs informations personnelles et se déconnecter.
- Gestion des groupes de vocabulaire : les utilisateurs peuvent créer, voir, modifier et supprimer des groupes de vocabulaire.
- Gestion du vocabulaire : les utilisateurs peuvent ajouter, voir, modifier et supprimer des mots dans des groupes spécifiques.
- Apprentissage et test : les utilisateurs peuvent évaluer leur mémorisation via des tests de vocabulaire aléatoires et voir leurs résultats.
- Analyse des résultats des tests : les utilisateurs peuvent consulter leurs historiques de tests et des analyses détaillées des données.

## 3 Processus Pratique

Pour initialiser le projet, les détails sont dans le fichier README à la racine du projet. Veuillez suivre les instructions fournies pour configurer et démarrer correctement le projet.

### 3.1 Maven et Tomcat

Tout d'abord, Maven est utilisé pour la construction et la gestion des dépendances du projet, aidant à automatiser et standardiser la construction du projet. Tomcat est utilisé pour exécuter et déployer l'application Web Java, fournit un conteneur Servlet et des fonctionnalités de serveur Web. Le projet a une structure séparée entre le front-end et le back-end. Le back-end est responsable de la logique métier, tandis que le front-end s'occupe de l'interface utilisateur. Le back-end est écrit en Java, et le front-end utilise JSP (HTML, CSS, JavaScript).

### 3.2 Gestion de Serveur (Back-end)

#### Couche PO (Persistent Object)

Rôle : La couche PO est responsable de la correspondance entre les tables de la base de données et les objets Java. Chaque classe PO correspond à une table de la base de données et est utilisée pour la persistance des données. Contenu spécifique :

- User : Classe PO représentant les informations utilisateur, incluant ID, nom d'utilisateur, mot de passe, surnom, avatar, humeur, etc.
- Word : Classe PO représentant les informations de vocabulaire, incluant ID, titre, contenu, ID du groupe, date de publication, etc.
- WordGroup : Classe PO représentant les informations de groupe de vocabulaire, incluant ID du groupe, nom, description, ID utilisateur, etc.

- TestResult : Classe PO représentant les résultats des tests, incluant ID, ID utilisateur, ID du groupe, nombre total de mots, nombre de bonnes réponses, taux de précision et temps écoulé.

## Couche VO (Value Object)

Rôle : La couche VO est utilisée pour transférer des données entre le front-end et le back-end. Semblable à PO, mais VO n'est généralement pas directement mappée à des tables de base de données, elle est utilisée pour encapsuler les structures de données renvoyées par le service au front-end. Contenu spécifique :

- ResultInfo<T> : Utilisé pour encapsuler les résultats des opérations renvoyées par le back-end, incluant le code d'état, le message et l'objet de données retourné. T est un type générique pouvant être de n'importe quel type.

## Couche DAO (Data Access Object)

Rôle : La couche DAO est responsable des interactions avec la base de données, exécutant des opérations de création, lecture, mise à jour et suppression des données. Contenu spécifique :

- UserDao : Gère les opérations de base de données liées aux utilisateurs, telles que l'ajout, la consultation et la mise à jour des informations utilisateur.
- WordDao : Gère les opérations de base de données liées aux mots de vocabulaire, telles que l'ajout, la consultation de la liste, et la mise à jour des informations des mots.
- WordGroupDao : Gère les opérations de base de données liées aux groupes de vocabulaire, telles que l'ajout, la consultation de la liste, et la suppression des groupes.
- TestResultDao : Gère les opérations de base de données liées aux résultats des tests, telles que la sauvegarde et la consultation des résultats des tests.

## Couche Service

Rôle : La couche Service traite la logique métier, appelle les méthodes de la couche DAO pour les opérations de données et effectue les traitements et conversions nécessaires. Contenu spécifique :

- UserService : Traite la logique métier liée aux utilisateurs, comme l'inscription, la connexion et la mise à jour des informations personnelles.
- WordService : Traite la logique métier liée aux mots de vocabulaire, comme l'ajout ou la mise à jour de mots, la pagination des listes de mots, la suppression de mots, et la sauvegarde des résultats de tests.
- WordGroupService : Traite la logique métier liée aux groupes de vocabulaire, comme l'ajout ou la mise à jour de groupes, la consultation des listes de groupes et la suppression de groupes.
- TestResultService : Traite la logique métier liée aux résultats des tests, comme la sauvegarde des résultats et la consultation des résultats des utilisateurs.

## Couche Servlet

Rôle : La couche Servlet traite les requêtes du front-end, appelle les méthodes de la couche Service pour le traitement des affaires, et renvoie les résultats au front-end. La couche Servlet correspond à la couche contrôleur (Controller) dans l'architecture MVC. Contenu spécifique :

- UserServlet : Traite les requêtes liées aux utilisateurs, comme la connexion, la déconnexion, l'inscription, la vérification de l'unicité du surnom et la mise à jour des informations utilisateur.
- WordServlet : Traite les requêtes liées aux mots de vocabulaire, comme l'affichage des listes de mots, l'ajout ou la mise à jour de mots, la suppression de mots et la traduction de mots.
- WordGroupServlet : Traite les requêtes liées aux groupes de vocabulaire, comme l'affichage des listes de groupes, l'ajout ou la mise à jour de groupes et la suppression de groupes.

- StudyServlet : Traite les requêtes liées à l'apprentissage et aux tests, comme le démarrage de l'apprentissage et la sauvegarde des résultats des tests.
- ReportServlet : Traite les requêtes liées aux rapports de résultats de tests, comme l'affichage des listes de résultats de tests.

### 3.3 Gestion de Client(Front-end)

#### 3.3.1 Structure du Projet

Les fichiers front-end incluent principalement des ressources statiques (CSS, JS, images, etc.) et des fichiers JSP pour chaque module fonctionnel, organisés comme suit :

- statics
  - css : Contient les fichiers de styles (comme index.css)
  - js : Contient les fichiers JavaScript (comme config.js, group.js, mem.js, etc.)
  - images : Contient les ressources d'images
- webapp
  - group : Contient les pages liées à la gestion des groupes de vocabulaire (comme group.jsp)
  - mem : Contient les pages liées à la gestion de la mémorisation (comme mem.jsp)
  - user : Contient les pages liées à la gestion des utilisateurs (comme profile.jsp)
  - word : Contient les pages liées à la gestion et à l'apprentissage des mots de vocabulaire (comme memory.jsp, publish.jsp)
  - autres pages (comme index.jsp, login.jsp, signup.jsp, reports.jsp, study.jsp)

#### 3.3.2 Fichiers JSP

Les fichiers JSP combinent HTML, CSS et JavaScript pour créer des pages web dynamiques. Chaque fichier JSP contient généralement la mise en page de la page, des formulaires, l'affichage des données et des scripts pour interagir avec le back-end.

##### **index.jsp**

Fonction : Page d'accueil du projet, fournissant des liens vers les différentes fonctionnalités. Contenu :

- Barre de navigation : Inclut des liens vers la page d'accueil, la publication de vocabulaire, la gestion des catégories, la mémorisation, le rapport de données, etc.
- Bouton de connexion/déconnexion : Affiche le bouton de déconnexion après la connexion de l'utilisateur, sinon affiche le bouton de connexion.
- Carrousel d'images : Affiche des images statiques.

##### **login.jsp**

Fonction : Fournit une interface de connexion utilisateur. Contenu :

- Formulaire de connexion : Contient les champs de nom d'utilisateur, mot de passe et l'option de se souvenir de moi.
- Script JavaScript : Valide les entrées utilisateur et envoie une requête de connexion au back-end via Ajax.

##### **signup.jsp**

Fonction : Fournit une interface d'inscription utilisateur. Contenu :

- Formulaire d'inscription : Contient les champs de nom d'utilisateur et mot de passe.
- Script JavaScript : Valide les entrées utilisateur et envoie une requête d'inscription au back-end via Ajax.

## **word.jsp**

Fonction : Fournit des fonctionnalités de gestion des mots de vocabulaire, incluant la consultation, l'ajout, la modification et la suppression de mots. Contenu :

- Liste des mots : Affiche tous les mots avec des boutons pour les éditer et les supprimer.
- Script JavaScript : Charge les données des mots de manière asynchrone et gère les actions des utilisateurs.

### **3.3.3 Problèmes Rencontrés et Solutions**

#### **Problème de validation des formulaires**

Problème : Lors de l'implémentation de la connexion et de l'inscription des utilisateurs, la validation des formulaires n'était pas suffisante, permettant l'entrée de données non valides. Solution : Utiliser JavaScript pour valider les formulaires côté front-end, en assurant que les données saisies respectent le format requis. Effectuer également une validation côté back-end pour assurer la sécurité et la validité des données.

#### **Problème de vitesse de chargement des pages**

Problème : Certaines pages prenaient du temps à charger, affectant l'expérience utilisateur. Solution : Utiliser des requêtes asynchrones (Ajax) pour charger les données, réduisant ainsi le temps de chargement des pages. Optimiser le chargement des ressources statiques et utiliser des CDN pour accélérer le chargement des ressources externes.

#### **Problème de compatibilité entre navigateurs**

Problème : Les pages avaient des styles et des fonctionnalités inconsistantes selon les navigateurs. Solution : Utiliser HTML5 et CSS3 standards pour concevoir les pages, éviter les propriétés spécifiques aux navigateurs obsolètes. Utiliser des fichiers de réinitialisation CSS pour uniformiser les styles par défaut des différents navigateurs.

### **3.3.4 Implémentation des Fonctionnalités Front-end**

#### **Inscription et Connexion Utilisateur**

Validation des formulaires de la page d'inscription et de connexion, incluant les champs obligatoires, la validation de format, etc. Utilisation d'Ajax pour la soumission des formulaires, évitant le rechargement des pages et améliorant l'expérience utilisateur. Affichage des informations de base de l'utilisateur dans la barre de navigation après la connexion, avec une option de déconnexion.

#### **Gestion des mots de vocabulaire**

Page de liste des mots de vocabulaire, affichant le titre, le contenu, la date de publication, etc. Ajout, modification et suppression des mots de vocabulaire via des fenêtres modales. Utilisation de la pagination pour afficher la liste des mots, améliorant la vitesse de chargement des pages.

#### **Gestion des groupes de vocabulaire**

Page de liste des groupes, affichant le nom et la description des groupes. Ajout, modification et suppression des groupes via des fenêtres modales. Sélection du groupe lors de l'ajout ou de la modification des mots de vocabulaire.

## Apprentissage et Test

Page d'apprentissage, affichant aléatoirement les mots de vocabulaire pour l'étude et la révision. Page de test, générant des questions de test basées sur le contenu des mots de vocabulaire. Affichage des résultats détaillés du test après soumission, incluant le taux de précision et le temps écoulé.

### 3.4 API extension (pour traduire)

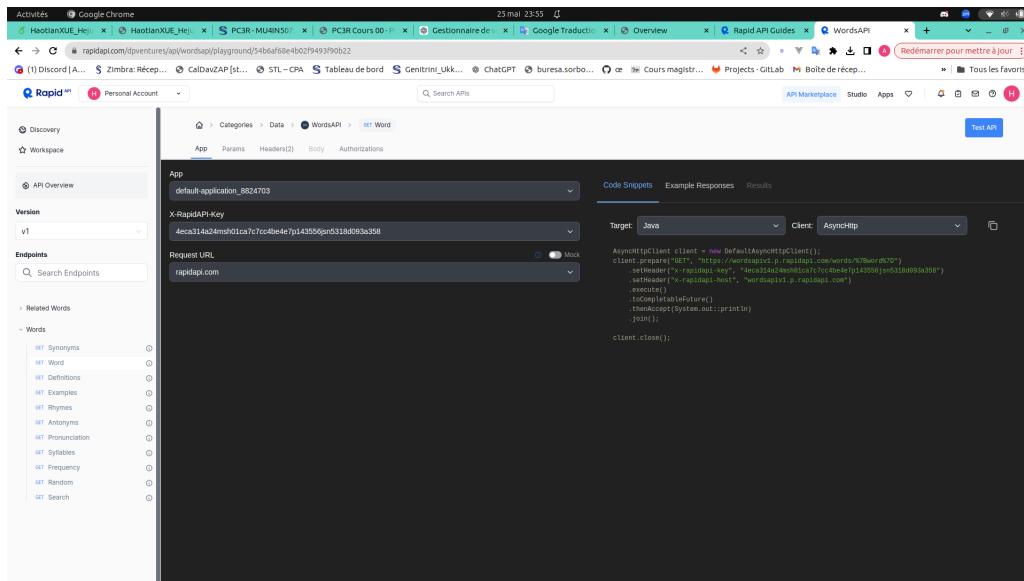


FIGURE 1 – WordsAPI sur Internet

## Obtenir la clé API

Tout d'abord, vous devez vous inscrire sur le site officiel de WordsAPI et obtenir une clé API. Cela est nécessaire pour authentifier vos demandes.

## Configuration de la requête backend

Utilisez le backend Java pour effectuer des requêtes API. Vous pouvez utiliser HttpURLConnection ou des bibliothèques similaires (comme Apache HttpClient) pour envoyer des requêtes HTTP et traiter les réponses.

## Analyse de la réponse API

La réponse de l'API est généralement au format JSON. Vous devez analyser ces données JSON et les convertir en objets Java. Vous pouvez utiliser des bibliothèques telles que Jackson ou Gson pour gérer l'analyse JSON.

## Intégration frontend

Le frontend appelle les interfaces fournies par le backend pour obtenir les données de vocabulaire et les afficher sur la page.

## 4 Fonctionnalités du Projet

### 4.1 Gestion des Utilisateurs

Description de la fonctionnalité : Les utilisateurs peuvent s'inscrire, se connecter, modifier leurs informations personnelles (comme le surnom et l'avatar) et se déconnecter. Processus pratique :

- Inscription et Connexion : On a d'abord implémenté les fonctionnalités d'inscription et de connexion des utilisateurs. Lors de l'inscription, les utilisateurs saisissent un nom d'utilisateur et un mot de passe, qui sont ensuite stockés dans la base de données. Lors de la connexion, le système vérifie les informations saisies et, en cas de succès, stocke les informations utilisateur dans la session.
- Gestion du Profil : Après une connexion réussie, les utilisateurs peuvent accéder à leur page de profil pour modifier leur surnom, avatar et humeur. On a utilisé des requêtes Ajax pour valider l'unicité du surnom en temps réel.

Problèmes rencontrés et solutions :

- Problème de synchronisation des données entre le serveur et le client lors de la vérification de l'unicité du surnom.
- Solution : Utiliser des requêtes Ajax pour envoyer en temps réel des demandes de vérification au serveur et mettre à jour les indications sur la page en fonction des résultats.

### 4.2 Gestion des Groupes de Vocabulaire

Description de la fonctionnalité : Les utilisateurs peuvent créer plusieurs groupes de vocabulaire, chaque groupe contenant plusieurs mots. Processus pratique :

- Création et Consultation : Les utilisateurs peuvent créer de nouveaux groupes de vocabulaire via un formulaire de la page front-end, les informations du groupe étant stockées dans la base de données. Les utilisateurs peuvent également consulter tous les groupes créés.
- Modification et Suppression : On a implémenté les fonctionnalités de modification et de suppression des groupes via des formulaires et des requêtes Ajax pour des mises à jour sans rafraîchissement de la page.

Problèmes rencontrés et solutions :

- Problème de rafraîchissement de la page lors de la suppression d'un groupe, affectant l'expérience utilisateur.
- Solution : Utiliser des requêtes Ajax pour supprimer des groupes et mettre à jour la liste des groupes dynamiquement.

### 4.3 Gestion du Vocabulaire

Description de la fonctionnalité : Dans un groupe spécifique, les utilisateurs peuvent ajouter, consulter, modifier et supprimer des mots de vocabulaire. Processus pratique :

- Ajout et Consultation : Les utilisateurs peuvent ajouter de nouveaux mots dans un groupe spécifique, incluant le titre et le contenu du mot. Ils peuvent également consulter la liste de tous les mots dans ce groupe.
- Modification et Suppression : Les utilisateurs peuvent modifier les titres et le contenu des mots ou supprimer les mots non nécessaires, ces opérations étant réalisées via des formulaires et des requêtes Ajax.

Problèmes rencontrés et solutions :

- Problème de perte de données lors de l'ajout de mots, notamment en cas de rafraîchissement ou de redirection de la page.
- Solution : Utiliser le stockage local (localStorage ou sessionStorage) de JavaScript pour sauvegarder temporairement les données saisies, permettant de les restaurer après un rafraîchissement ou une redirection de la page.

## 4.4 Apprentissage et Test

Description de la fonctionnalité : Les utilisateurs peuvent évaluer leur mémorisation via des tests de vocabulaire aléatoires et voir leurs résultats. Processus pratique :

- Mode Apprentissage : Les utilisateurs peuvent choisir un groupe de vocabulaire à étudier, le système affichant aléatoirement les mots et leurs explications. Les utilisateurs peuvent utiliser la fonction de traduction pour obtenir plusieurs définitions du mot et choisir celle qui leur convient pour l'apprentissage.
- Mode Test : Les utilisateurs peuvent effectuer des tests de vocabulaire, le système générant des questions aléatoires et demandant aux utilisateurs de choisir la bonne explication. Pendant le test, le système enregistre les réponses des utilisateurs, incluant les bonnes et les mauvaises réponses, le taux de précision et le temps écoulé.

Problèmes rencontrés et solutions :

- Problème de précision ou de répétition des options de test.
- Solution : Utiliser JavaScript pour éliminer les doublons dans les options et les trier aléatoirement, assurant que chaque test présente des options aléatoires et uniques.

## 4.5 Analyse des Résultats des Tests

Description de la fonctionnalité : Les utilisateurs peuvent consulter leurs historiques de tests et des analyses détaillées des données. Processus pratique :

- Affichage des Résultats des Tests : Après chaque test, le système génère un rapport détaillé montrant les performances de l'utilisateur. Le rapport inclut la date du test, le nom du groupe, le nombre total de mots, le nombre de bonnes réponses, le taux de précision et le temps écoulé.
- Consultation des Historiques : Les utilisateurs peuvent consulter leurs historiques de tests et des analyses détaillées des données via la page de rapport.

Problèmes rencontrés et solutions :

- Problème d'affichage incomplet ou d'ordre incorrect des données dans les résultats des tests.
- Solution : Optimiser les requêtes SQL pour assurer l'intégrité et l'ordre correct des données, en effectuant des débogages et en ajustant les requêtes pour résoudre les problèmes d'affichage.

# 5 Autres Problèmes et Solutions

## 5.1 Problème de Connexion à la Base de Données

Problème : Au début du développement, il y avait des problèmes d'instabilité de connexion à la base de données, parfois une exception de délai d'attente était levée.

Solution : Optimiser la configuration du pool de connexions à la base de données, en augmentant le nombre initial de connexions et le nombre maximal de connexions pour assurer une connexion stable même en cas de forte concurrence. Vérifier et fermer régulièrement les connexions inactives pour éviter les fuites de connexions.

## 5.2 Problème de Synchronisation des Données Front-end/Back-end

Problème : Lors des opérations de données sur le front-end (comme la vérification de l'unicité du surnom), il y avait des problèmes de synchronisation des données entre le front-end et le back-end, affectant l'expérience utilisateur.

Solution : Utiliser des requêtes Ajax pour les validations de données en temps réel, en envoyant des requêtes au back-end et en recevant les résultats pour assurer la cohérence des données entre le front-end et le back-end. Optimiser le code front-end pour réduire les requêtes inutiles et améliorer la vitesse de réponse de la page.

### **5.3 Problème de Perte des Données du Formulaire**

Problème : Lors de l'ajout ou de la modification de mots de vocabulaire, il y avait des problèmes de perte des données du formulaire, notamment lors du rafraîchissement ou de la redirection de la page.

Solution : Utiliser la fonction de stockage local (localStorage ou sessionStorage) de JavaScript pour sauvegarder temporairement les données saisies par les utilisateurs, permettant de les restaurer après un rafraîchissement ou une redirection de la page. Cela évite la perte de données et améliore l'expérience utilisateur.

### **5.4 Problème de Pagination des Requêtes**

Problème : Lors de l'implémentation de la pagination pour les listes de mots de vocabulaire, il y avait des problèmes d'inexactitude des données ou de pagination irrationnelle.

Solution : Optimiser les requêtes SQL de pagination pour assurer l'exactitude des résultats. Ajouter une navigation de pagination et des messages d'information sur le front-end pour permettre aux utilisateurs de naviguer facilement entre les pages.

## **6 Conclusion**

Le développement de ce projet nous a permis de comprendre en profondeur les différentes étapes du développement d'applications Web, y compris l'interaction front-end/back-end, les opérations sur les bases de données et l'optimisation de l'expérience utilisateur. En résolvant des problèmes concrets, on a amélioré nos compétences en résolution de problèmes et en programmation. Ce projet nous a non seulement aidé à consolider les connaissances acquises, mais a également jeté une base solide pour notre avenir professionnel dans ce domaine.

## 7 Affichage du travail du projet

(a) login\_home

(b) login\_about

(c) login\_contact

(d) profil

(e) index

(f) publish

(g) group

(h) memory

(i) report

(j) wordlist dans un groupe

(k) apprendre les mots dans un groupe

MySQL Workbench interface showing the 'user' table structure and data. The table has columns: user\_id, username, upwd, nick, head, and last\_login. Data includes rows for users like 'root', 'zoe', 'admin', and 'admin123'.

(a) table user dans le database

MySQL Workbench interface showing the 'group' table structure and data. The table has columns: group\_id, groupname, groupdesc, and grouporder. Data includes rows for groups like 'Physique', 'maths', 'Informatique', and 'test'.

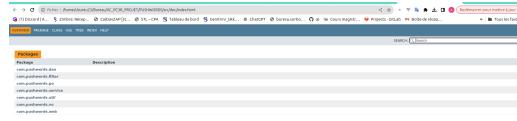
(b) table group dans le database

MySQL Workbench interface showing the 'word' table structure and data. The table has columns: word\_id, word, group\_id, title, content, and pubtime. Data includes rows for words like 'Hello', 'world', 'Python', and 'Java'.

(c) table word dans le database

MySQL Workbench interface showing the 'test\_result' table structure and data. The table has columns: result\_id, user\_id, group\_id, totalWords, correctAnswers, accuracy, timeTaken, and testDate. Data includes rows for test results from users like 'root' and 'zoe'.

(d) table test result dans le database



(e) javadoc