Université de Rouen Normandie

Faculté des Sciences et Techniques

Rapport mini projet Langages Web

Version : 1.0
Date : 26 Décembre 2024

Réalisé par :

- FEGHOUL Celina

- OUCHAOU Chafaa

Année Universitaire 2024-2025

Contents

1 Rappel **2**

2 Rôles et fonctionnalités de l'application **2**

3 Architecture de l'application **3**
3.1 BACKEND : . . . 3
3.2 End points : . . . 3

4 API Utilisateur **3**
4.1 Inscription . . . 3
4.2 Connexion . . . 4
4.3 Déconnexion . . . 4

5 API Grilles **4**
5.1 Retourner toutes les grilles . . . 4
5.2 Filtrer les grilles par niveau de difficulté et date de création . . . 5
5.3 Retourner une grille spécifique . . . 5
5.4 Ajouter une grille . . . 5

6 API Sauvegarde des Jeux **6**
6.1 Sauvegarder une grille . . . 6
6.2 Récupérer une grille sauvegardée . . . 6
6.3 Supprimer une grille sauvegardée . . . 6

7 API Administration **7**
7.1 Accéder à l'espace administrateur . . . 7
7.2 Se connecter en tant qu'administrateur . . . 7
7.3 Obtenir la liste des utilisateurs . . . 7
7.4 Supprimer un utilisateur . . . 8

8 Sécurité **11**
8.1 Sécurité des mots de passes : . . . 11
8.2 sécurité contre les attackes . . . 11
8.2.1 Injection SQL . . . 11

9 Points réussis **12**
9.1 Ajout de grilles . . . 12
9.2 Résolution des grilles . . . 12

10 Conclusion **12**

11 Perspectives d'avenir **12**

1 Rappel

Le but de ce mini projet est de créer une application de création et de résolution de mots croisés.

2 Rôles et fonctionnalités de l'application

Acteur	Fonctionnalités
Utilisateur anonyme	• Un utilisateur anonyme peux créer un compte. • Se connecter a son compte. • Consulter les grilles existantes. • Filtrer les grilles selon le niveau de difficulté et la date de publication. • Jouer a des grilles.
Utilisateur connecté	• Un utilisateur connecté a la possibilité de créer une grille. • Consulter les grilles. • Jouer a des grilles et les résoudre. • Les sauvegarder.
Administrateur de la plateforme	• Il n'existe qu'un seul administrateur par défaut et a comme identifiants : – **Mail** : admin@gmail.com – **Mot de passe** : 123456 • Il a la possibilité de créer un utilisateur et d'en supprimer. • Supprimer les grilles déjà créés.

3 Architecture de l'application

Pour répondre au besoin énoncé, nous avons jugé utile d'utiliser les principes que nous avons étudiés en cours, tels que l'utilisation de l'architecture MVC, PHP orienté objet pour la partie backend, ainsi que Javascript, HTML, AJAX et CSS pour la partie frontend. En ce qui concerne la base de données, nous avons utilisé un serveur MySQL.

Nous allons maintenant détailler notre architecture par couches.

3.1 BACKEND :

Nous avons opté pour l'utilisation des API REST, dont la réponse est de type JSON. Pour ce faire, notre architecture se base sur différents niveaux :

- **Controllers** : Ils traitent et valident les données, puis retournent une réponse sous un format JSON.

Le fichier `routes.php` représente le fichier le plus important dans l'implémentation du modèle MVC (Modèle, Vue, Contrôleur). Il constitue le fichier d'entrée permettant de traiter les requêtes HTTP entrantes et de les rediriger vers le contrôleur approprié. Son fonctionnement est le suivant :

- Il établit la connexion à la base de données pour rendre les controllers accessibles.
- Il parse l'URI pour récupérer le chemin.
- Chaque route instancie le contrôleur ainsi que la méthode appropriée.
- Dans le cas où la route demandée n'est pas définie, une réponse HTTP 404 est envoyée sous un format JSON.

3.2 End points :

Dans ce qui suit, nous allons parler des différents end points de notre API.

4 API Utilisateur

4.1 Inscription

URL : `http://localhost/projet-d-web/api/register`
Méthode : POST
Description : Permet à un nouvel utilisateur de s'inscrire dans l'application.
Données brutes :

```
{
    "username" : "MyTest",
    "email" : "MyTest1@gmail.com",
    "password" : "123456"
}
```

Réponse :

```
{
    "succes": "Utilisateur inscrit"
}
```

4.2 Connexion

URL : `http://localhost/projet-d-web/api/login`
Méthode : POST
Description : Permet à un utilisateur déjà inscrit de se connecter.
Données brutes :

```
{
    "email" : "MyTest1@gmail.com",
    "password" : "123456"
}
```

Réponse :

```
{
    "succes": "Utilisateur connecté"
}
```

4.3 Déconnexion

URL : `http://localhost/projet-d-web/api/logout`
Méthode : POST
Description : Permet de déconnecter un utilisateur.
Réponse :

```
{
    "succes": "Déconnexion réussie"
}
```

5 API Grilles

5.1 Retourner toutes les grilles

URL : `http://localhost/projet-d-web/api/games`
Méthode : GET
Description : Cette API retourne toutes les grilles disponibles dans la base de données.
Réponse :

```
[
    {
        "id": 58,
        "name": "test",
        "description": "test"
    },
    {
        "id": 66,
        "name": "test4",
        "description": "test4"
    }
]
```

5.2 Filtrer les grilles par niveau de difficulté et date de création

URL : `http://localhost/projet-d-web/api/games/filtered?niveau_difficulte=expert&date_de_creation=2024-12-21`
Méthode : GET
Description : Permet de filtrer les grilles selon le niveau de difficulté et la date de création.
Réponse :


```
[
    {
        "id": 58,
        "name": "test",
        "description": "test",
        "niveau_difficulte": "expert",
        "date_de_creation": "2024-12-21 14:42:54"
    }
]
```


5.3 Retourner une grille spécifique

URL : `http://localhost/projet-d-web/api/game/58`
Méthode : GET
Description : Permet de récupérer une grille spécifique à partir de son ID.
Réponse :


```
{
    "concatenatedGrid": "TELESCOPESETATORIONLRAUNCLELUMIEREIOURESNONNERASNONNEINBINN..,
    "rows": 10,
    "cols": 10,
    "horizontalDesc": "Les outils des astronomes,Louis XIV selon Louis XIV...",
    "verticalDesc": "Telle notre bonne vieille planète,Les astronomes sont toujours..."
}
```


5.4 Ajouter une grille

URL : `http://localhost/projet-d-web/api/addGrid`
Méthode : POST
Description : Permet d'ajouter une nouvelle grille à la base de données.
Données brutes :


```
{
    "nom": "test4",
    "description": "test4",
    "niveauDifficulte": "débutant",
    "colonnes": 2,
    "lignes": 2,
    "defHorizontales": ["A", "B"],
    "defVerticales": ["A", "B"],
    "casesNoire": ["1,1"],
    "solutions": "ABC"
}
```


Réponse :

```
{
    "succes": "Grille ajoutée avec succès."
}
```

6 API Sauvegarde des Jeux

6.1 Sauvegarder une grille

URL : `http://localhost/projet-d-web/api/saveGame?gridId=67`
Méthode : POST
Description : Permet à un utilisateur de sauvegarder une grille.
Données brutes :

```
{
    "solutionPartielle": "TELESCOPESETATORIONLRAUNCLELUMIEREIOURESNONNERAS..."
}
```

Réponse :

```
{
    "succes": "Grille sauvegardée avec succès"
}
```

6.2 Récupérer une grille sauvegardée

URL : `http://localhost/projet-d-web/api/getSavedGame/5`
Méthode : GET
Description : Permet de récupérer une grille précédemment sauvegardée.
Réponse :

```
{
    "id_saved_grids": 5,
    "solution_partielle": "newSol",
    "save_date": "2024-12-24 16:59:22",
    "nom": "test4",
    "nbr_lignes": 2,
    "nbr_colonnes": 2,
    "def_horizontales": "A,B",
    "def_verticales": "A,B",
    "cases_noire": "1,1",
    "solutions": "ABC"
}
```

6.3 Supprimer une grille sauvegardée

URL : `http://localhost/projet-d-web/api/deleteSaveGame/5`
Méthode : POST
Description : Permet à un utilisateur de supprimer une grille sauvegardée.
Réponse :

```
{
    "succes": "Grille supprimée avec succès"
}
```

7 API Administration

7.1 Accéder à l'espace administrateur

URL : `http://localhost/projet-d-web/api/admin`
Méthode : GET
Description : Permet d'accéder à l'espace d'administrateur de l'application.

7.2 Se connecter en tant qu'administrateur

URL : `http://localhost/projet-d-web/api/admin/login`
Méthode : POST
Données brutes :

```
{
    "email": "admin@gmail.com",
    "password": "123456"
}
```

Réponse :

```
{
    "youpi": "Utilisateur admin est connecté"
}
```

7.3 Obtenir la liste des utilisateurs

URL : `http://localhost/projet-d-web/api/admin/getUsers`
Méthode : GET
Réponse :

```
[
    {
        "username": "Celina8",
        "email": "cel@gmail.com"
    },
    {
        "username": "user",
        "email": "hdh@gmail.com"
    }
]
```

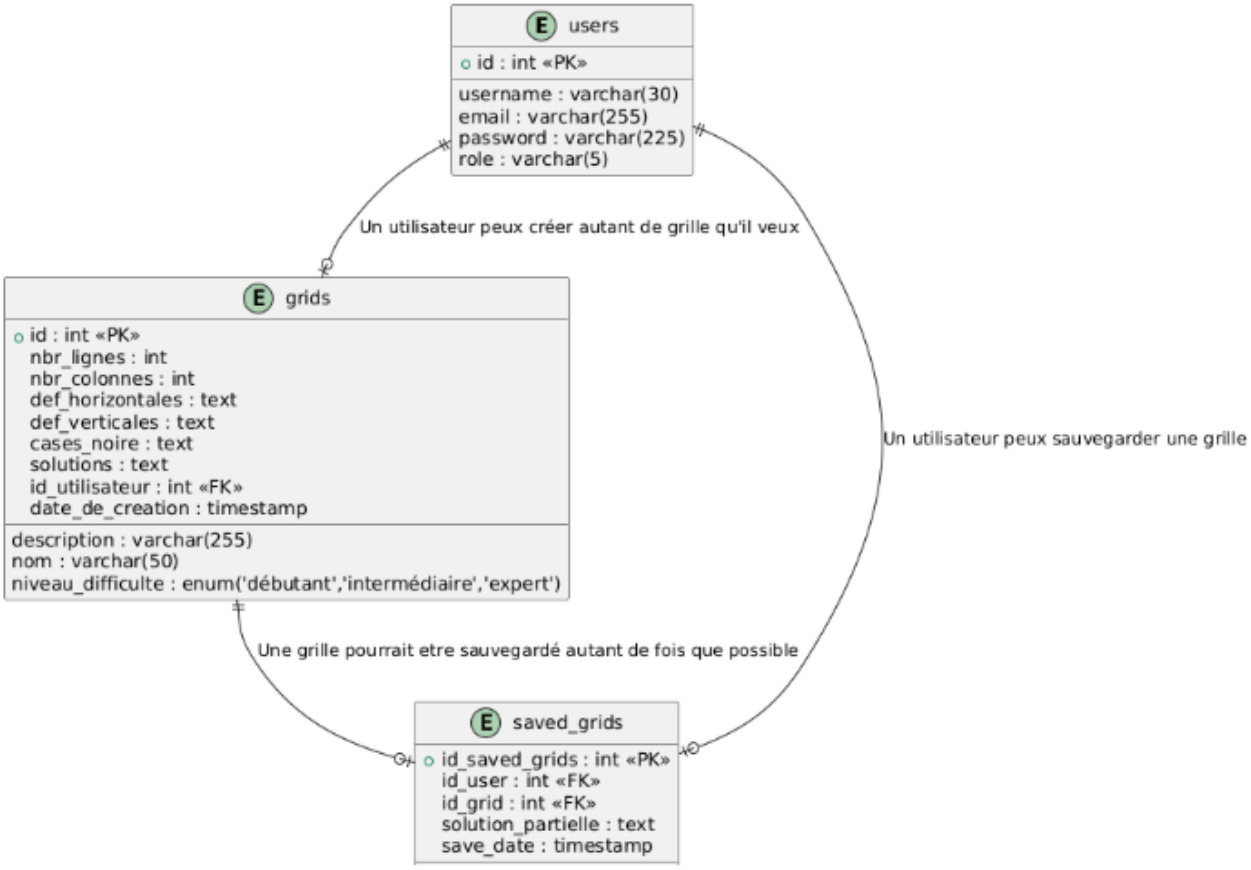
7.4 Supprimer un utilisateur

URL : `http://localhost/projet-d-web/api/deleteUser/2`
Méthode : POST
Réponse :

```
{
    "succes": "Utilisateur supprimé"
}
```

- **Models** : Permettent d'interagir avec la base de données et retournent les données appropriées. Chaque contrôleur fera ensuite appel au modèle pour obtenir les informations nécessaires.

Modèle de données : La figure suivante représente le diagramme d'entités relations.



Nom de la table : users

Description : Cette table contient les informations des utilisateurs inscrits ainsi que celles des administrateurs.

Attributs	Description
`id`	Identifiant unique pour chaque utilisateur (clé primaire).
`username`	Nom d'utilisateur choisi lors de l'inscription, utilisé pour l'identification.
`email`	Adresse email de l'utilisateur, qui doit être unique.
`password`	Mot de passe sécurisé (généralement haché) pour l'authentification.
`role`	Rôle attribué à l'utilisateur : peut être "admin" pour les administrateurs ou "user" pour les utilisateurs standards.

Nom de la table : grids

Description : Cette table contient les informations des grilles créées par les utilisateurs connectés.

Attributs	Description
`id`	Identifiant unique pour chaque grille (clé primaire).
`nbr_lignes`	Nombre de lignes dans la grille.
`nbr_colonnes`	Nombre de colonnes dans la grille.
`def_horizontales`	Définitions des mots horizontaux dans la grille, sous forme de texte ou tableau.
`def_verticales`	Définitions des mots verticaux dans la grille, sous forme de texte ou tableau.
`cases_noire`	Coordonnées des cases noires dans la grille (celles qui ne contiennent pas de lettres).
`solutions`	Réponse complète de la grille sous forme textuelle.
`id_utilisateur`	Identifiant de l'utilisateur ayant créé la grille (clé étrangère vers la table `users`).
`date_de_creation`	Date et heure de la création de la grille.
`description`	Description générale ou objectif de la grille.
`nom`	Nom donné à la grille pour la différencier des autres.
`niveau_difficulté`	Niveau de difficulté de la grille (ex. : débutant, intermédiaire, expert).

Nom de la table : saved_grids

Description : Cette table contient les informations des grilles sauvegardées par les utilisateurs.

Attributs	Description
`id_saved_grids`	Identifiant unique pour chaque grille sauvegardée (clé primaire).
`id_user`	Identifiant de l'utilisateur ayant sauvegardé la grille (clé étrangère vers la table `users`).
`id_grid`	Identifiant de la grille sauvegardée (clé étrangère vers la table `grids`).
`solution_partielle`	Solution partielle ou état actuel de la grille au moment de la sauvegarde.
`save_date`	Date et heure de la sauvegarde.

8 Sécurité

8.1 Sécurité des mots de passes :

Les mots de passes sont hachés avant d'être stockés dans la base de donnée, ce qui garantie la sécurité pour les utilisateurs de notre application.
Note : notre application autorise la création de mots de passe "faibles" sans forcer un nombre minimal de caractères ou de caractères spéciaux. Cela est fait pour faciliter la création des comptes. Pour envisager un déploiement et une mise en production, les contraintes de sécurité devront être adoptées pour garantir des mots de passe forts.

8.2 sécurité contre les attackes

8.2.1 Injection SQL

Utilisation de requêtes préparées ($db->prepare et execute), ce qui protège contre l'injection SQL.

9 Points réussis

Dans ce qui suit, on souhaite mettre en évidence quelques points réussi de notre application, notamment en ce qui concerne la facilité d'ajout de grilles, la résolution des grilles, ainsi que la vérification automatique de leur validité.

9.1 Ajout de grilles

L'ajout de la grille dans notre application a été conçue de telle maniére a ce qu'un utilisateur puisse facilement créer une grille, en remplissant un formulaire simple et intuitive. Des messages d'erreurs appropriés seront affichés a l'utilisateur lui permettant de savoir si la grille qu'il souhaite ajouter est valide.

9.2 Résolution des grilles

Cette fonctionnalité a été conçue d'une manière intuitive, permettant a l'utilisateur de savoir si la solution qu'il a fournit correspond bien a la solution attendue en lui affichant des messages d'erreurs.

10 Conclusion

Ce mini projet nous a permis a enrichir nos compétences en programmation web.
Le processus de développement de ce mini projet a nécessité beaucoup de réflexions notamment sur comment représenter les données et comment controller la grille entrée par l'utilisateur. Ces défis techniques nous ont permis de mettre en pratique tous les concepts vus durant le cours.
En outre, ça nous a permis de découvrir le fonctionnement des API et a quel point ils sont importants.

11 Perspectives d'avenir

À l'avenir Il est crucial d'améliorer la sécurité de l'application pour la protéger contre les attaques, comme les attaques par force brute, en limitant les tentatives de connexion et en utilisant des systèmes de verrouillage temporaires. Parallèlement, améliorer l'interface utilisateur (UI) et la rendre adaptée aux appareils mobiles et tablettes. En termes de fonctionnalités, il serait bien de permettre aux utilisateurs d'accéder à leur profil, d'intégrer un système de messagerie pour favoriser les interactions, d'ajouter la possibilité de s'abonner à d'autres utilisateurs pour suivre leur activité et de permettre de voir les grilles qu'un utilisateur a lui-même créées. En plus de ça il sera bien adopter une architecture basée sur des microservices offrirait une plus grande modularité, une évolutivité simplifiée et faciliterait la maintenance du projet. Il serait également pertinent d'enrichir l'application en intégrant des fonctionnalités d'accessibilité, notamment pour les personnes malvoyantes, afin de la rendre accessible à un public plus large.