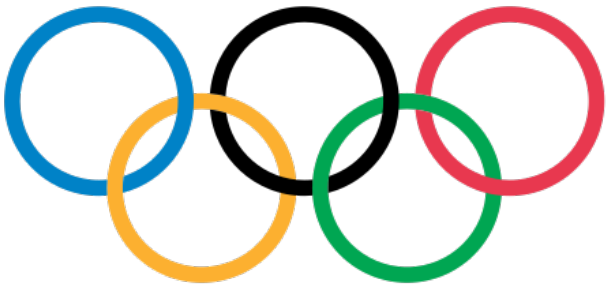
Cahier des charges techniques

PARIS 2024



OSSENI SEMIYOU

Sommaire

1. Contexte du projet

1.1. Présentation du projet

Votre agence web a été sélectionnée par le comité d'organisation des jeux olympiques de Paris 2024 pour développer une application web permettant aux organisateurs, aux médias et aux spectateurs de consulter des informations sur les sports, les calendriers des épreuves et les résultats des JO 2024.

Votre équipe et vous-même avez pour mission de proposer une solution qui répondra à la demande du client.

1.2. Date de rendu du projet

Le projet doit être rendu au plus tard le 22 mars 2024.

2. Besoins fonctionnels

Le site web devra avoir une partie accessible au public et une partie privée permettant de gérer les données.

Les données seront stockées dans une base de données relationnelle pour faciliter la gestion et la mise à jour des informations. Ces données peuvent être gérées directement via le site web à travers un espace administrateur.

3. Ressources matérielles et logiciels nécessaires à la réalisation du projet

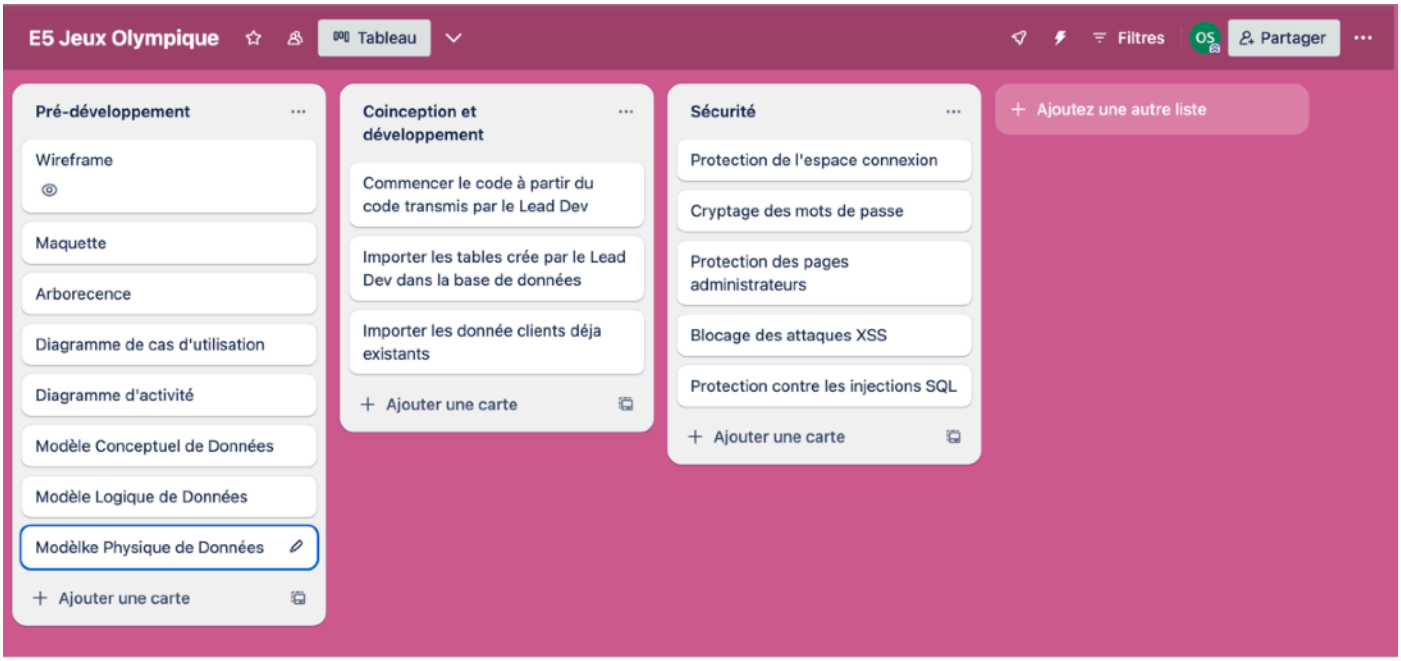
Les ressources matériels utilisé a la réalisation du projet du projet son mon Mac book air 13 pouces 2022

Et les ressources logiciels utilisé sont les logiciels tel que :
- Visual code studio /IDE
- Php My Admin / Gestion de base de donnée
- Figma / Wireframe
- Paradigm/
- Mocodo/ shéama `

REPRENDRE RÉPONSES MISSION 1

4. Gestion du projet

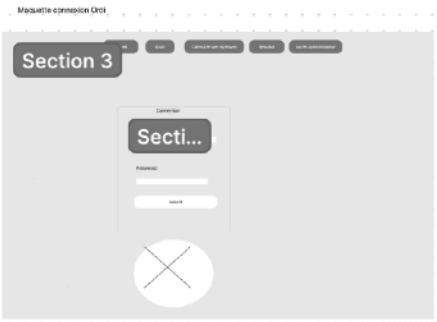
Pour réaliser le projet, nous utiliserons la méthode Agile Kanban. Nous utiliserons également l'outil de gestion de projet en ligne Trello.



5. Conception du projet

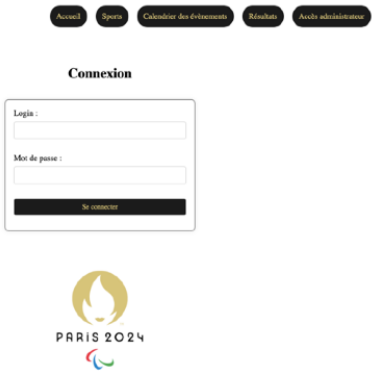
5.1. Le front-end

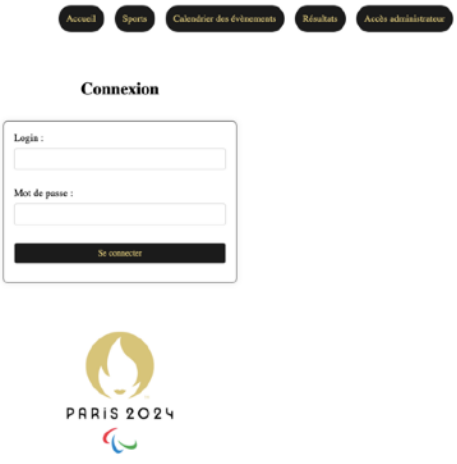
5.1.1. Wireframes



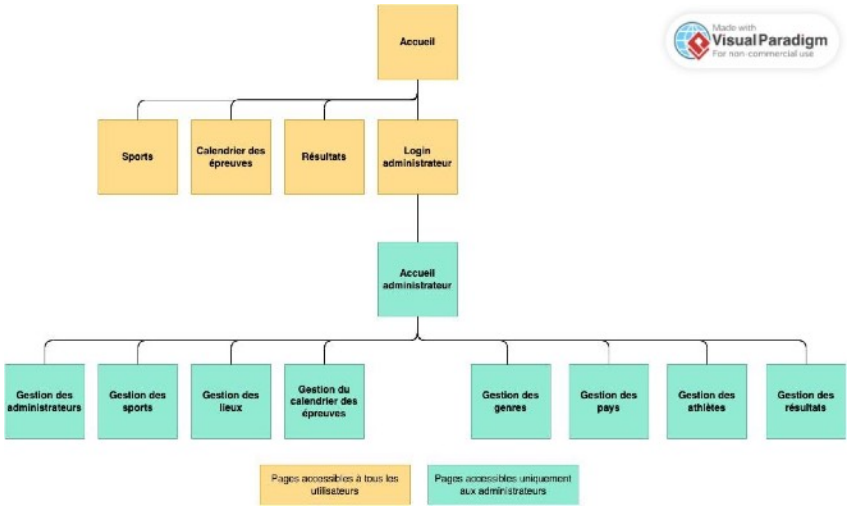
Version responsive

5.1.2. Maquettes



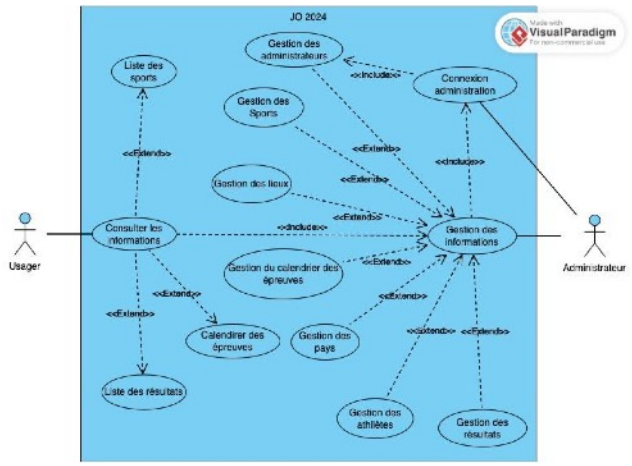


5.1.3. Arborescences

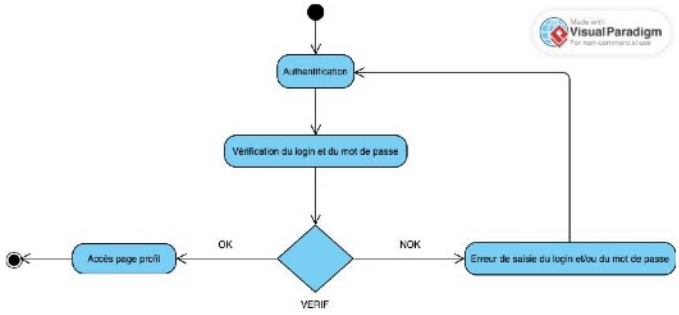


5.2. Le back-end

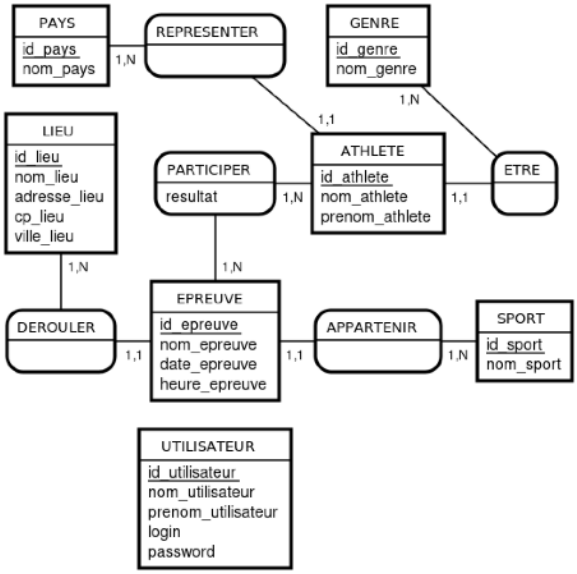
5.2.1. Diagramme de cas d’utilisation



5.2.2. Diagramme d’activités



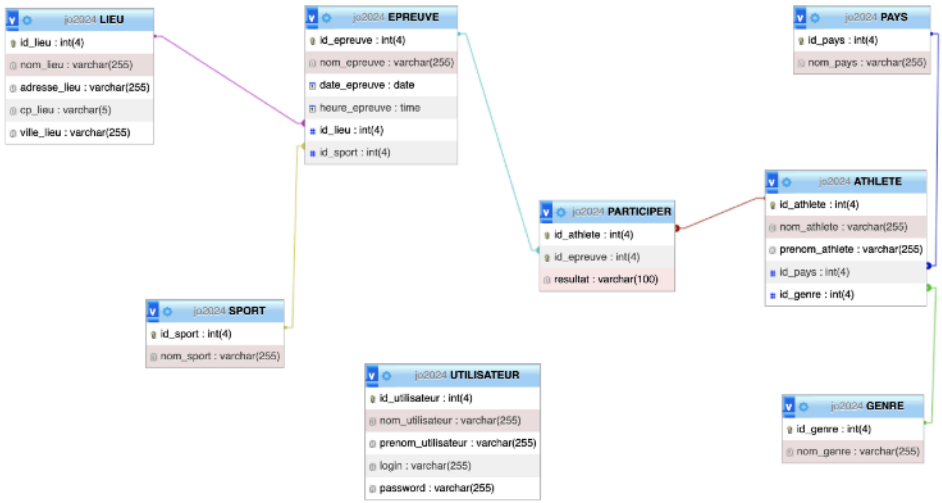
5.2.3. Modèles Conceptuel de Données (MCD)



5.2.4. Modèle Logique de Données (MLD)

- ATHLETE (id_athlete, nom_athlete, prenom_athlete, #id_pays, #id_genre)
- EPREUVE (id_epreuve, nom_epreuve, date_epreuve, heure_epreuve, #id_lieu, #id_sport)
- GENRE (id_genre, nom_genre)
- LIEU (id_lieu, nom_lieu, adresse_lieu, cp_lieu, ville_lieu)
- PARTICIPER (#id_athlete, #id_epreuve, resultat)
- PAYS (id_pays, nom_pays)
- SPORT (id_sport, nom_sport)
- UTILISATEUR (id_utilisateur, nom_utilisateur, prenom_utilisateur, login, password)

5.2.5. Modèle Physique de Données (MPD)



6. Technologies utilisées

6.1. Langages de développement Web

Pour la conception du projet, nous avons l'intention d'employer les langages de développement web suivants, chacun assumant un rôle spécifique dans la création et le fonctionnement des pages web interactives :

HTML5 : Ce langage de balisage est indispensable pour structurer le contenu des pages web, offrant un cadre organisationnel pour les textes, les images, les liens et autres éléments.

CSS3 : En tant que langage de feuilles de style en cascade, il est essentiel pour la présentation visuelle des pages web, permettant de définir le style, la mise en forme, et la disposition des éléments HTML, contribuant ainsi à l'aspect esthétique et à l'expérience utilisateur.

JavaScript : Ce langage de programmation côté client est incontournable pour intégrer des fonctionnalités interactives et dynamiques aux pages web. Il permet la manipulation en temps réel du contenu et des éléments de la page, ainsi que l'interaction avec l’utilisateur.

6.2. Base de données

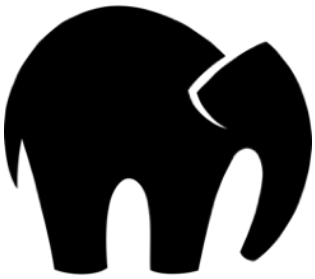
Pour la gestion de la base de données, nous avons prévu d'utiliser les technologies suivantes, combinant à la fois un langage de programmation côté serveur et un langage de requête structuré :

1. PHP8 : Employé comme langage de programmation côté serveur, PHP8 dynamisera les interactions entre la base de données et le front-end de l'application web, facilitant ainsi le traitement des données et la génération de contenu dynamique.

2. SQL : En tant que langage de requête structuré indispensable pour la manipulation des données stockées dans la base de données, SQL sera utilisé pour exécuter des requêtes, des mises à jour et des opérations de gestion de données, assurant ainsi la cohérence et l'intégrité des informations stockées.

3. MAMP : En recourant à MAMP, une plateforme de développement local, nous pourrons créer et gérer efficacement une base de données MySQL. Cette solution offre un environnement de développement complet, intégrant un serveur Apache, une base de données MySQL et le langage de programmation PHP, simplifiant ainsi le processus de développement et de test de notre application web.

4. FileZilla : En tant que client FTP open-source, FileZilla facilitera le transfert et la gestion des fichiers entre un ordinateur local et un serveur distant, offrant également un support SSH pour des transferts sécurisés.



7. Sécurité

7.1. Login

Pour administrer l'authentification des utilisateurs, nous projetons de mettre en place un formulaire de connexion spécialisé qui collectera les informations d'identification, comprenant le nom d'utilisateur et le mot de passe. Ces données seront ensuite confrontées aux informations enregistrées dans la base de données pour confirmer l'existence d'un compte correspondant. Il est essentiel de noter que cette fonctionnalité sera principalement déployée pour l'accès à l'espace administrateur, soulignant ainsi l'importance cruciale de la sécurisation de ce processus.

```
<h1>Connexion</h1>
<form action="../database/auth.php" method="post">
    <label for="login">Login :</label>
    <input type="text" name="login" id="login" required><br><br>
    <label for="password">Mot de passe :</label>
    <input type="password" name="password" id="password" required><br><br>
    <input type="submit" value="Se connecter">
</form>
```

7.2. Cryptage des mots de passe

Pour renforcer la sécurité des mots de passe, nous utilisons la fonction "password_hash" qui génère des hachages sécurisés. Cela se traduit par le cryptage des mots de passe stockés dans la base de données, ajoutant une couche de sécurité supplémentaire. Cette méthode assure que les mots de passe ne sont pas stockés en texte brut, mais plutôt sous forme de hachages irréversibles, renforçant ainsi la protection des informations sensibles des utilisateurs contre d'éventuelles attaques.

7.3. Protection des pages administrateurs

Dans le but de renforcer la sécurité des pages administrateurs, nous implémentons une procédure de déconnexion rigoureuse en utilisant les fonctions "session_unset" et "session_destroy". Cette approche garantit le nettoyage complet et la suppression de toutes les variables liées à la session actuelle dès que l'utilisateur quitte la page. Cette mesure préventive vise à éliminer toute possibilité d'exploitation par des tiers malveillants, en particulier en cas de copie de l'URL de la page connectée et de tentative d'utilisation sur une session déconnectée. En effaçant les données de session immédiatement après la sortie de la page, nous réduisons de manière significative les risques de compromission de la sécurité, préservant ainsi l'intégrité des comptes administrateurs et des données sensibles associées.

7.4. Protection contre les attaques XSS (Cross-Site Scripting)

Pour prévenir les attaques XSS (Cross-Site Scripting), nous avons instauré plusieurs mesures de sécurité. En premier lieu, nous utilisons la fonction "htmlspecialchars" pour filtrer et échapper les caractères spéciaux présents dans les données utilisateur, neutralisant ainsi les tentatives d'injection de code malveillant dans les pages web.

En outre, nous avons adopté l'utilisation de requêtes préparées lors de toute interaction avec la base de données. Les requêtes préparées constituent une approche de programmation sécurisée qui sépare les instructions SQL des données utilisateur, réduisant ainsi le risque d'injection SQL et d'autres formes d'attaques.

En complément, nous maintenons un backlog de sécurité pour détecter et corriger promptement les vulnérabilités potentielles. Ce backlog englobe des audits de sécurité réguliers, des correctifs de sécurité, et des mises à jour système visant à garantir un niveau de sécurité optimal pour nos services.

En appliquant de manière proactive et continue ces mesures de sécurité, nous renforçons la protection de nos applications contre les attaques XSS et autres menaces potentielles, assurant ainsi la sécurité et la confidentialité des données de nos utilisateurs.

7.5. Protection contre les injections SQL

Pour contrer les attaques par injection SQL, nous avons mis en place une stratégie qui intègre l'utilisation de la fonction "htmlspecialchars". Cette fonction revêt une importance cruciale en convertissant les caractères spéciaux présents dans les données utilisateur en entités HTML, réduisant ainsi le risque d'injection de code malveillant dans les requêtes SQL.

En convertissant des caractères spéciaux tels que les guillemets, les apostrophes et les signes inférieurs en équivalents d'entités HTML, nous prévenons toute interprétation erronée de ces caractères par le moteur SQL. Par conséquent, même si des données utilisateur contiennent des caractères potentiellement dangereux, ils seront traités comme du texte ordinaire et ne seront pas interprétés comme des éléments de syntaxe SQL.

Cette approche représente une mesure préventive efficace contre les attaques par injection SQL, renforçant ainsi la sécurité de nos applications et protégeant les données sensibles stockées dans nos bases de données contre toute tentative d'exploitation malveillante.