Hajer Klai

Faïza Goumrhar

RAPPORT DE PROJET

Planning Poker

Année universitaire : 2024

TABLE DES MATIÈRES

I) Contexte du projet..... 3
Phase 1 : Tentative avec PHP.....3
Objectif initial :..... 3
Problèmes rencontrés :..... 3
Leçons apprises :..... 3
Phase 2 : Migration vers JavaScript.....4
Objectif modifié :.....4
Approche choisie :.....4
Description du projet final :.....4
II) Comparaison des technologies utilisées..... 5
III) Résultats obtenus.....5
IV) Recommandations futures..... 6
V) Les difficultés..... 6

I) Contexte du projet

L'objectif initial était de développer un outil pour une fonctionnalité (par exemple, une gestion de backlog ou une estimation de difficulté) en utilisant PHP. Cependant, des problèmes techniques et des ajustements dans les besoins ont conduit à la migration vers une solution basée sur JavaScript. Ce rapport documente les défis rencontrés, les choix technologiques, et les résultats finaux.

Phase 1 : Tentative avec PHP

Objectif initial :

- Développer un outil basé sur PHP pour gérer un backlog et estimer la difficulté des tâches via des votes.

Problèmes rencontrés :

1. **Problèmes de configuration :**
    - Difficultés à configurer un serveur local (XAMPP, WAMP, ou le serveur PHP intégré).
    - Erreurs liées à la version de PHP ou à des bibliothèques manquantes.
2. **Complexité inutile :**
    - PHP nécessitait une gestion des sessions, une interaction serveur plus lourde, et des fichiers de configuration complexes (base de données, etc.), alors que le projet pouvait être géré côté client.
3. **Temps limité :**
    - Le temps nécessaire pour surmonter les obstacles techniques ne semblait pas rentable pour un projet de cette taille.

Leçons apprises :

- **Planification et choix technologique :** Identifier les outils adaptés dès le début.
- **Expérience avec PHP :** Renforcé les connaissances sur la configuration de serveurs et la gestion des fichiers.

Phase 2 : Migration vers JavaScript

Objectif modifié :

- Créer une application web interactive uniquement avec JavaScript,html et css, gérant les fonctionnalités nécessaires côté client.

Approche choisie :

- Utilisation d'un fichier HTML (`index.html`) pour structurer l'interface utilisateur.
- Intégration d'un script JavaScript (`app.js`) pour la logique métier.

Description du projet final :

1. **Fonctionnalités implémentées :**
    - Gestion dynamique des joueurs via un formulaire.
    - Chargement et affichage d'un backlog à partir d'un fichier JSON.
    - Système de votes pour évaluer la difficulté des tâches (avec des règles strictes ou basées sur une moyenne).
    - Génération de fichiers JSON pour sauvegarder les résultats et l'état du jeu.
2. **Avantages de JavaScript :**
    - Simplicité d'exécution via un navigateur, sans besoin de serveur.
    - Gestion entièrement côté client pour des projets interactifs.
    - Rapidité de prototypage et flexibilité.
3. **Code important extrait :**

Création dynamique des champs pour les joueurs :

```
function createPlayerFields() {
    const numPlayers = document.getElementById('numPlayers').value;
    const playerFields = document.getElementById('player-fields');
    playerFields.innerHTML = '';
    for (let i = 0; i < numPlayers; i++) {
        playerFields.innerHTML += `<div>
            <label for="player${i}-name">Pseudo du joueur
${i+1}:</label>
            <input type="text" id="player${i}-name"
name="player${i}-name" required>
        </div>`;
    }
}
```

Système de vote :

```
function submitVotes() {
    const votes = [];
    for (let i = 0; i < numPlayers; i++) {

votes.push(document.getElementById(`player${i}-vote`).value);
    }
    features[currentFeatureIndex].votes = votes;
    validateVotes();
}
```

4. **Design interactif :**
    - Un formulaire HTML et des sections dynamiques (gestion des joueurs, affichage des votes, résultats finaux).
    - Système de gestion des fichiers (chargement/sauvegarde de backlog ou résultats).

II) Comparaison des technologies utilisées

Critères	**PHP**	**JavaScript**
Configuration	Serveur requis (ex: XAMPP)	Directement dans le navigateur
Performance	Côté serveur, plus robuste	Côté client, rapide pour les petits projets
Courbe d'apprentissage	Nécessite une installation préalable	Accessible à tous les niveaux
Flexibilité	Plus adapté aux projets avec une base de données	Idéal pour des applications interactives autonomes

III) Résultats obtenus

- Un outil fonctionnel, interactif, et facile à utiliser via un navigateur web.
- Documentation claire et générée automatiquement pour les résultats des votes et le backlog.
- Temps optimisé en évitant des complexités inutiles liées au serveur.

IV) Recommandations futures

1. **Choix technologique :** Toujours analyser les besoins avant de choisir une stack.
2. **Meilleure maîtrise de PHP :** Consacrer du temps à comprendre la configuration d'un environnement PHP.
3. **Évolutivité du projet :** Ajouter des fonctionnalités avancées comme une gestion de base de données si nécessaire (MySQL ou MongoDB).

Règles de vote :

- **Unanimité** : Dans ce mode de jeu, tous les joueurs doivent être d'accord sur une décision pour qu'elle soit validée. Nous choisissons cette règle lorsque nous voulons que les décisions soient prises de manière stricte et collective, exigeant l'accord total des participants.
- **Moyenne** : Dans ce cas, les résultats des votes sont agrégés pour déterminer une décision basée sur la moyenne des votes. Ce système est plus flexible et démocratique, permettant à la majorité de décider même si tous les joueurs ne sont pas d'accord à 100%.

Interface et UX (utilisateur) :

Nous avons conçu une interface simple et fonctionnelle qui permet aux utilisateurs d'interagir facilement avec le jeu. Après avoir importé le fichier JSON et configuré les paramètres de jeu (comme le mode de vote), les utilisateurs peuvent entrer leurs choix de vote, puis soumettre les résultats. Cette interface est pensée pour rendre l'expérience fluide et agréable.

Réglages supplémentaires et personnalisation :

Nous avons prévu une certaine personnalisation des paramètres du jeu, permettant de modifier les règles de vote et la manière dont les résultats sont affichés. Cette flexibilité nous permet de nous adapter à différents types de jeux ou de situations, selon ce que nous souhaitons tester ou explorer.

Structure du code :

Notre code est organisé de manière claire et efficace : le **HTML** est utilisé pour l'affichage de l'interface, le **CSS** gère le style, et le **JavaScript** se charge de la logique et des interactions, telles que l'importation du fichier JSON et la gestion des votes. Nous avons également veillé à l'expérience utilisateur, en nous assurant que l'application soit intuitive et réactive.

Objectifs et contexte du projet :

Le but de notre application est de proposer une interface ludique et interactive, principalement pour la gestion de votes. Elle peut être utilisée dans des contextes variés, comme des jeux en groupe, des simulations, ou même dans un cadre professionnel pour aider à prendre des décisions en équipe. L'objectif est de rendre l'expérience de vote plus

fluide et accessible tout en permettant une certaine personnalisation du processus décisionnel.

V) Les difficultés

Lors de la réalisation de mon projet, j'ai rencontré plusieurs défis techniques, notamment lors de l'exécution de mes tests Jest et de l'intégration continue avec GitHub Actions. J'ai documenté le processus de résolution des problèmes afin de tirer des leçons et améliorer les pratiques futures.

Déroulement :

Installation de Jest et `jest-environment-jsdom` :

Problème : Mes tests n'étaient pas reconnus par Jest, ce qui empêchait leur exécution.
Solution : J'ai installé `jest-environment-jsdom` en utilisant la commande suivante :

npm install --save-dev jest-environment-jsdom

Cette installation a permis à Jest de gérer correctement l'environnement de test.

Définition des Tests dans `app.test.js` :

Problème : Les tests créés n'étaient pas détectés par Jest.
Solution : J'ai ajouté des tests unitaires simples et précis pour valider les principales fonctions de l'application. Voici un exemple de test de base :
javascript

```
test('basic addition', () => {
  expect(1 + 1).toBe(2);
});
```

Configuration de GitHub Actions :

Problème : J'avais des difficultés à configurer un workflow GitHub Actions pour automatiser les tests.

Solution : Malgré plusieurs tentatives de configuration d'un fichier YAML (`test.yml`) pour automatiser l'exécution des tests, je n'ai pas réussi à intégrer GitHub Actions de manière efficace. Je n'ai pas pu résoudre les erreurs persistantes lors de l'exécution des workflows.

Gestion des Conflits Git :

Problème : J'ai rencontré des conflits lors des opérations de fusion et de rebase des branches.
Solution : J'ai utilisé les commandes Git appropriées pour résoudre les conflits et synchroniser les modifications locales avec le dépôt distant :

```
git stash
git pull --rebase origin main
git stash pop
git add .
git rebase --continue
```

Problèmes avec `.DS_Store` :
Problème : Le fichier `.DS_Store` causait des conflits fréquents lors des opérations Git.
Solution : J'ai ajouté `.DS_Store` au fichier `.gitignore` et supprimé le fichier du cache Git

```
echo ".DS_Store" >> .gitignore
git rm --cached .DS_Store
git add .gitignore
git commit -m "Ajout de .DS_Store à .gitignore et suppression du cache"
```

Ce que je retiens de nos efforts :

Malgré les efforts déployés, je n'ai pas réussi à intégrer GitHub Actions pour automatiser les tests. Toutefois, j'ai appris à gérer les conflits Git et à configurer des tests unitaires efficaces avec Jest. Ces expériences m'ont permis de mieux comprendre les outils et les pratiques de développement collaboratif.

Conclusion :

En conclusion, notre projet CAPI représente une solution interactive et flexible pour la gestion de votes dans des contextes variés, que ce soit pour des jeux, des simulations ou des décisions en groupe.
Grâce à l'intégration d'un fichier JSON, nous avons pu centraliser et organiser les données des joueurs de manière efficace, tout en offrant une interface simple et intuitive. Les règles de vote, basées sur l'unanimité ou la moyenne, permettent d'adapter le processus décisionnel selon les besoins spécifiques de chaque situation.

Ce projet illustre notre capacité à combiner la gestion des données, la logique de jeu et l'expérience utilisateur pour créer une application fonctionnelle et agréable à utiliser. En somme, ce projet a non seulement répondu à nos objectifs initiaux, mais a également ouvert des pistes intéressantes pour de futures améliorations et personnalisations.