

# PROJET CAPI - PLANNING POKER



## Planning Poker

**CONCEPTION AGILE DE PROJETS INFORMATIQUES**

**MASTER 1 INFORMATIQUE UNIVERSITÉ LUMIÈRE LYON 2**

**Maîtres d'oeuvres:**

Lansana CISSE

Samy Chennoufi

Université Lumière Lyon 2

Institut de communication

**Maître d'ouvrage:** Valentin Lachand-Pascal

 **INSTITUT  
de la  
communication**

Année Universitaire 2023-2024

## **SOMMAIRE**

- 1 Introduction
- 2 Développement du Projet
  - 2.1 Structure du projet – Architecture
  - 2.2 Justifications des Patterns
  - 2.3 Justification des Choix Techniques
  - 2.4 Mise en Place de l'Intégration Continue
  - 2.5 Tests Unitaires
  - 2.6 Génération de Documentation
- 3 Problèmes reconcentrés et perspectives
- 4 Conclusion

# **1. INTRODUCTION**

Dans le monde dynamique des méthodes agiles, le planning poker se distingue comme une technique de planification clé, utilisée pour évaluer la complexité des tâches dans les projets de développement logiciel. Cette méthode, à la fois collaborative et stratégique, implique plusieurs participants qui utilisent des cartes avec des valeurs prédéfinies pour estimer la charge de travail associée à chaque tâche. L'essence du planning poker réside dans sa capacité à faciliter une estimation collective et équilibrée, en intégrant diverses perspectives et expertises.

Dans ce présent rapport, nous nous proposons d'explorer les possibilités d'intégration de la méthode moyenne et de la méthode classique du planning poker dans un site web.

Le scrum master, qui est responsable de la définition des tâches et de la collecte des coordonnées des participants à la réunion, organise le planning poker en local.

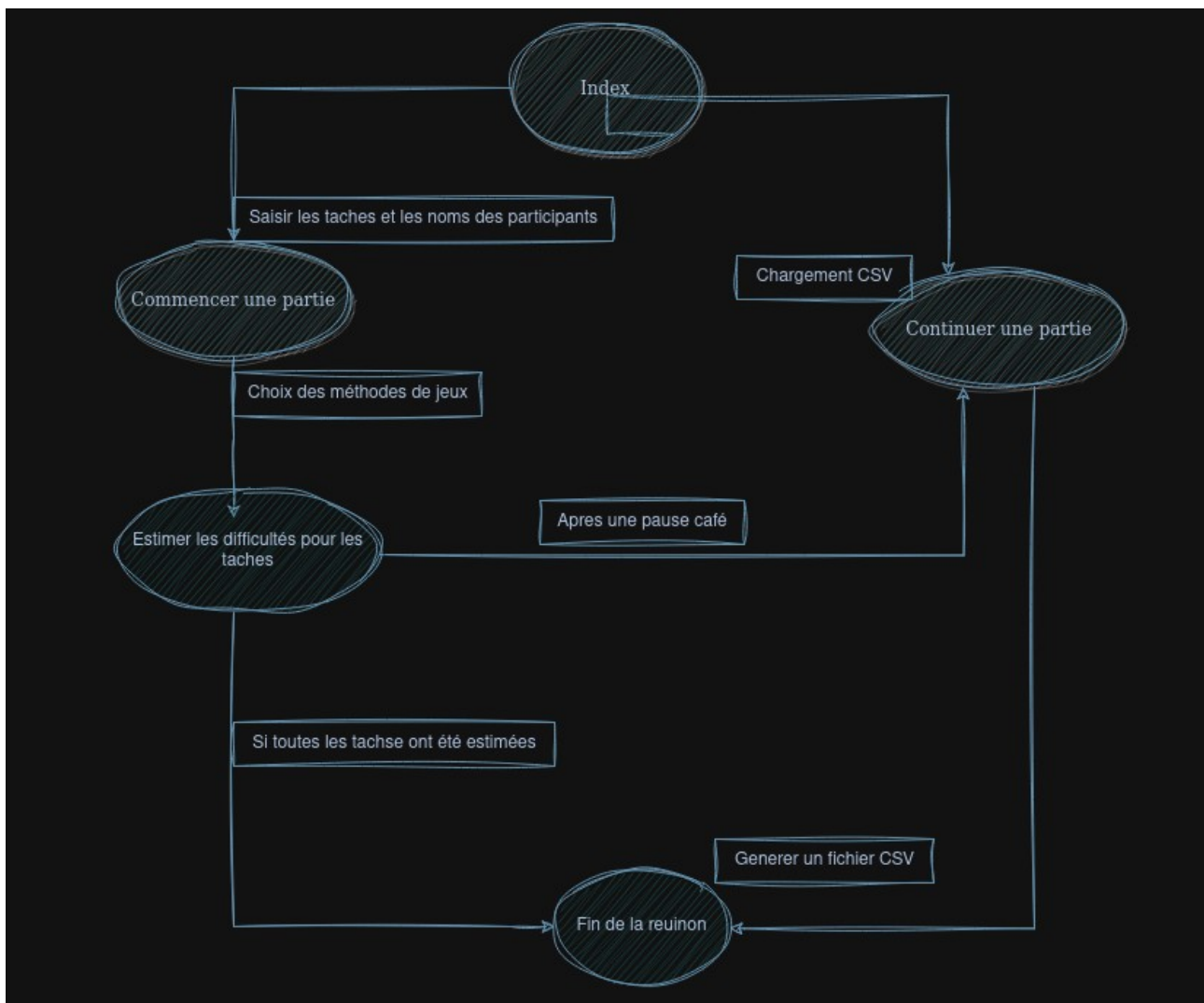
## 2. DÉVELOPPEMENT DU PROJET

### 2.1 Structure du projet – Architecture

Le projet CAPI Planning Poker utilise un modèle MVC classique pour séparer les données, l'affichage et le traitement des requêtes. Les données sont stockées dans des fichiers CSV, ce qui permet de réduire la dépendance à une base de données.

Le *front-end* est développé avec HTML, CSS et JavaScript. HTML est utilisé pour la structure de la page, CSS pour le style de la page et JavaScript pour la logique de la page. Bootstrap est utilisé pour simplifier le développement et la maintenance du front-end.

Le *back-end* est développé avec PHP. PHP est utilisé pour la gestion des données et la logique de l'application.



Img1 :Architecture du site

## 2.2 Justification des patterns

Dans ce projet nous avons utilisé trois patterns tout au long du développement dans les différents fichiers.

- **Module Pattern** : Utilisé dans : `tache.js`

Le Module Pattern permet d'encapsuler des méthodes et des variables privées, évitant ainsi les conflits dans l'espace de noms global. Cela est particulièrement utile dans `tache.js`, où les fonctions comme `recupererLesDonnees` sont encapsulées dans le module, réduisant les risques de collision avec d'autres scripts.

Ce pattern aide à organiser le code de manière logique, regroupant les fonctionnalités apparentées. Dans `tache.js`, cela permet de séparer clairement les fonctionnalités comme la récupération de données et leur affichage.

- **Singleton** : Utilisé dans : `difficultes.js`

Le Singleton assure qu'une classe n'a qu'une seule instance et fournit un point d'accès global à cette instance. Dans `difficultes.js`, des variables globales comme `votes` et `data` suggèrent un besoin de maintenir un état unique accessible dans tout le script.

Ce pattern est idéal pour gérer les ressources partagées. Dans le cas de `difficultes.js`, les données partagées à travers l'application peuvent être gérées de manière efficace.

- **Façade** : Utilisé dans : `continuer_partie.js`

Simplification d'Interface : Le pattern Facade offre une interface simple pour un sous-système complexe. `continuer_partie.js` simplifie l'interaction avec le système de fichiers en masquant la complexité des opérations de fichier. Il réduit la complexité du code pour les utilisateurs de l'API, permettant une utilisation plus facile et moins sujette aux erreurs.

## 2.3 Justification des Choix Techniques



*Langage de Programmation:* HTML, CSS, Bootstrap, JavaScript, PHP

- **HTML (HyperText Markup Language)**

HTML est le langage de balisage standard pour créer des pages web. Il structure le contenu et est essentiel pour toute application web. HTML est soutenu par tous les navigateurs web, garantissant la compatibilité et l'accessibilité.

- **CSS (Cascading Style Sheets)** : CSS est utilisé pour styliser et mettre en forme le contenu HTML. Il permet de créer des designs riches et engageants. En combinaison avec des frameworks comme Bootstrap, facilite la création de designs responsives qui s'adaptent aux différentes tailles d'écran.
- **Bootstrap** : Un framework front-end qui permet un développement rapide avec des templates prêts à l'emploi. Il offre une cohérence dans le design tout en étant adaptatif aux différents appareils, améliorant ainsi l'expérience utilisateur sur diverses plateformes.

- JavaScript :JavaScript permet de créer des interfaces dynamiques. Il enrichit les pages web avec des fonctionnalités interactives comme les animations et les formulaires dynamiques. JavaScript joue un rôle clé dans le développement full-stack, étant utilisable tant côté client que côté serveur (Node.js).
- PHP : largement reconnu pour sa robustesse dans la construction d'applications web complexes. Il gère efficacement les interactions côté serveur, y compris l'accès aux bases de données et la logique métier.

## ➔ *Structure des Classes et Modules - Modules et Fonctions en JavaScript.*

La structure modulaire permet de diviser le code en parties plus petites et réutilisables, facilitant ainsi la maintenance et le développement. L'encapsulation aide à protéger l'état interne et à prévenir les interférences entre différentes parties du code.

## ➔ *Gestion de l'État et des Données - Json et des sessions PHP*

Vu que le jeu se déroule en local nous avons jugé nécessaire d'utiliser des fichiers json plutôt que de faire appelle à une base de données relationnelles. Dans les applications interactives, maintenir l'état est crucial pour une expérience utilisateur cohérente. Le stockage de données côté client (JavaScript) permet une réponse rapide aux interactions utilisateur, tandis que le côté serveur (PHP) assure la sécurité et la gestion des données persistantes.

## ➔ *Choix de Dépôt de Travail : GitHub*

Nous avons choisi GitHub comme notre dépôt de travail principal en raison de ses capacités exceptionnelles en termes de collaboration, gestion de version et intégration continue. Cette plateforme renforce notre collaboration grâce à des outils tels que les branches, les pull-requests et un système de revue de code complet, facilitant l'échange d'idées et la contribution collective au projet.

En tant que plateforme basée sur Git, GitHub nous offre une gestion robuste des versions. Cette fonctionnalité est cruciale pour nous, car elle permet un suivi efficace et sécurisé de toutes les modifications apportées au code, garantissant ainsi que chaque étape de notre travail est bien documentée et réversible si nécessaire.

## 2. 5 Mise en Place de l'Intégration Continue

Nous avons utilisé GitHub Actions pour la mise en place de l'intégration continue tout au long du projet. L'adoption de cette technologie a été cruciale pour maintenir l'intégrité et la qualité du code, surtout dans un contexte où les modifications étaient fréquentes et provenaient de deux contributeurs



Code	Blame	29 lines (25 loc) • 556 Bytes
------	-------	-------------------------------

```
1  name: Integration Continue
2
3  on:
4    push:
5      branches:
6        - main
7    pull_request:
8      branches:
9        - main
10
11  jobs:
12    build:
13      runs-on: ubuntu-latest
14
15      strategy:
16        matrix:
17          node-version:
18            - 18.x
19
20      steps:
21        - uses: actions/checkout@v3
22        - name: Use Node.js ${{ matrix.node-version }}
23          uses: actions/setup-node@v3
24          with:
25            node-version: ${{ matrix.node-version }}
26        - run: npm install
27        - run: npm install jest-environment-jsdom
28        - run: npm run build --if-present
29        - run: npm test
```

*img2 : Fichier d'integration continue yml*

## 2.6 Tests Unitaires

Nous avons mis en œuvre une série de tests unitaires en utilisant les bibliothèques Jest, Babel, et ts-jest de Node.js. L'intégration de ces tests a été réalisée sur GitHub, en exploitant à la fois le fichier package.json et divers fichiers de test spécifiques à chaque composant du programme. Cette approche nous a permis de garantir une couverture de test cohérente et efficace pour l'ensemble de notre code.

 lansacisse package.json ✓ 73a2d6b · 1 hour ago  History

Code Blame 24 lines (24 loc) · 630 Bytes Raw Copy Download Edit

```
1  {
2    "name": "ApplicationPlanningPoker",
3    "version": "1.0.0",
4    "description": "Le Planning Poker permet une estimation collective et rapide des tâches à réaliser dans un projet en utilisant le consensus de l'équipe",
5    "main": "accueil.js",
6    "scripts": {
7      "test": "jest --config jest.config.js"
8    },
9    "jest": {
10     "testEnvironment": "jsdom"
11   },
12   "author": "",
13   "license": "MIT",
14   "devDependencies": {
15     "jest": "^29.7.0",
16     "jest-environment-jsdom": "^29.7.0"
17   },
18   "dependencies": {
19     "@babel/preset-env": "^7.23.5",
20     "babel-jest": "^29.7.0",
21     "import-local": "^3.1.0",
22     "ts-jest": "^29.1.1"
23   }
24 }
```

Attention, il ne vous reste que 1 minute 37 secondes avant la déconnexion. Pensez à sauvegarder vos fichiers!

Img3 : Fichier packag.json



11 workflow runs			Event ▾	Status ▾	Branch ▾	Actor ▾
●	Fichier de testJSON	main	Integration Continue #40: Commit e94cb21 pushed by lansanacisse	now Queued		...
●	Fichier de testJSON	main	Generate JSDoc #13: Commit e94cb21 pushed by lansanacisse	now Queued		...
✓	package.json	main	Generate JSDoc #12: Commit 73a2d6b pushed by lansanacisse	2 hours ago 22s		...

img4 : Test dynamique apres une mise à jours

## 2.4 Documentation du code

Pour la mise en place de la documentation, nous avons opté pour GitHub Actions, une fonctionnalité native de GitHub, en élaborant un fichier YAML situé dans le répertoire .github/workflows. Ce fichier YAML joue un rôle essentiel dans la définition de notre flux de travail d'intégration continue, permettant l'automatisation de plusieurs aspects de notre projet, y compris la réalisation des tests et la production automatique de la documentation du code.

Code

Blame

34 lines (28 loc) • 792 Bytes

```

1  name: Generate JSDoc
2
3  on:
4    push:
5      branches: [ main ]
6    pull_request:
7      branches: [ main ]
8  jobs:
9    build:
10     runs-on: ubuntu-latest
11
12     steps:
13       - uses: actions/checkout@v2
14         # Vérifie le code source du dépôt
15
16       - name: Setup Node.js
17         uses: actions/setup-node@v2
18         with:
19           node-version: '14'
20           # Remplacez '14' par la version de Node.js que vous utilisez
21
22       - name: Install JSDoc
23         run: npm install -g jsdoc
24         # Installe JSDoc
25
26       - name: Generate Documentation
27         run: jsdoc -c jsdoc.json
28         # Génère la documentation en utilisant votre fichier de configuration JSD
29
30       - name: Deploy Documentation
31         uses: peaceiris/actions-gh-pages@v3
32         with:
33           github_token: ${ secrets.GITHUB_TOKEN }
34           publish_dir: ./out

```

Img5 : Fichier yml pour la documentation dynamique

All workflows

Showing runs from all workflows

Q

Filter workflow runs

9 workflow runs				Event ▾	Status ▾	Branch ▾	Actor ▾
✔ package.json	Generate JSDoc #12: Commit <a href="#">73a2d6b</a> pushed by lansanacisse			main	1 hour ago 22s	...	
✔ package-lock.json	Generate JSDoc #11: Commit <a href="#">4620e87</a> pushed by lansanacisse			main	1 hour ago 20s	...	
✔ ci.yml	Generate JSDoc #10: Commit <a href="#">f441f02</a> pushed by lansanacisse			main	1 hour ago 23s	...	
✔ ci.yml	Generate JSDoc #9: Commit <a href="#">396dfe9</a> pushed by lansanacisse			main	2 hours ago 20s	...	
✔ Ajout de jest-environment-jsdom et mise à jour de la configuration Jest	Generate JSDoc #8: Commit <a href="#">895839b</a> pushed by lansanacisse			main	2 hours ago 22s	...	
✔ README.md	Generate JSDoc #7: Commit <a href="#">4b395ae</a> pushed by lansanacisse			main	2 hours ago 22s	...	
✔ Documentation	Generate JSDoc #6: Commit <a href="#">50c2027</a> pushed by lansanacisse			main	2 hours ago 23s	...	
✔ Documentation					2 hours ago		

img6 : Validation de la documentation github action

ApplicationPlanningPoker / Documentation /

lansanacisse Documentation ✔

50c2027 · 18 hours ago History

Name	Last commit message	Last commit date
..		
fonts	Documentation	18 hours ago
scripts	Documentation	18 hours ago
styles	Documentation	18 hours ago
index.html	Documentation	18 hours ago

img7 : Documentation

### **3. DIFFICULTÉS ET PERSPECTIVES**

Au cours du développement, nous avons rencontré plusieurs défis, tout en identifiant des pistes d'amélioration, particulièrement dans les tests d'intégration. Par ailleurs, la gestion des mises à jour des fichiers sur GitHub a généré des conflits. Cette situation a mis en évidence le besoin d'une gestion plus rigoureuse et d'une coordination accrue pour prévenir les incohérences et garantir l'intégrité du code.

Malgré ces obstacles, notre projet a tiré profit de ces expériences. Chaque défi est devenu une opportunité d'apprentissage, nous incitant à peaufiner nos méthodes et à consolider notre application. Envisageant d'introduire des fonctionnalités interactives comme le chat et une base de données plus performante, nous ambitionnons de rendre le Planning Poker en ligne plus attractif et collaboratif. Ceci améliorera la communication et le sentiment d'unité au sein des équipes, même à distance. Ces évolutions préparent le terrain pour une application plus robuste et mieux adaptée aux exigences des équipes modernes, constituant ainsi une avancée significative dans l'évolution de notre projet.

## 4. CONCLUSION

À l'issue de notre projet de Planning Poker, nous pouvons avec fierté faire le bilan d'une aventure jalonnée de succès et d'enrichissements professionnels. L'intégration harmonieuse de diverses technologies, notamment HTML, CSS, Bootstrap, JavaScript et PHP, a été cruciale pour atteindre nos objectifs. Elle nous a permis de développer une application non seulement répondant aux exigences initiales mais se distinguant également par son innovation et la qualité de son expérience utilisateur. L'emploi stratégique de GitHub a été un autre facteur clé, favorisant une collaboration efficace et une gestion de projet optimale, mettant en lumière l'importance d'une sélection adéquate des outils pour garantir la qualité de notre travail.

Ce projet Planning Poker, en plus d'atteindre ses objectifs spécifiques, a renforcé notre cohésion d'équipe et affiné nos compétences techniques. Les défis rencontrés, loin de nous freiner, ont enrichi notre expérience et élargi notre compréhension des meilleures pratiques en développement logiciel. La réussite de ce projet témoigne de notre capacité à travailler ensemble de manière productive, à résoudre des problèmes complexes et à créer des solutions qui allient fonctionnalité et innovation. Cette expérience a consolidé notre approche collective en matière de résolution de problèmes et a démontré notre capacité à réaliser des projets ambitieux avec compétence et créativité.