

# État du Sprint 3 Réalisations vs Objectifs

## Objectifs du Sprint 3 :

- **Finaliser l'UI complète** : GameView, LeaderboardView, SettingsView, intégration des boutons et navigation.
- **Implémenter la logique de jeu avancée** : modes Solo/Duel, indices, sélection de genres, timer, saisie des réponses.
- **Intégrer audio et persistance dans l'UI** : lecture réelle, sauvegarde automatique, chargement playlists.
- **Réaliser des tests complets** : unité + manuel, Javadoc complète.
- **Documentation finale** : README, manuel utilisateur, diagrammes mis à jour.

## Réalisations :

### UI Complète

- **MainMenu** : Boutons pour Mode Solo, Mode Duel, Classement, Paramètres, Quitter tous fonctionnels.
- **GameView** : Interface de jeu complète avec timer, champs de saisie titre/artiste, feedback visuel, affichage du score en temps réel.
- **LeaderboardView** : Affichage du classement des scores sauvegardés avec tri et formatage.
- **SettingsView** : Paramètres complets (nombre de manches, durée d'extrait, volume, activation des indices, bonus de rapidité, sélection de genre musical).

### Logique de Jeu

- **Mode Solo** : Implémentation complète avec scoring détaillé (points pour titre/artiste corrects, bonus de rapidité selon le temps de réponse).
- **Mode Duel** : Implémentation complète avec gestion alternée des tours, scores séparés, et affichage du gagnant en fin de partie.
- **Timer fonctionnel** : Compte à rebours avec auto-soumission des réponses à expiration.
- **Gestion des rounds** : Transition fluide entre les manches, affichage des scores intermédiaires, écran de fin de partie.
- **Système d'indices** : Activation et utilisation des indices (première lettre) avec pénalité de points.

### Intégration Audio et Persistance

- **AudioService** : Lecture réelle des extraits via API iTunes, fallback vers playlist locale en cas d'échec, gestion du volume.
- **GameController** : Génération automatique de playlists depuis l'API iTunes par genre, chargement automatique des extraits, gestion des erreurs réseau.

- **Persistence complète** : Sauvegarde automatique des scores (leaderboard), paramètres utilisateur, et playlists au format JSON.

## Tests

- **Tests unitaires** : Présents pour AudioService, GameController, ScoreService, PersistenceService, Player, Track.
- **Tests manuels** : Validation complète de l'UI et des interactions utilisateur.
- **Javadoc** : Documentation générée et complète pour toutes les classes principales.

## Documentation

- **README** : Présent avec instructions d'installation et d'utilisation.
- **Diagrammes** : Diagrammes de classes et d'architecture présents

## Répartition des 100 points :

- **Iskander** : 33 points — Intégration audio complète (API iTunes, fallback, gestion volume), sélection de genres, tests AudioService.
- **Léo** : 34 points — Persistence complète (scores, paramètres, playlists), système de leaderboard, tests PersistenceService et ScoreService.
- **Achraf** : 33 points — UI complète (GameView, LeaderboardView, SettingsView, MainMenu), logique de jeu (modes Solo/Duel, timer, scoring, indices), tests manuels UI.

## Écarts :

- **Documentation** : README et diagrammes présents, mais mise à jour pour le Sprint 3 non réalisée.
- **Tests d'intégration** : Tests unitaires solides, mais tests d'intégration end-to-end limités (interaction complète audio + UI + persistance).
- **Gestion d'erreurs** : Fallback audio implémenté, mais gestion d'erreurs réseau perfectible (messages utilisateur, retry automatique).

## Conclusion Sprint 3 :

Environ 95 % réalisé.

L'application est complète et pleinement jouable avec toutes les fonctionnalités principales implémentées (modes Solo/Duel, audio réel, scoring avancé, indices, persistance). L'UI est finalisée et fonctionnelle. Les écarts restants concernent principalement le polish (animations, effets visuels), la robustesse (gestion avancée des erreurs), et la documentation (mise à jour des diagrammes).

---

## Plan pour le Sprint 4

### Objectifs :

- **Polish UI/UX** : Animations de transition, effets visuels/sonores, feedback amélioré.
- **Robustesse et gestion d'erreurs** : Gestion complète des erreurs réseau, validation des inputs, messages clairs.
- **Amélioration du classement** : Filtres (mode, genre), statistiques détaillées, export de données.
- **Tests et documentation finale** : Couverture de tests étendue, mise à jour complète de la documentation.

## Répartition des tâches détaillée :

### Iskander :

- Optimisation de l'AudioService : gestion avancée des erreurs réseau, retry automatique.
- Implémentation du cache pour les requêtes API fréquentes.
- Ajout d'effets sonores pour les événements de jeu (bonne/mauvaise réponse, fin de manche).
- Tests unitaires supplémentaires pour la gestion d'erreurs audio.

### Léo :

- Amélioration du Leaderboard : filtres par mode (Solo/Duel), statistiques (taux de réussite, moyenne).
- Export des données du classement (CSV ou JSON).
- Validation robuste des inputs utilisateur (normalisation, trim, caractères spéciaux).
- Tests unitaires pour les nouvelles fonctionnalités de persistance.

### Achraf :

- Polish de l'UI : animations de transition entre vues (fade, slide), effets visuels sur réponses.
- Animations pour le bonus de rapidité (jauge dynamique, couleurs).
- Écrans de transition entre manches avec récapitulatif.
- Tests manuels complets de l'UI avec gestion d'erreurs.
- Mise à jour de la documentation : README, diagrammes UML à jour.

## Livrable attendu :

Application BlindTest complète et polie :

- Modes Solo/Duel fluides avec animations et effets.
- Gestion d'erreurs robuste avec messages clairs.
- Leaderboard enrichi avec filtres et statistiques.
- Documentation complète et diagrammes à jour.
- Couverture de tests étendue (>80%).

## Risques :

- **Animations JavaFX** : Complexité d'apprentissage
- **Performance** : Optimisation du cache et du chargement audio à tester sur différentes machines.

- **Temps limité** : Prioriser le polish critique (gestion erreurs, tests) sur le polish esthétique si nécessaire.
- Charge équilibrée via revues hebdomadaires et communication régulière.

## **Lien GitHub :**

<https://github.com/illeska/blindtest>