Stock Merch

Projet Master

Thomas Siest

M2 CYBER 2024-2025

Sommaire

Besoins ciblé et/ou problématique à traiter 3
Motiver l'idée du projet 3
Se positionner par rapport aux projets similaires existants 4
Méthodologie et Organisation du travail 5
Cahier des charges 6
Projet Stock Merch 6
Présentation générale du projet 8
Objectifs du projet 8
Périmètre fonctionnel 8
Fonctionnalités principales (MVP) 8
Fonctionnalités secondaires (hors MVP – à implémenter plus tard) 8
Contraintes techniques 9
Contraintes de développement 9
Méthodologie de travail 9
Planning très prévisionnel 10
Livrables attendus 10
Conception de la solution 11
Réalisation / Mise en œuvre 14
Tests / Validation / Vérification 15

Besoins ciblé et/ou problématique à traiter

En ma qualité de trésorier d'une association régie par la loi 1901, consacrée à la valorisation des licences d'animes musicaux (telles que Love Live, BanG Dream, Idolm@ster...), je fais face à une nécessité constante et cruciale : l'administration de l'inventaire de produits dérivés (ou « merch ») lors d'événements tels que les conventions. Juste pour vous informer, cette organisation porte le nom de United Glowstick.

À l'heure actuelle, cette gestion est souvent réalisée manuellement ou à l'aide d'outils non appropriés (Excel, notes manuscrites), ce qui entraîne confusion, erreurs d'inventaire et des problèmes pour déterminer les prix ou les quantités restantes. De plus, en pleine convention, il faut déjà gérer le stand, les animations, le personnel, ce qui rend la gestion du merch encore plus complexe.

Il devient alors difficile de répondre à des questions simples :

- Combien avons-nous de tels goodies ?
- À quel prix devons-nous le vendre ?
- Combien avons-nous vendu ?
- Sont-ils concernés par une enchère ?

Ces limites freinent la fluidité de notre activité et peuvent avoir un impact direct sur notre rentabilité et notre organisation.

Motiver l'idée du projet

Ce projet répond à un double enjeu, et donc à une double opportunité : **outiller efficacement les associations lors d'événements**, mais aussi **offrir une plateforme de gestion personnelle de collection aux passionnés**.

Pour les **associations**, cette application devient un outil de pilotage central :

- meilleure organisation des ventes de goodies
- accès rapide à toutes les informations produits depuis un mobile d'événements
- meilleur suivi des ventes, des stocks et des bénéfices
- simplification du travail de trésorier
- gain de temps et réduction des erreurs logistiques

Pour les **fans ou collectionneurs individuels**, elle offre une **solution dédiée à leur passion** :

- enregistrement précis de leur collection personnelle (nom, photo, prix, lieu de stockage…)
- historique d'achats et de valeurs
- possibilité de comparer des objets similaires entre utilisateurs (utile pour échanges ou estimations)
- interface claire, simple, disponible à la fois sur mobile et sur ordinateur

Le projet sera conçu selon une logique moderne, propre et maintenable :

- développement mobile-first avec une version web responsive si possible
- gestion agile avec outils collaboratifs (JIRA, GitHub…)
- respect des principes fondamentaux pour garantir un code propre et « scalable »
- Mise en œuvre systématique de tests unitaires pour chaque fonctionnalité, afin d'assurer fiabilité et robustesse

En somme, ce projet vise à devenir un **véritable assistant de gestion de collection de goodies**, **utile autant en interne associatif que pour la communauté des fans**, avec un fort potentiel de réutilisation et d'amélioration continue.

Se positionner par rapport aux projets similaires existants

Il existe aujourd'hui plusieurs solutions de gestion de stock ou de collections, mais aucune n'est **spécifiquement conçue pour les besoins d'associations orientées pop culture / événements**, ni pour les **fans de goodies liés à des licences musicales japonaises**.

Voici quelques types d'outils que l'on peut trouver :

- **Applications de gestion d'inventaire classiques** (ex : Sortly, Inventory Now, Stock and Inventory Simple)
  Ces outils sont souvent conçus pour des petites entreprises ou des commerces, avec une orientation B2B. Ils proposent une gestion de stock basique, mais ils manquent de personnalisation (pas d'espace collectionneur, pas de lien avec des communautés ou événements, etc)
- **Applications de collection / base de données personnelles** (ex : MyFigureCollection, Collectorz.com, Libib)
  Ces plateformes sont plus orientées vers les collectionneurs, mais elles ne proposent pas de gestion financière précise, ni de suivi des ventes lors d'événements, ni d'outils

collaboratifs pour les associations. Certaines sont très riches mais complexes, ou mal adaptées à un usage mobile rapide en convention.

- **Feuilles Excel personnalisées**
  De nombreuses d'associations ou fans utilisent des Google Sheets ou Excel pour suivre leurs objets. Bien que flexibles, ces outils deviennent rapidement limités et peu pratiques dès qu'on souhaite les utiliser en situation réelle (pas d'interface mobile, pas de photos, pas de multi-utilisateur, etc).

Stock Merch vient donc combler un double vide :

1. Un **outil simple, adapté et mobile-first** pour les associations lors d'événements.
2. Une **interface personnalisable et communautaire** pour les fans/collectionneurs, avec gestion de l'historique, des prix, des photos et des comparaisons.

Cette double orientation (professionnelle et personnelle) rend le projet **unique dans son approche**, et permet une réelle **valeur ajoutée** vis-à-vis par rapport aux solutions existantes.

Méthodologie et Organisation du travail

Le projet Stock Merch sera réalisé selon une approche agile, reposant sur des itérations courtes, des revues fréquentes et une gestion structurée du backlog à l'aide d'outils comme JIRA.

Méthodologie :

- Adoption d'une méthode agile (type Scrum/Kanban) avec un backlog priorisé et des tâches découpées en user stories.
- Livraisons progressives accompagnées de tests réguliers afin de vérifier la conformité des fonctionnalités aux attentes.
- Intégration des retours de l'équipe associative et des utilisateurs finaux (collectionneurs ou bénévoles).

Documentation continue et mise en place systématique de tests unitaires.

Outils employés :

J'utiliserai JIRA pour la gestion des tâches et le suivi de progression, GitHub pour la gestion du code, le stockage de la documentation. Pour ce qui est des tests unitaires, j'installerai Vitest et React Testing Library. Pour le front, je prendrai React + Capacitor (implémenté à la fin). Pour le backend, j'utiliserai une base de données qui se met à jour en direct. Pour enfin utiliser, TanStack Query pour mes requêtes API.

Projet Stock Merch

Thomas Siest

Sommaire

Présentation générale du projet 8
Objectifs du projet 8
Périmètre fonctionnel 8
Fonctionnalités principales (MVP) 8
Fonctionnalités secondaires (hors MVP – à implémenter plus tard) 8
Contraintes techniques 9
Contraintes de développement 9
Méthodologie de travail 9
Planning très prévisionnel 10
Livrables attendus 10

Présentation générale du projet

Nom du projet : Stock Merch
Porté par : Thomas Siest – Étudiant M2 CYBER 2024-2025
Contexte : Projet de Master visant à développer une application de gestion de stocks de goodies (ou « merch ») destinée à une utilisation associative (notamment pour l'association United Glowstick) ainsi qu'à des collectionneurs individuels.

Objectifs du projet

Le projet vise à développer une application web mobile-first (et mobile-native dans un second temps) pour :

- Faciliter la gestion des stocks de goodies lors des événements associatifs ;
- Permettre aux collectionneurs de gérer leur collection personnelle de manière simple et intuitive.

Périmètre fonctionnel

Fonctionnalités principales (MVP)

Pour les associations :

- Gestion CRUD des stocks de produits
- Visualisation des quantités en temps réel
- Suivi des ventes et bénéfices par événement
- Accès multi-utilisateur avec gestion des permissions

Pour les utilisateurs individuels :

- Ajout d'articles personnels (nom, description, photo, prix d'achat, lieu de stockage)
- Statistiques personnelles (nombre total d'objets, valeur estimée, etc.)

Fonctionnalités secondaires (hors MVP – à implémenter plus tard)

- Mode enchères pour certains produits
- Historique d'achats
- Comparaison de collections entre utilisateurs
- Exportation ou partage de collection

Contraintes techniques

- Développement initial mobile-first en React (application web responsive)
- Intégration de Capacitor prévue en fin de projet pour publication mobile (iOS, Android, PWA)
- Backend serverless via Convex.dev
- Récupération et mise à jour des données via TanStack Query
- Stockage d'images sur un service tiers (Cloudinary ou Supabase Storage)
- Hébergement sur Vercel, versionnage via GitHub

Contraintes de développement

- Respect des principes SOLID
- Architecture modulaire
- Tests unitaires obligatoires pour chaque nouvelle fonctionnalité (avec Vitest)
- Maquettage uniquement des fonctionnalités essentielles (via Figma)
- Documentation continue (technique et fonctionnelle)

Méthodologie de travail

- Méthodologie agile : organisation par itérations courtes (Scrum ou Kanban)
- Gestion du projet via JIRA / GitHub Projects
- Revue de code systématique
- Démonstrations régulières à l'équipe de l'association pour validation fonctionnelle

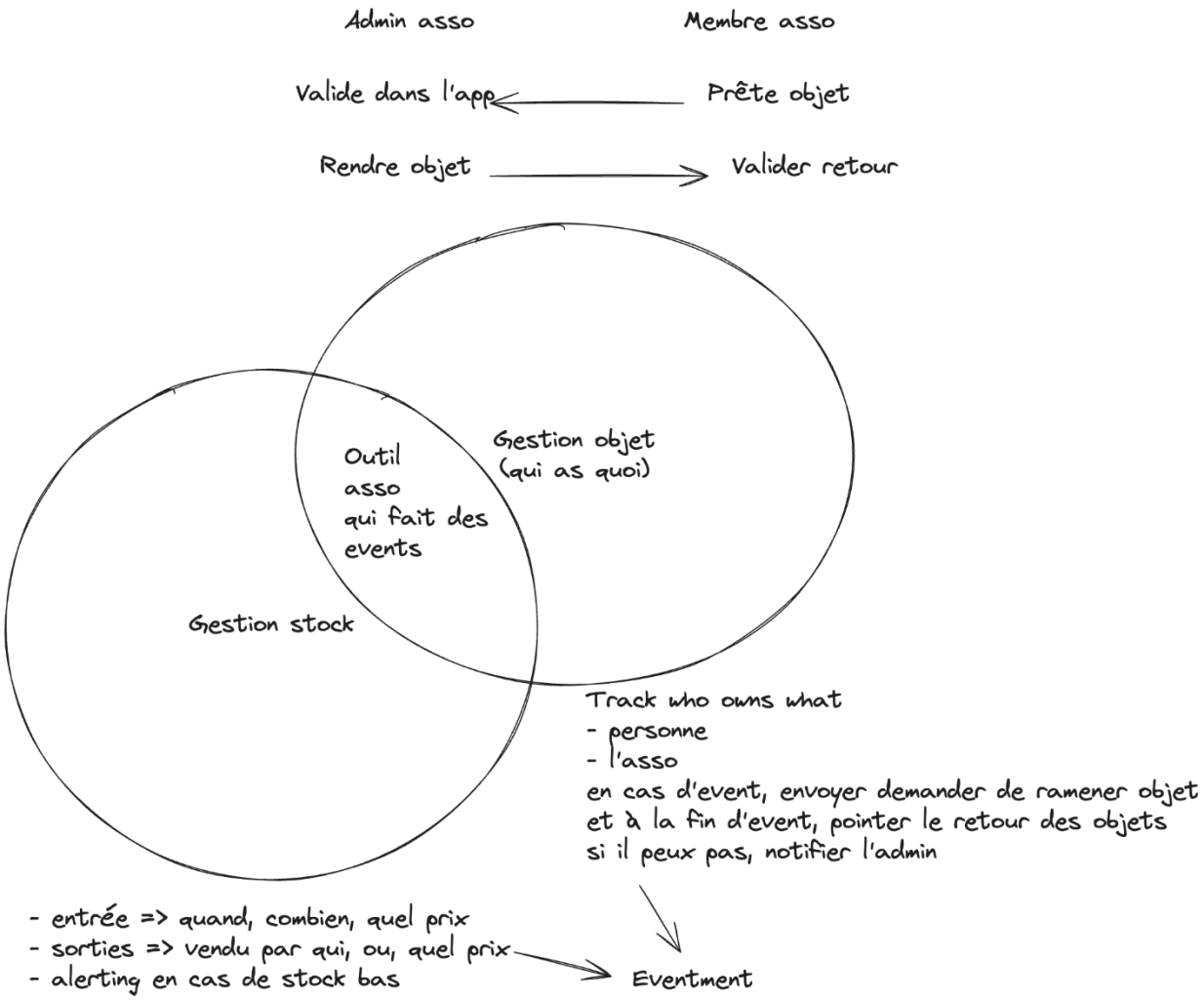
Planning très prévisionnel

Semaine	Dates	Objectifs principaux
Semaine 1	24 mars → 30 mars	Finalisation du cahier des charges, revue des besoins prioritaires (MVP), préparation backlog + découpage des tâches dans JIRA
Semaine 2	31 mars → 6 avril	Maquettes Figma basiques, initialisation repo GitHub, configuration de l'environnement React + Vite + Convex + TanStack
Semaine 3	7 avril → 13 avril	Développement authentification, modèles backend, affichage goodies
Semaine 4	14 avril → 20 avril	CRUD goodies, stockage de photos, premiers tests unitaires
Semaine 5	21 avril → 27 avril	Quantité en temps réel, permissions utilisateurs
Semaine 6	28 avril → 4 mai	Ventes par événement, bénéfices, statistiques personnelles
Semaine 7	5 mai → 11 mai	Tests manuels et automatiques, responsive, retours des utilisateurs
Semaine 8	12 mai → 18 mai	Correctifs, intégration Capacitor si possible, documentation utilisateur
Semaine 9	19 mai → 25 mai	Documentation technique, préparation démo, cas d'usage
Semaine 10	26 mai → 31 mai	Révision finale du code, soutenance, livraison finale

Livrables attendus

- Application web responsive fonctionnelle
- Code source versionné sur GitHub
- Jeux de tests unitaires et rapport de couverture
- Présentation orale (soutenance)
- Cahier des charges (ce document)

Conception de la solution



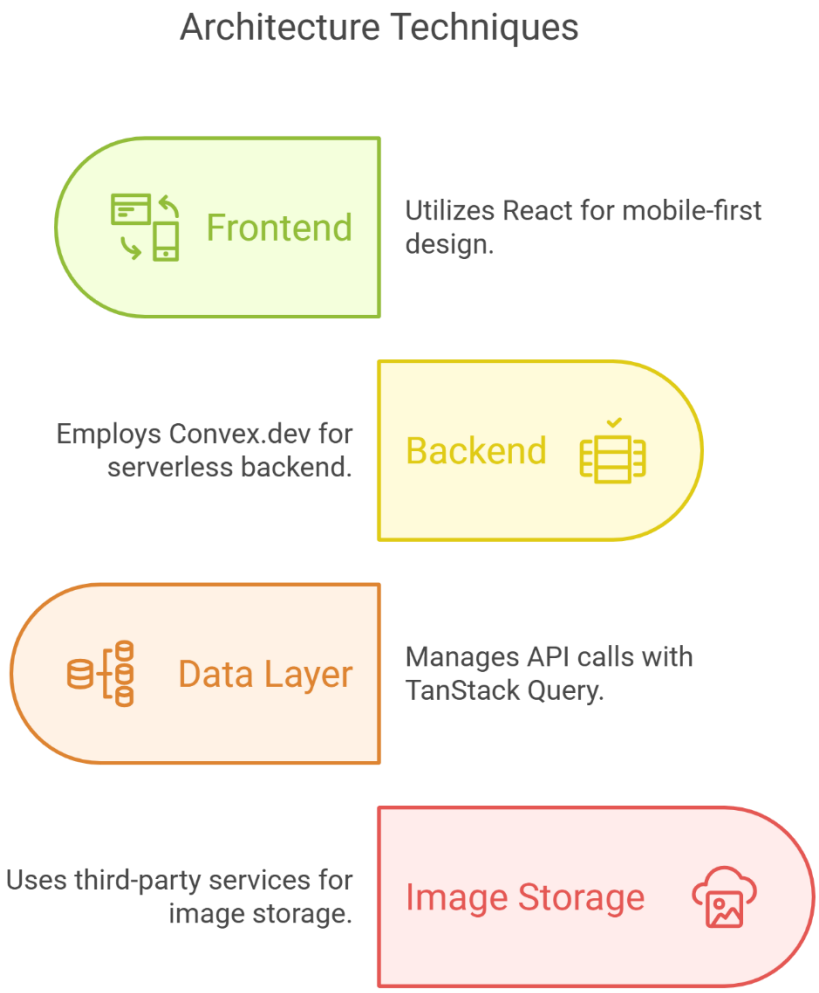
J'ai conçu Stock Merch avec pour but la maintenabilité, la scalabilité et la portabilité de la solution. Mon développement suivra une approche mobile-first, en commençant par une application web pour les supports mobiles. Mon application vise deux types d'utilisateurs : les associations et les collectionneurs individuels.

Elle proposera aux associations une gestion intégrale de l'inventaire de produits : ajout, changement et élimination d'articles. L'application offrira aussi la possibilité de surveiller les stocks en direct, d'enregistrer les transactions et de déterminer automatiquement le profit issu des événements. Un système de gestion des utilisateurs, doté de divers niveaux d'autorisations, sera également instauré afin d'assurer une utilisation collaborative sécurisée.

L'application mettra à disposition des collectionneurs une interface dédiée pour consigner leurs objets personnels, incluant des détails comme le nom, la description, une image, le coût d'acquisition ou encore l'emplacement de rangement. Des statistiques personnalisées seront générées automatiquement afin d'offrir une vue d'ensemble sur la collection de l'utilisateur, incluant sa valeur estimée et le nombre total d'objets.

Certaines fonctionnalités secondaires seront intégrées ultérieurement. Il s'agira notamment de la consultation de l'historique des achats, de la comparaison entre collections d'utilisateurs, de la mise en place d'un mode enchère, ainsi que d'un système d'export ou de partage de collection.

Enfin, l'architecture technique sera construite autour d'une séparation claire des responsabilités entre le frontend, le backend et la gestion des données. J'ai choisi des outils modernes et adaptés à mon approche mobile-first afin d'assurer un développement fluide, efficace et pérenne, en accord avec les bonnes pratiques enseignées dans ma formation.



On distinguera deux principaux scénarios d'utilisation de l'application. Dans le cadre d'un accord, le bureau de l'association aura un accès aisé à l'interface pour vérifier le stock en direct, apporter des changements sur les articles en vente et consigner les transactions de façon simple et intuitive. Simultanément, pour un usage privé, les collectionneurs auront à leur disposition une zone spécifique pour gérer leur collection personnelle, avec l'option d'inclure, structurer et superviser leurs objets, tout en ayant accès à des résumés statistiques concernant la valeur et l'étendue de leur collection.

Réalisation / Mise en œuvre

Le choix technologique repose sur des outils modernes, open-source, et adaptés à une démarche agile et incrémentale.

Partie	Technologie	Justification
Frontend	React	Développement rapide, communauté large, adapté à la création d'UI responsive.
Capacitor	Surcouche en plus du react (à la fin du projet)	Permet la création d'applications mobiles (iOS, Android, PWA)
Backend	Convex.dev	Solution serverless idéale pour un backend en temps réel, avec persistance des données et appels API simplifiés.
Data Fetching	TanStack Query	Gestion optimisée du cache, des revalidations, des mutations, avec une excellente intégration React.
Tests	Vitest	Léger, rapide, compatible avec Vite et parfait pour les tests unitaires (composants, logique métier).
Maquettage	Figma	Prototypage rapide pour l'UI/UX des fonctionnalités essentielles.
CI/CD & Déploiement	GitHub + Vercel	Déploiement continu du frontend sur Vercel, facilité de collaboration et de gestion des versions via GitHub.

Cette stack technique permet une mise en œuvre rapide, avec une forte évolutivité pour intégrer de futures fonctionnalités (chat communautaire, système d'échange, notifications...).

Tests / Validation / Vérification

La qualité de mon application est un des piliers dans mon développement. Dès le début, je vais mettre en place des tests unitaires pour chaque fonctionnalités. De plus, il y aura sûrement des vérifications directement sur les environnements dynamiques pour chaque Pull Request. Mon objectif étant de rester au-dessus de 80% de validation de tests.

En pratique, trois types de tests seront effectuées. Les tests unitaires me donnent la possibilité de vérifier la logique métier, en particulier les calculs et traitements de données au sein de l'application. Puis, des tests de composants sont effectués pour garantir la correcte représentation des éléments d'interface et leur réponse appropriée dans divers scénarios d'utilisation. Pour finir, j'intègre des tests d'intégration partiels qui ont pour but de valider le fonctionnement correct entre les modules essentiels (par exemple : ajout d'un article et mise à jour de l'inventaire).

Pour ça, je vais employer des logiciels à la pointe tels que Vitest, qui me donne la possibilité d'effectuer les tests unitaires de manière efficace. Ce dernier sera également intégré dans mes pipelines CI à travers GitHub Actions. De ce fait, chaque demande de Pull Request déclenchera automatiquement une série de tests. J'emploierai aussi React Testing Library pour reproduire les principales interactions des utilisateurs, comme si nous étions dans la vrai vie.