Projet DIGICHEESE — Gestion de projet (cadre & plan)

Livrables attendus (projet DIGICHEESE)

Le projet **DIGICHEESE** consiste à livrer :

- **Une API FastAPI** (endpoints REST) avec **Swagger** accessible via `/docs`
- **Une base MySQL** avec les tables nécessaires au **CRUD Admin obligatoire** :
  - communes
  - objets
  - conditionnements
  - poids
  - poids-vignettes
  - utilisateurs
  - adresses
- **Des tests automatisés (Pytest)** :
  - au minimum **1 test nominal** + **1 test erreur** par ressource admin
  - tests **passants** + tests **non passants** (référencés dans **SquashTM**)
- **Un dépôt Git propre** :
  - branches `master`, `dev`, `test`, `prod`
  - une branche par développeur
  - `requirements.txt`
  - scripts SQL
  - documentation, etc.

Objectifs (gestion de projet)

En gestion de projet, l'objectif est de :

- **Décomposer** le projet en **User Stories** + **tâches** + **critères d'acceptation**
- **Organiser** un board Agile (**Scrum** ou **Kanban**) avec des colonnes et champs adaptés
- **Simuler** un sprint fictif (issues déplacées, estimées, assignées)
- Faire une **analyse de collaboration / retour d'expérience** sur vos **5 jours Python**

Choix de l'outil (attendu TP)

Le TP autorise :

- **Option A : Jira**
- **Option B : GitHub Projects**

Recommandations :

- **Jira** : recommandé si tu veux une gestion "pro" (burndown, stories, points)
- **GitHub Projects** : recommandé si tu veux rester cohérent avec le dépôt code

Dans tous les cas, le **rendu attendu** est :

- un **lien** vers le projet (Jira ou GitHub) + **invitation** au correcteur (si demandé)
- un **backlog complet** (user stories + tâches + estimations)
- un **board** mis à jour avec un **sprint fictif**

Plan général de gestion de projet

Projet : Refonte du SI de gestion des cadeaux fidélité — Fromagerie **DIGICHEESE**

0. Choix de l'outil de gestion de projet

- **Outil retenu :** Jira ou GitHub Projects
- **Objectif :** piloter le développement de l'API Python (**FastAPI + MySQL**)

1. Cadrage du projet (vision, contexte et objectifs)

1.1 Contexte du projet

Refonte complète d'un système existant vieux de **20 ans** (Access + VBA).

Problèmes identifiés :

- instabilité
- maintenance difficile
- interface obsolète

1.2 Objectifs du projet

- Moderniser le SI de gestion des cadeaux fidélité
- Centraliser les données dans une base fiable
- Faciliter l'évolution future (API, web, services)

1.3 Périmètre fonctionnel

Application interne (Intranet) destinée aux salariés.

Profils / rôles fonctionnels :

- Administrateur
- Opérateur colis

2. Identification des acteurs et rôles projet

2.1 Acteurs métier (issus du cahier des charges)

- Administrateur
- Gestionnaire de colis (**OP-colis**)
- Gestionnaire de stock (**OP-stocks**)

2.2 Rôles projet (gestion Agile)

- Product Owner (PO)
- Développeurs backend (Python / FastAPI)
- Éventuellement : référent base de données / qualité

3. Organisation fonctionnelle par grands domaines (macro-périmètre)

Cette partie découpe le projet comme dans le cahier des charges.

3.1 Domaine Administration

Fonctionnalités de paramétrage et de gestion :

- Gestion des utilisateurs
- CRUD des paramètres métiers :
  - communes
  - objets (articles)
  - conditionnements
  - poids
  - poids-vignettes
- Génération d'impressions

3.2 Domaine Gestion des colis

Fonctionnalités opérationnelles :

- Gestion des clients
- Gestion des commandes (CRUD + calculs)

- Gestion du conditionnement
- Statistiques par période
- Mailing client

3.3 Domaine Gestion des stocks

Fonctionnalités de suivi :

- Gestion du stock
- Mise à jour annuelle des stocks
- Impression des états de stock

4. Décomposition du projet en Épics (structure du backlog)

À partir du cahier des charges, le backlog est structuré en Épics fonctionnels :

- **EPIC-SETUP** : Initialisation du projet (repo, branches, environnement)
- **EPIC-AUTH** : Authentification et gestion des rôles
- **EPIC-COLIS** : Gestion des colis et commandes
- **EPIC-STOCK** : Gestion et mise à jour des stocks
- **EPIC-DATABASE** : Modélisation et gestion de la base de données
- **EPIC-TESTS** : Tests automatisés (nominal + erreurs)
- **EPIC-DOC** : Documentation technique et fonctionnelle

> Ces épics correspondent aux chapitres fonctionnels, use cases et schémas du cahier des charges.

5. Déclinaison en User Stories (logique Agile)

Pour chaque Epic :

- Définition de **User Stories** basées sur :
  - les use cases Admin
  - les use cases Gestionnaire de colis
  - les use cases Gestionnaire de stock

Chaque User Story contient :

- description fonctionnelle
- valeur métier
- critères d’acceptation
- dépendances

> À détailler après.

6. Planification Agile (sprints ou flux Kanban)

- Création d'un backlog priorisé
- Sélection des User Stories critiques :
  - CRUD Admin (obligatoires)
  - base de données
  - API
  - tests
- Création d'un sprint fictif ou d'une itération simulée
- Déplacement des tickets sur le board :
  - `To Do → In Progress → Code Review → Done`

7. Suivi de l'avancement et collaboration

- Mise à jour régulière du board
- Ajout de commentaires (choix techniques, blocages)
- Liens avec :
  - commits Git
  - branches
  - documentation
  - Swagger

8. Qualité, validation et conformité

- Vérification de la couverture fonctionnelle :
  - conformité avec le cahier des charges
- Validation :
  - des endpoints
  - des règles métier
  - des données persistées
- Tests :
  - tests unitaires
  - tests d'erreurs
  - validation des cas limites

9. Rétrospective et bilan de projet

- Analyse de la communication d'équipe
- Répartition du travail
- Solutions mises en place
- Axes d'amélioration pour un futur projet

Sections à détailler

- **PARTIE 1 — Cadrage du projet** : 001 - Cadrage du projet
- **PARTIE 2 — Identification des acteurs et rôles projet** : 002 - Identification des acteurs et rôles projet
- **PARTIE 3 — Organisation fonctionnelle par grands domaines (macro-périmètre)** : à détailler
- **PARTIE 4 — Décomposition du projet en Epics (structure du backlog)** : à détailler
- **PARTIE 5 — Déclinaison en User Stories (logique Agile)** : à détailler
- **PARTIE 6 — Planification Agile (sprints ou flux Kanban)** : à détailler
- **PARTIE 7 — Suivi de l'avancement et collaboration** : à détailler
- **PARTIE 8 — Qualité, validation et conformité** : à détailler
- **PARTIE 9 — Rétrospective et bilan de projet** : à détailler