



Rapport de Gestion de Projet
Gestion d'un stock pharmaceutique

MARCOUX Corentin - BAUDAT Louis - LOBREAU Romain - MOHAMED-NOUR MAROC

Table des matières

Table des matières	2
Sprint 1 (30 Septembre - 13 Octobre)	3
Objectifs et Tâches du sprint	3
Mêlée 1	3
Résumé	3
Tasks Board	4
Rendu	4
Mêlée 2	5
Résumé	5
Tasks Board	6
Rendu	6
Mêlée 3	7
Résumé	7
Tasks Board	8
Rendu	8
Mêlée 4	10
Résumé	10
Tasks Board	11
Sprint 2 (14 Octobre - 03 Novembre)	11
Objectifs et Tâches du sprint	11
Mêlée 1	11
Résumé	12
Tasks Board	12
Rendu	12
Mêlée 2	15
Résumé	16
Tasks Board	16
Rendu	16
Mêlée 3	18
Résumé	19
Tasks Board	19
Rendu	19
Mêlée 4	20
Résumé	21
Tasks Board	21
Rendu	21

SAE - Rapport de Gestion de Projet

Sprint 1 (30 Septembre - 13 Octobre)

Objectifs et Tâches du sprint

L'objectif de ce premier sprint était de mettre en place les outils collaboratifs, rédiger les éléments de base du cahier des charges, et établir les bases visuelles et conceptuelles.

Mêlée 1

30 Septembre 2024

Objectif : Initialisation du projet et mise en place des outils de collaboration.

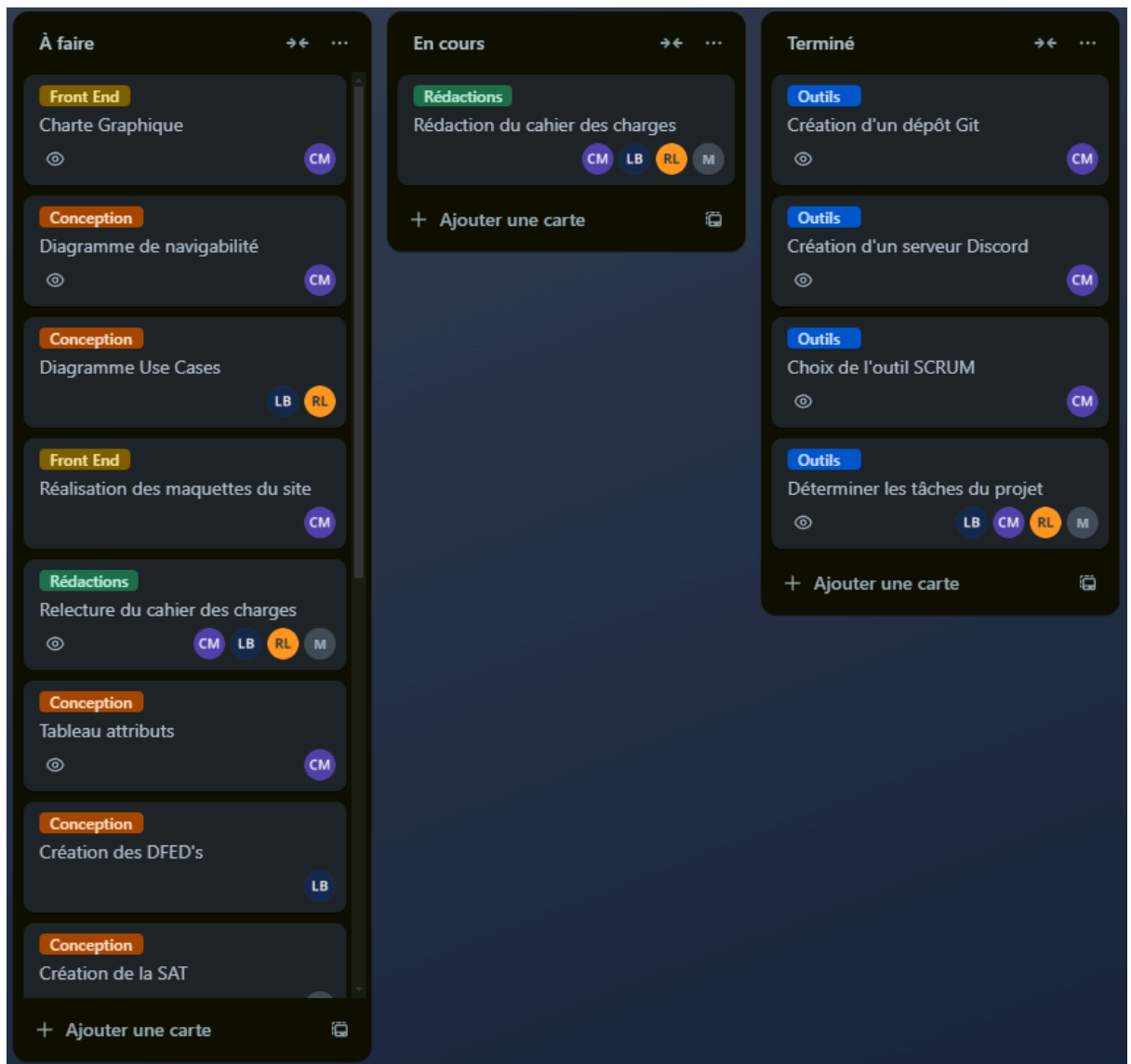
Résumé

- **Création du dépôt Git :** Mise en place du dépôt Git pour versionner et centraliser le projet. Cela permet d'assurer le suivi des modifications du code et de faciliter la collaboration.
- **Création du serveur Discord :** Mise en place du serveur Discord pour assurer une bonne communication entre les membres de l'équipe. Avec différents canaux de discussions.
- **Choix de l'outil SCRUM :** Sélection d'un outil SCRUM pour la gestion des sprints et du suivi des tâches.
- **Début de la rédaction du cahier des charges**

Lors de cette première mêlée, nous avons mis en place les fondations techniques du projet, à savoir les outils de communication, de gestion SCRUM, et de versionnement. Cela a permis de s'assurer que toute l'équipe dispose des outils nécessaires pour travailler efficacement ensemble. Nous avons aussi débuté la rédaction du cahier des charges du projet.

SAE - Rapport de Gestion de Projet

Tasks Board



Rendu

Outil SCRUM :

Nous utiliserons **TRELLO** pour la gestion des tâches du projet. Chacun des membres du groupe peut y accéder.

→ Lien du projet TRELLO :

<https://trello.com/invite/b/670d57608e9a5065afde77eb/ATTIc38df0007d0d1b6778fc24aceaa6f3b21959D4D4/sae3-gestion-dun-stock-pharmaceutique>

Git du projet :

SAE - Rapport de Gestion de Projet

The screenshot shows a GitHub repository page for 'SAE3-Gestion-stock-pharmaceutique'. The repository has 1 commit, 1 branch, and 0 tags. The project storage is 210 B. The README file is highlighted. The commit history table shows the following data:

Name	Last commit	Last update
files	Ajout des documents + images	1 minute ago
img	Ajout des documents + images	1 minute ago
README.md	Initial commit	4 minutes ago

Project information:

- 1 Commit
- 1 Branch
- 0 Tags
- 210 B Project Storage

README

- [Add LICENSE](#)
- [Add CHANGELOG](#)
- [Add CONTRIBUTING](#)
- [Enable Auto DevOps](#)
- [Add Kubernetes cluster](#)
- [Set up CI/CD](#)
- [Add Wiki](#)
- [Configure Integrations](#)

Serveur Discord :



Mêlée 2

30 Septembre 2024

Objectif : Réalisation de la charte graphique

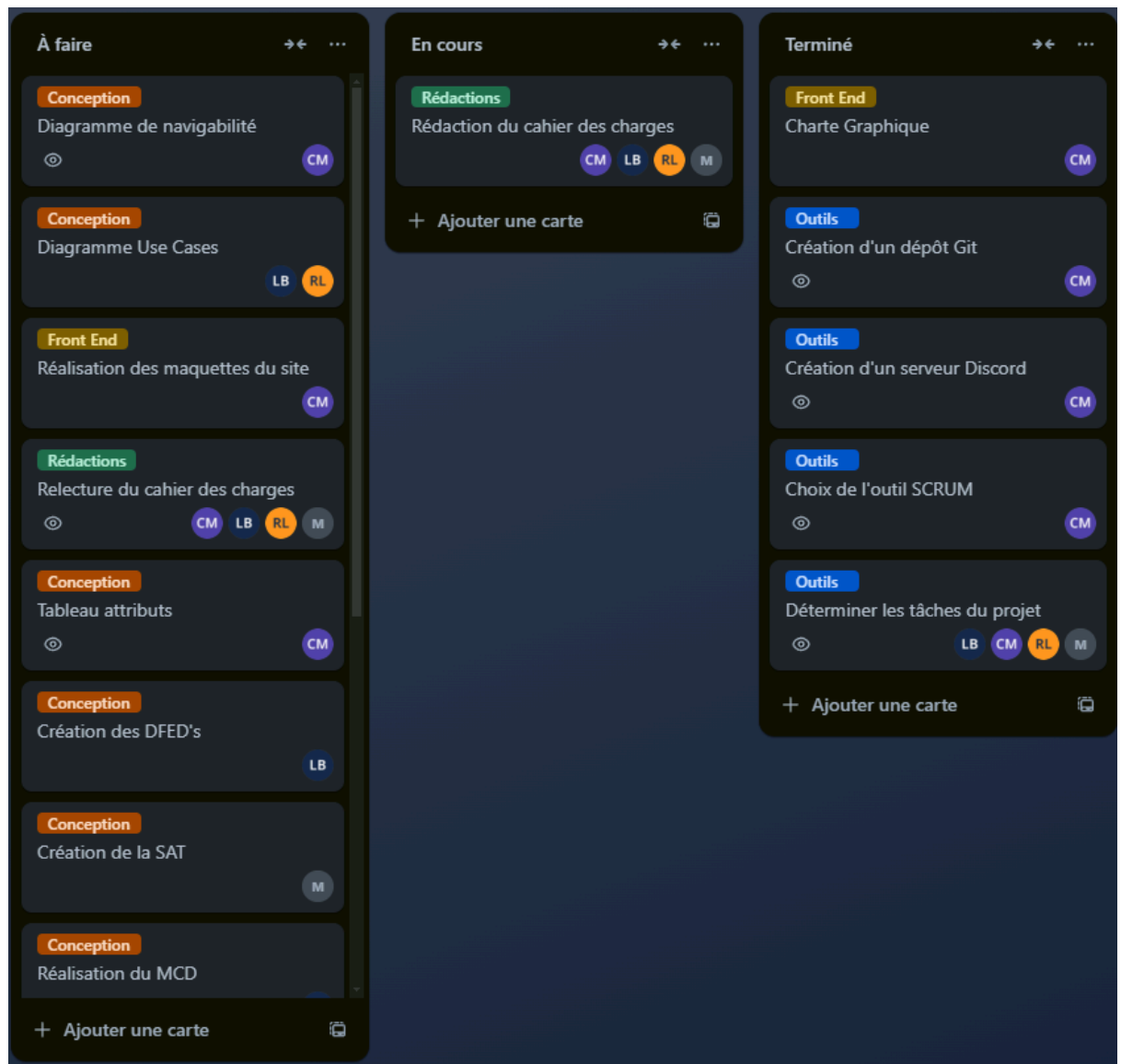
Résumé

- **Réalisation de la charte graphique :** Création d'une charte graphique qui définit les couleurs, polices et styles visuels qui seront utilisés dans l'interface du site web. Cette charte graphique assure une identité visuelle cohérente à travers toutes les pages de l'application.

Lors de cette deuxième mêlée, nous avons défini les premiers aspects visuels de notre site.

SAE - Rapport de Gestion de Projet

Tasks Board



Rendu

Charte graphique :

SAE - Rapport de Gestion de Projet

Charte Graphique

Logos :



Palette De Couleur :



Typographie :

Rubik

Arsenal

Palanquin

Titre de niveau 1 - "Rubik"

Titre de niveau 2 - "Arsenal"

Fuerit toto in consulatu sine provincia, cui fuerit, antequam designatus est, decreta provincia. Sortietur an non? Nam et non sortiri absurdum est, et, quod sortitus sis, non habere. Proficiscetur paludatus? Quo? Quo pervenire ante certam diem non licebit. Ianuario, Februario, provinciam non habebit; Kalendis ei denique Martiis nascetur repente provincia.

Mêlée 3

04 Octobre 2024

Objectif : Définir les aspects visuels et conceptuels du projet.

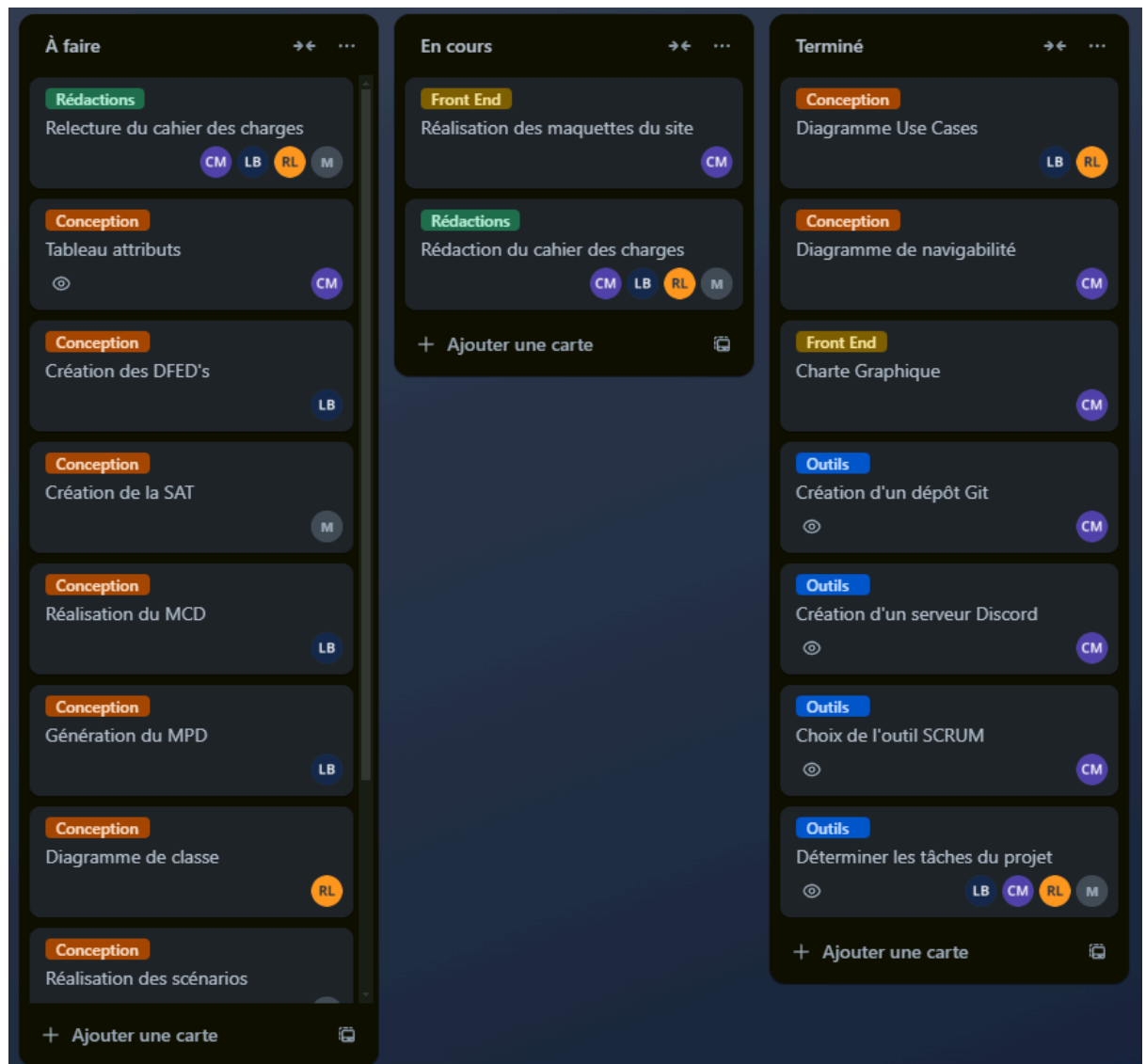
Résumé

- **Création du diagramme de navigabilité** : Un diagramme de navigabilité a été produit pour représenter les différentes pages du site et les liens de navigation entre elles.
- **Création du Diagramme UC (cas d'utilisation)**
- **Début des maquettes du site** : Création des premières maquettes de l'interface, en s'appuyant sur la charte graphique et les parcours définis par le diagramme de navigabilité. Ces maquettes servent à visualiser la structure du site et à valider les choix UX/UI.

SAE - Rapport de Gestion de Projet

Lors de cette troisième mêlée, l'équipe a défini l'architecture de navigation du site, créé le diagramme UC pour clarifier les interactions utilisateurs ainsi que créé les premières maquettes du sites à partir de la charte graphique.

Tasks Board



Rendu

Diagramme de navigabilité :

SAE - Rapport de Gestion de Projet

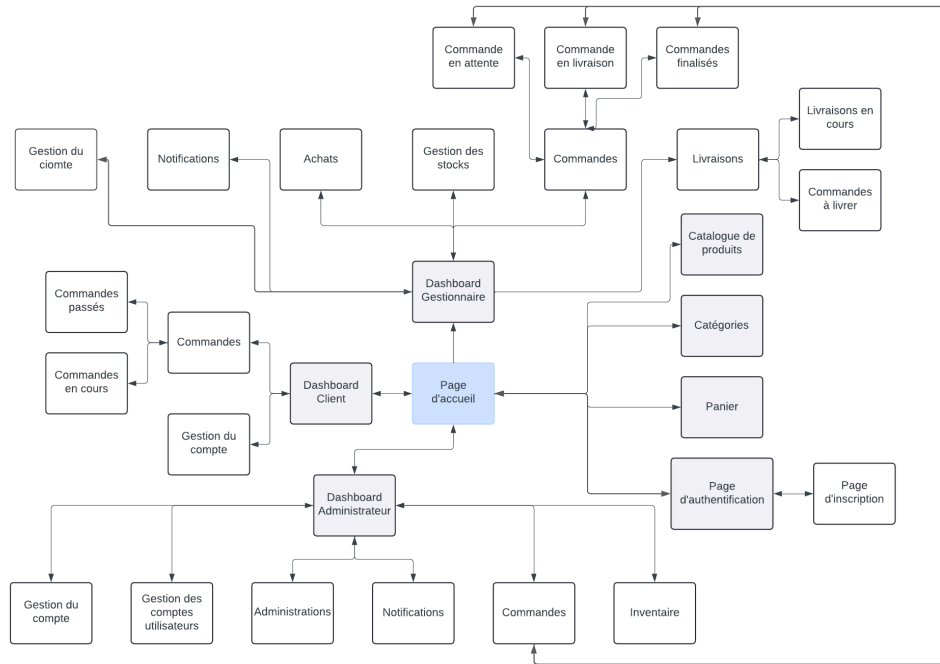
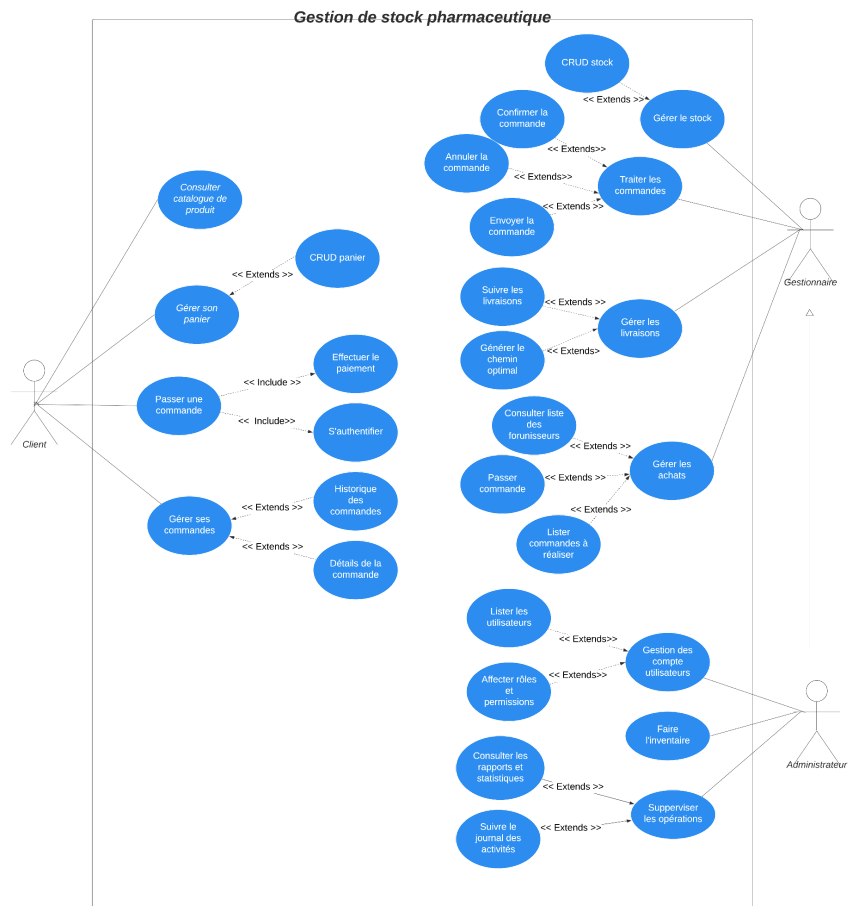
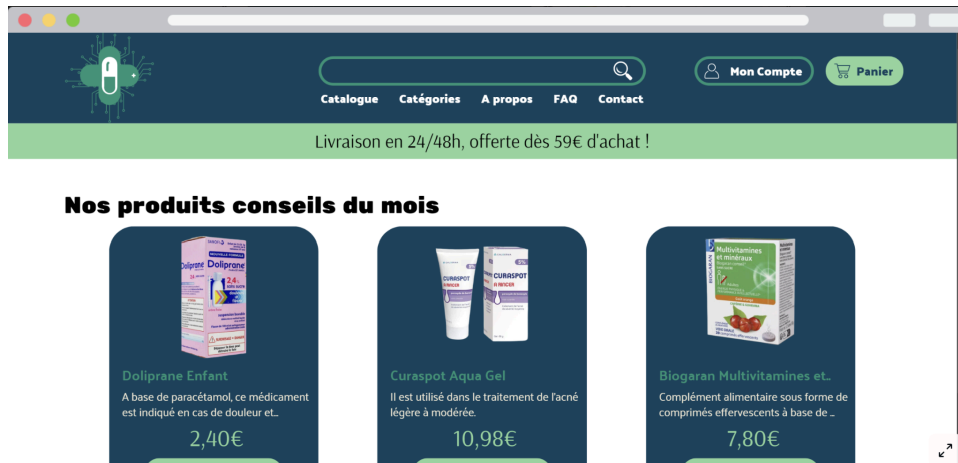


Diagramme UC :



SAE - Rapport de Gestion de Projet

Première maquette :



Mêlée 4

11 Octobre 2024

Objectif : Finaliser le cahier des charges

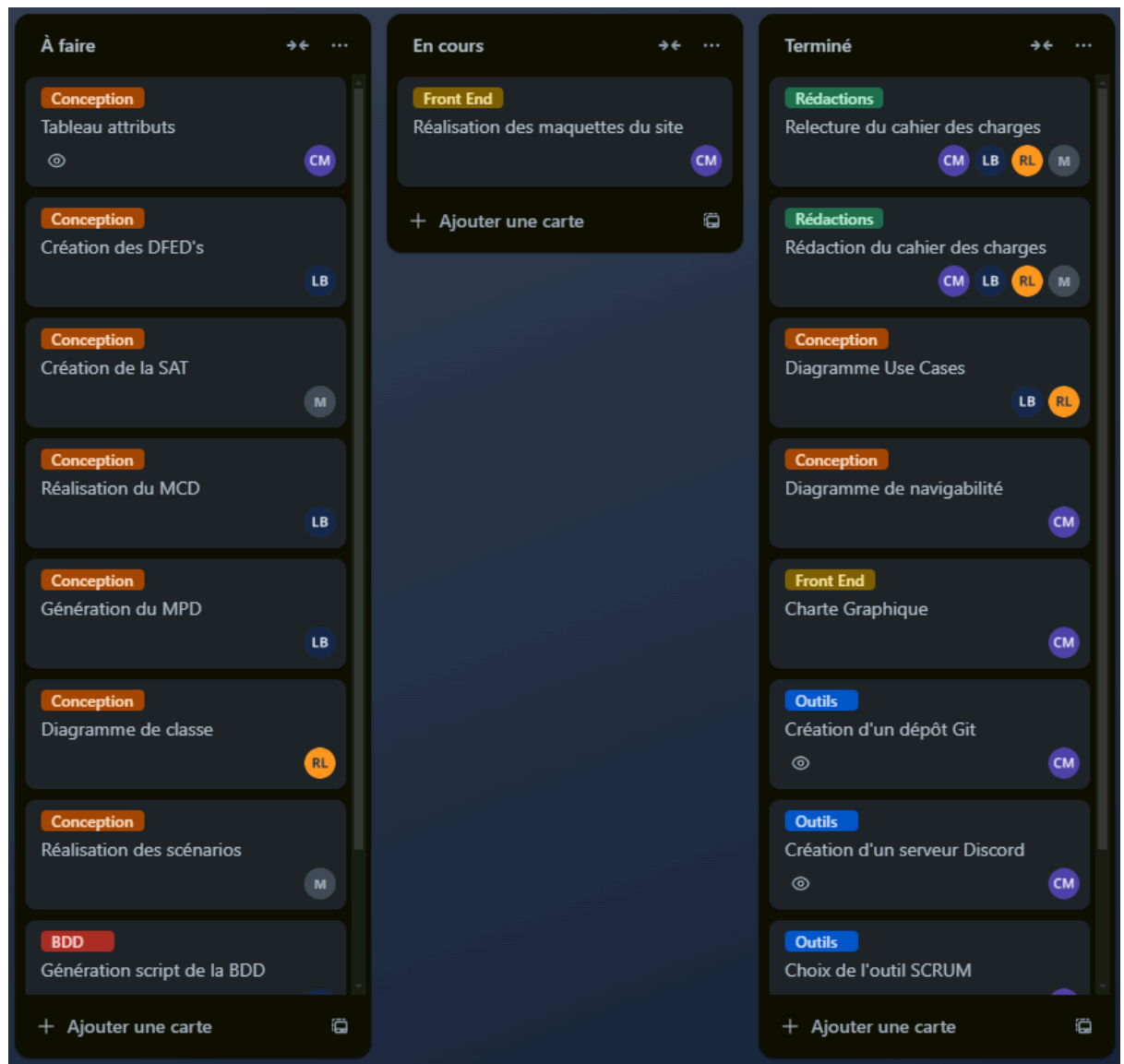
Résumé

- **Finalisation du cahier des charges**
- **Relecture et validation du cahier des charges**

Lors de cette quatrième et dernière mêlée du sprint, l'équipe a finalisé et relu le cahier des charges.

SAE - Rapport de Gestion de Projet

Tasks Board



Sprint 2 (14 Octobre - 03 Novembre)

Objectifs et Tâches du sprint

L'objectif du second sprint était de structurer la base de données en réalisant les différents diagrammes de conception et de préparer les fondations nécessaires pour le développement et l'implémentation des fonctionnalités du site.

Mêlée 1

17 Octobre 2024

SAE - Rapport de Gestion de Projet

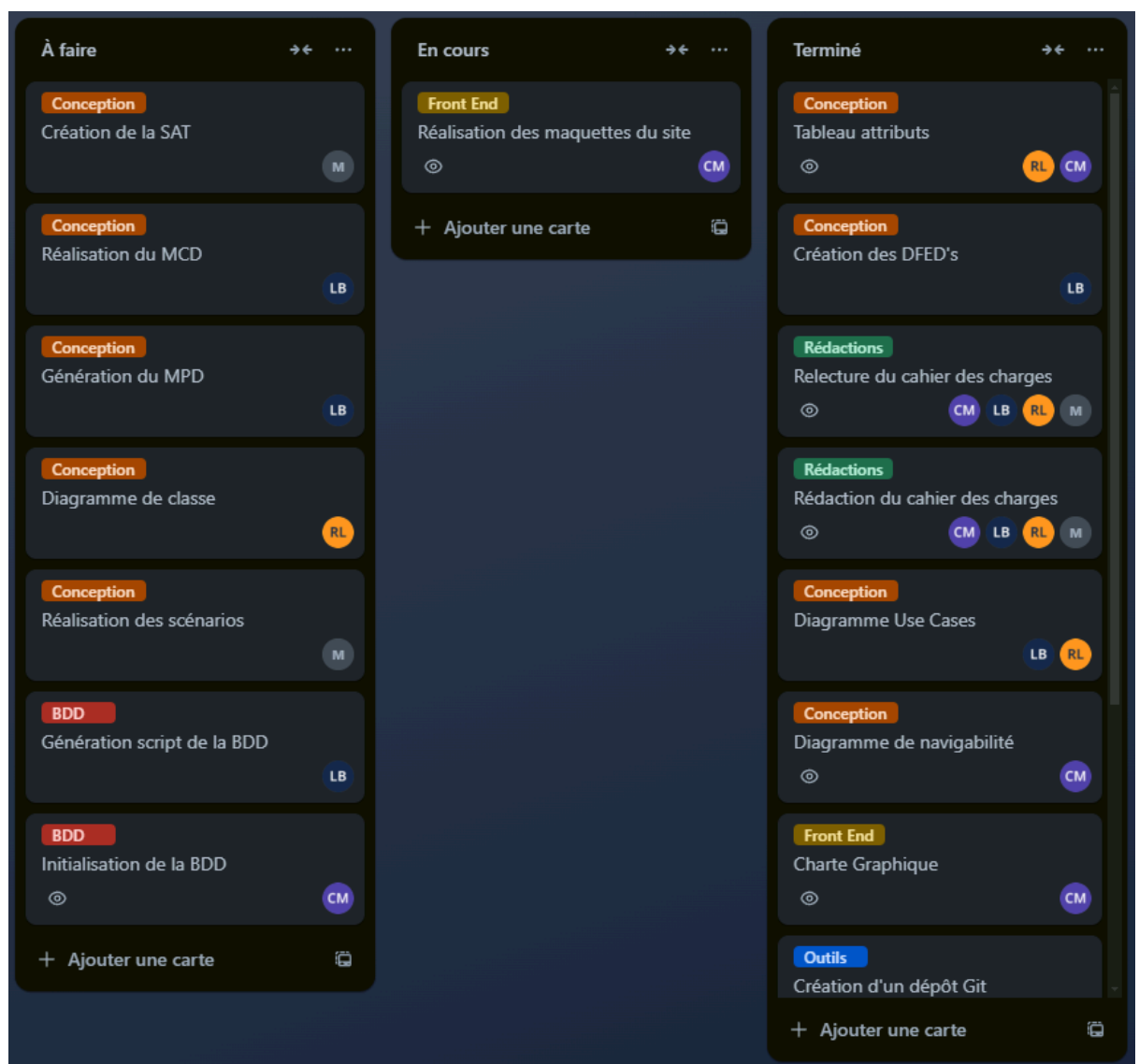
Objectif : Créer le tableau d'attributs et les DFEDs pour structurer les données du projet et planifier les flux d'information.

Résumé

- **Création du tableau d'attributs :** Définition des attributs nécessaires pour les entités principales du projet.
- **Création des DFED's**

Cette première mêlée a permis de clarifier la structure des données et les flux d'information nécessaires.

Tasks Board



Rendu

Tableau d'attributs :

SAE - Rapport de Gestion de Projet

Attribut	Description	Type
idProd	Identifiant du produit	INTEGER
libProd	Libellé du produit	VARCHAR100
descProd	Description du produit	VARCHAR150
prixProd	Prix du produit	FLOAT
imgProd	Image du produit	BLOB
dateExpiration	Date d'expiration du produit	DATE
idCat	Identifiant d'une catégorie	INTEGER
libCat	Libellé d'une catégorie	VARCHAR30
idCli	Identifiant d'un client	INTEGER
nomCli	Nom d'un client	VARCHAR50
mailCli	Mail d'un client	VARCHAR50
telCli	Numéro de téléphone d'un client	CHAR10
loginCli	Nom d'utilisateur d'un client	VARCHAR50
mdpCli	Mot de passe hashé d'un client	SHA512(VARCHAR50)
cpCli	Code postal d'un client	CHAR5
villeCli	Ville d'un client	VARCHAR50
prénom	Prénom d'un client physique	VARCHAR30
dateNais	Date de naissance d'un client	DATE
numRPPS	Numéro RPPS d'un cabinet médical	CHAR11
idFour	Identifiant d'un fournisseur	INTEGER
libFour	Nom d'un fournisseur	VARCHAR50
cpFour	Code postal d'un fournisseur	CHAR5
villeFour	Ville du fournisseur	VARCHAR50
telFour	Numéro de téléphone d'un fournisseur	CHAR10

SAE - Rapport de Gestion de Projet

mailFour	Adresse mail d'un fournisseur	VARCHAR50
idCmde	Identifiant de la commande	INTEGER
numeroCmde	Numéro de la commande	VARCHAR20
dateCmde	Date de la commande	DATE
quantiteCmde	Quantite du produit commandé	INTEGER
prixUnitaireCmde	Prix unitaire du produit commandé	FLOAT
idAppr	Identifiant de l'approvisionnement	INTEGER
dateAppr	Date de l'approvisionnement	DATE
quantiteAppr	Quantité d'un produit reçue	INTEGER
prixUnitaireAppr	Prix unitaire du produit reçue	FLOAT
idStock	Identifiant du stock	INTEGER
quantiteStock	Quantite disponible dans le stock	INTEGER
seuilAlerte	Seuil d'alerte pour réapprovisionnement	INTEGER
idEmploye	Identifiant de l'employé	INTEGER
nomEmploye	Nom de l'employé	VARCHAR30
pnomEmploye	Prénom de l'employé	VARCHAR30
telEmploye	Numéro de téléphone d'un employé	CHAR10
mailEmploye	Adresse Mail d'un employé	VARCHAR30
loginEmploye	Nom d'utilisateur d'un employé	VARCHAR30
mdpEmploye	Mot de passe hashé d'un employé	SHA512(VARCHAR30)
idInventaire	Identifiant de l'inventaire	INTEGER
dateInventaire	Date de l'inventaire	DATE
idLivraison	Identifiant d'une livraison	INTEGER

SAE - Rapport de Gestion de Projet

dateLivraison	Date d'une livraison	DATE
quantiteLiv	Quantité d'une livraison	INTEGER
adrLivraison	Adresse d'une livraison	VARCHAR50
cpLivraison	Code postal d'une livraison	CHAR5
villeLivraison	Ville d'une livraison	VARCHAR50

DFED's :

Produit : idProd -> libProd, descProd, prixProd, imgProd, datePeremption

Catégorie : idCat -> libCat

Commande : idCmde -> numeroCmde, dateCmde,

LigneCmde : idCmde, idProduit -> quantiteCmde, prixUnitaireCmde

Approvisionnement : idAppr -> dateAppr

LigneApprovisionnement : idAppr, idProd -> quantiteAppr, prixUnitaireAppr

Fournisseur : idFour -> libFour, adrFour, cpFour, villeFour, telFour, mailFour

Stock : idStock -> quantiteStock, seuilAlerte

Inventaire : idInventaire -> dateInventaire

Employé : idEmploye -> nomEmploye, pnomEmploye, mailEmploye, telEmploye, loginEmploye, mdpEmploye

Livraison : idLivraison -> dateLivraison, quantiteLiv, adrLiv, cpLiv, villeLiv

Client : idCli -> nomCli, mailCli, telCli, loginCli, mdpCli, adrCli, cpCli, villeCli

ClientPhysique : idCli -> pnomCli, dateNais

CabinetMédical : idCli -> numRPPS

Mêlée 2

18 Octobre 2024

Objectif : Définir la structure de la base de données en réalisant la SAT et le MCD, en vue d'organiser les tables et leurs relations.

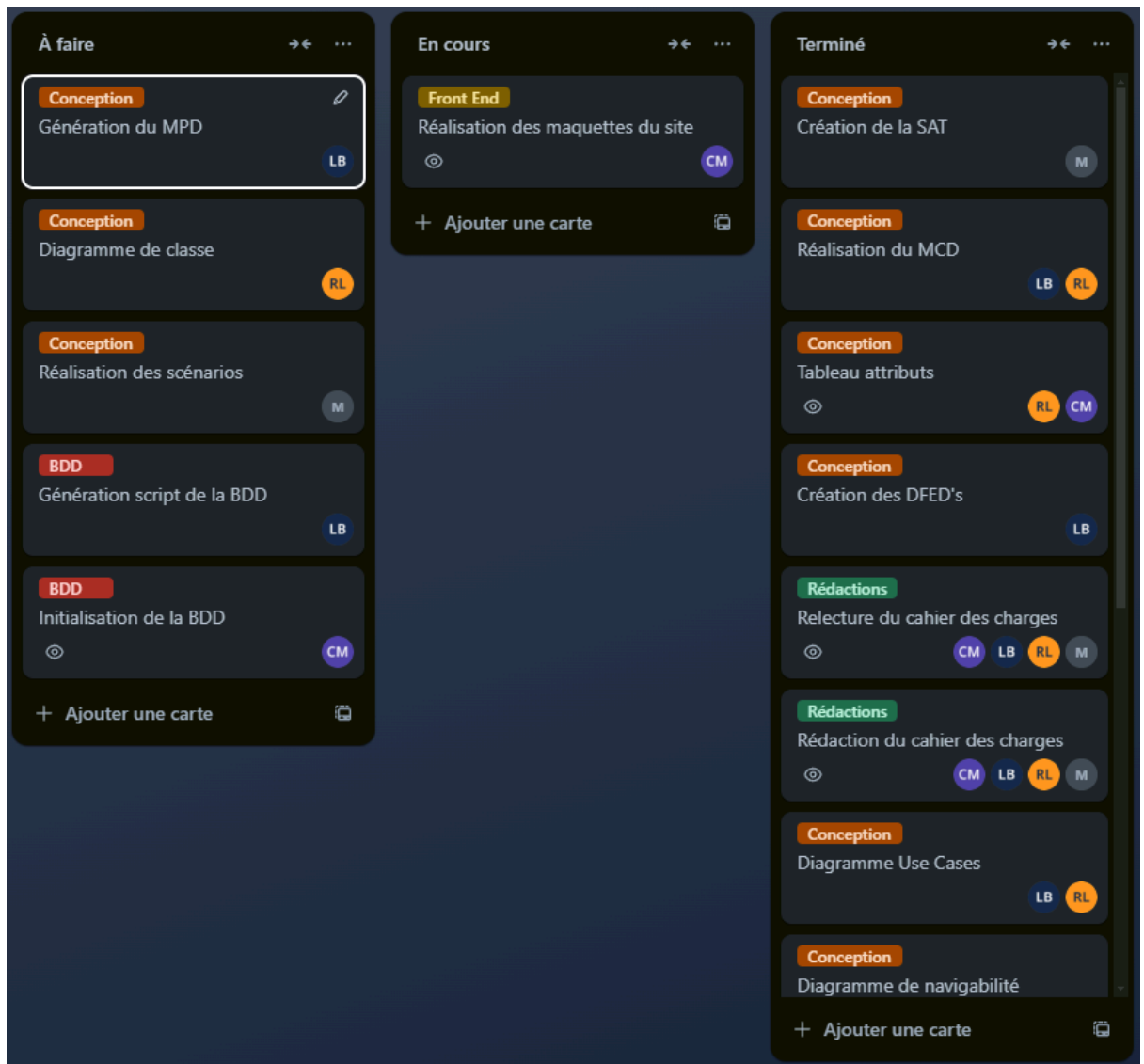
SAE - Rapport de Gestion de Projet

Résumé

- Réalisation de la SAT
- Conception du MCD
- Fin de la conception des maquettes

L'équipe a travaillé sur la structure de la base de données avec des analyses détaillées (SAT et MCD) pour garantir une organisation logique et efficace des données.

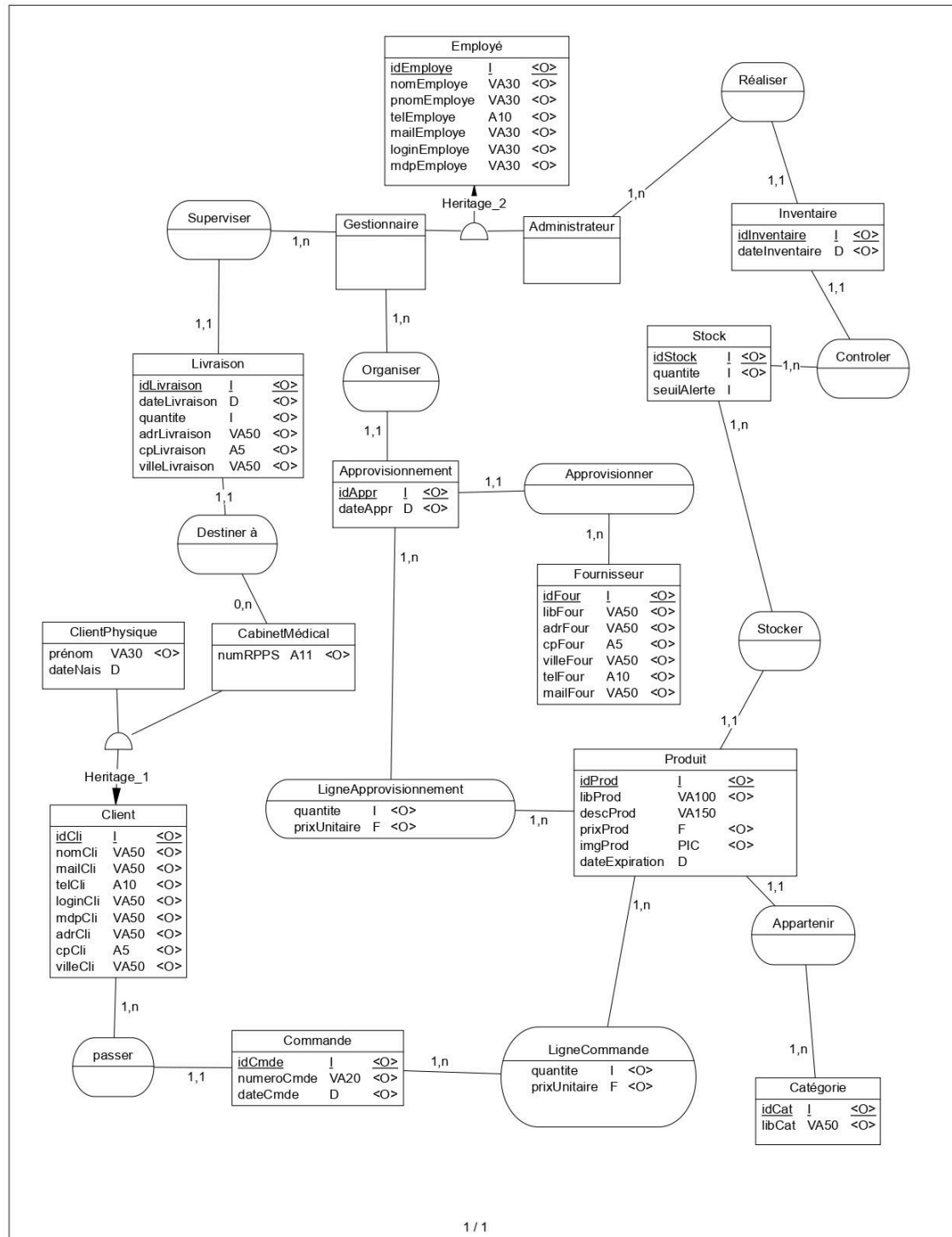
Tasks Board



Rendu

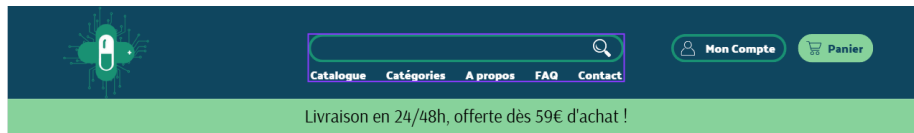
MCD :

SAE - Rapport de Gestion de Projet

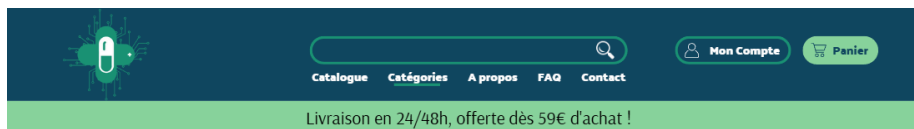
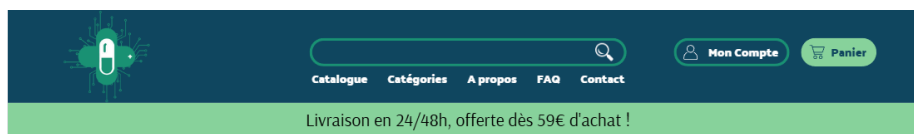


Maquettes du site :

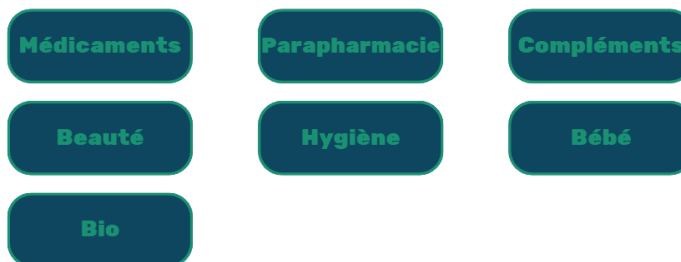
SAE - Rapport de Gestion de Projet



Nos produits conseils du mois



Nos catégories de produits



Mêlée 3

22 Octobre 2024

Objectif : Transformer le modèle conceptuel en MPD et élaborer le diagramme de classe.

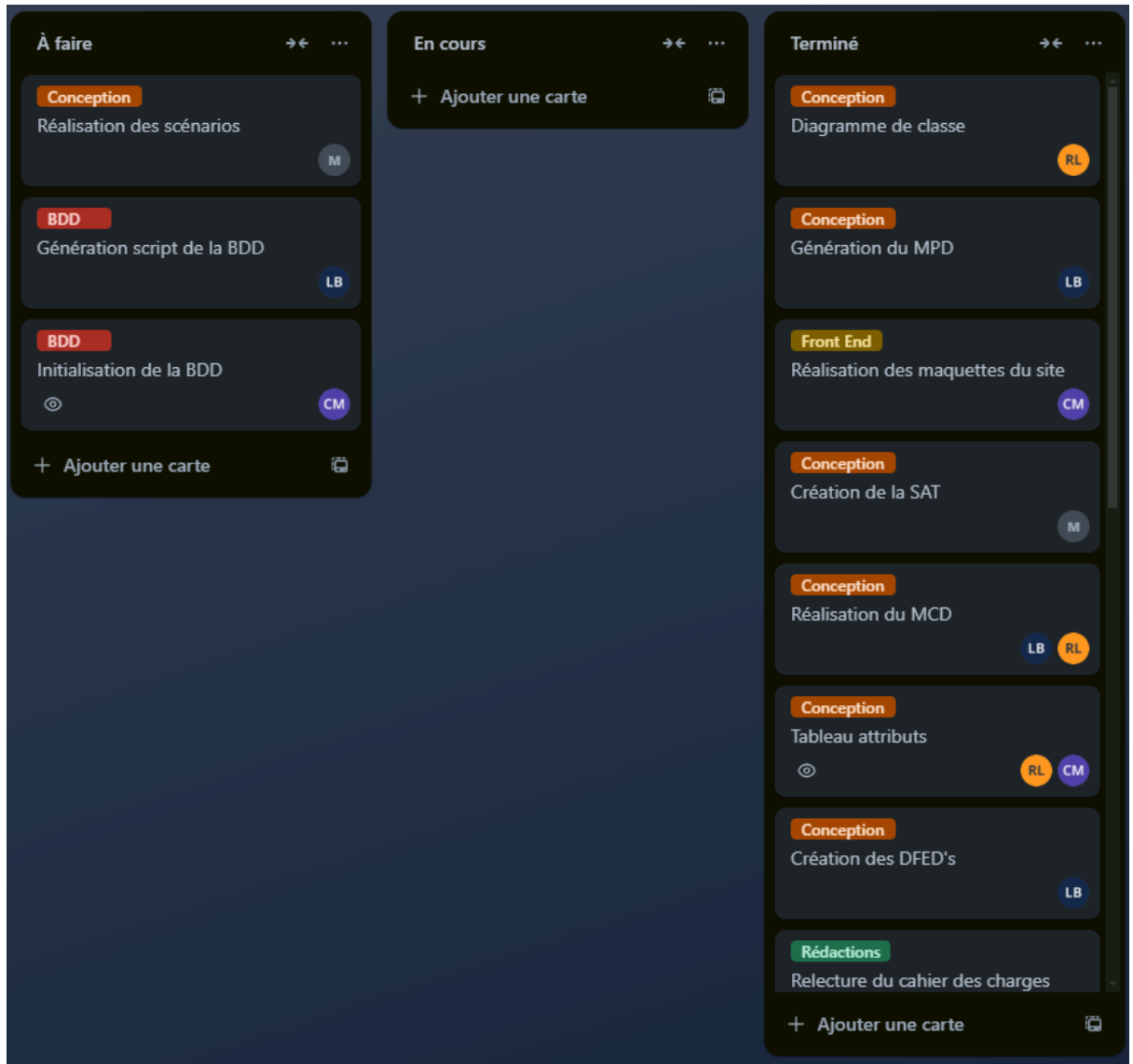
SAE - Rapport de Gestion de Projet

Résumé

- **Conversion du MCD en MPD**
- **Création du diagramme de classe**

Cette troisième mêlée a abouti à une structure physique de la base de données et un diagramme de classe détaillé pour l'implémentation.

Tasks Board



Rendu

MPD :

Diagramme de base de données relationnelles pour le système d'information de gestion de la pharmacie.

Tables et attributs :

- Employe**
 - idEmploye (INTEGER, pk, <fk>)
 - nomEmploye (VARCHAR2(30), not null)
 - prnomEmploye (VARCHAR2(30), not null)
 - mailEmploye (VARCHAR2(30), not null)
 - loginEmploye (VARCHAR2(30), not null)
 - mdpEmploye (VARCHAR2(30), not null)
- Gestionnaire**
 - idEmploye (INTEGER, pk, <fk>)
 - nomEmploye (VARCHAR2(30), not null)
 - prnomEmploye (VARCHAR2(30), not null)
 - mailEmploye (VARCHAR2(30), not null)
 - loginEmploye (VARCHAR2(30), not null)
 - mdpEmploye (VARCHAR2(30), not null)
- Administrateur**
 - idEmploye (INTEGER, pk, <fk>)
 - nomEmploye (VARCHAR2(30), not null)
 - prnomEmploye (VARCHAR2(30), not null)
 - mailEmploye (VARCHAR2(30), not null)
 - loginEmploye (VARCHAR2(30), not null)
 - mdpEmploye (VARCHAR2(30), not null)
- Fourisseur**
 - idfourn (INTEGER, pk, <fk>)
 - libfourn (VARCHAR2(50), not null)
 - adrFourn (VARCHAR2(50), not null)
 - cpFourn (CHAR(10), not null)
 - villfourn (VARCHAR2(50), not null)
 - maillfourn (CHAR(10), not null)
- ClientPhysique**
 - idCli (INTEGER, pk, <fk>)
 - nomCli (VARCHAR2(50), not null)
 - mailCli (VARCHAR2(50), not null)
 - CHAR(10) (CHAR(10), not null)
 - loginCli (VARCHAR2(50), not null)
 - mdpCli (VARCHAR2(50), not null)
 - adrCli (VARCHAR2(50), not null)
 - cpCli (CHAR(5), not null)
 - villCli (VARCHAR2(50), not null)
 - prnom (VARCHAR2(30), not null)
 - dateNaiss (DATE, not null)
- CabinetMedical**
 - idCli (INTEGER, pk, <fk>)
 - nomCli (VARCHAR2(50), not null)
 - mailCli (VARCHAR2(50), not null)
 - CHAR(10) (CHAR(10), not null)
 - loginCli (VARCHAR2(50), not null)
 - mdpCli (VARCHAR2(50), not null)
 - adrCli (VARCHAR2(50), not null)
 - cpCli (CHAR(5), not null)
 - villCli (VARCHAR2(50), not null)
 - numRPPS (CHAR(11), not null)
- Client**
 - idCli (INTEGER, pk, <fk>)
 - nomCli (VARCHAR2(50), not null)
 - mailCli (VARCHAR2(50), not null)
 - CHAR(10) (CHAR(10), not null)
 - loginCli (VARCHAR2(50), not null)
 - mdpCli (VARCHAR2(50), not null)
 - adrCli (VARCHAR2(50), not null)
 - cpCli (CHAR(5), not null)
 - villCli (VARCHAR2(50), not null)
- Appvisionnement**
 - idAppr (INTEGER, pk, <fk>)
 - idfourn (INTEGER, <fk>)
 - idEmploye (INTEGER, <fk>)
 - dateAppr (DATE, not null)
- LigneAppvisionnement**
 - idAppr (INTEGER, pk, <fk>)
 - idProd (INTEGER, pk, <fk>)
 - quantite (INTEGER, not null)
 - prixUnitaire (FLOAT, not null)
- Stock**
 - idStock (INTEGER, pk, <fk>)
 - quantite (INTEGER, not null)
 - seuilAlerte (INTEGER, not null)
- Produit**
 - idProd (INTEGER, pk, <fk>)
 - idCat (INTEGER, <fk>)
 - idStock (INTEGER, <fk>)
 - libProd (VARCHAR2(100), not null)
 - descProd (VARCHAR2(150), not null)
 - pratiProd (FLOAT, not null)
 - impProd (BLOB, not null)
 - dateExpiration (DATE, not null)
- Catégorie**
 - idCat (INTEGER, pk, <fk>)
 - libCat (VARCHAR2(50), not null)
- Commande**
 - idCmde (INTEGER, pk, <fk>)
 - idCli (INTEGER, <fk>)
 - numéroCmde (VARCHAR2(20), not null)
 - dateCmde (DATE, not null)
- LigneCommande**
 - idCmde (INTEGER, pk, <fk>)
 - idProd (INTEGER, pk, <fk>)
 - quantite (INTEGER, not null)
 - prixUnitaire (FLOAT, not null)
- Inventaire**
 - idInventaire (INTEGER, pk, <fk>)
 - idStock (INTEGER, <fk>)
 - idEmploye (INTEGER, <fk>)
 - dateInventaire (DATE, not null)

Contraintes de clé étrangère :

- idEmploye = idEmploye (Employe → Gestionnaire, Employe → Administrateur)
- idEmploye = idEmploye (Gestionnaire → Appvisionnement)
- idfourn = idfourn (Fourisseur → Appvisionnement)
- idAppr = idAppr (Appvisionnement → LigneAppvisionnement)
- idProd = idProd (LigneAppvisionnement → Produit)
- idCat = idCat (Produit → Catégorie)
- idProd = idProd (LigneCommande → Produit)
- idCmde = idCmde (LigneCommande → Commande)
- idStock = idStock (Stock → Inventaire)
- idCli = idCli (ClientPhysique → Client)
- idCli = idCli (CabinetMedical → Client)

```

classDiagram
    class Categorie {
        -nomCategorie
    }
    class LigneCommande {
        -quantite : int
        -sousTotal : float
        +verifierDisponibilite() bool
        +verifierExpiration() bool
    }
    class Commande {
        <<Enum>>
        -En commande
        -Livré
        -Annulé
    }
    class Client {
        +chercherProduit(nom : string) : Amas<Medicament>
        +passerCommande(commande : commande)
    }
    class Gestionnaire {
        +gererStock()
        +traiterCommande(commande : commande)
    }
    class Administrateur {
        +faireInventaire() : Amas<Medicament>
        +consulterProduitsACommander() : Amas<Medicament>
    }
    class Medicament {
        -nomMed : string
        -descriptionMed : string
        -quantiteMed : int
        -dateExpirationMed : Date
        -prixMed : float
        +verifierDisponibilite() bool
        +verifierExpiration() bool
    }
    class Approvisionnement {
        -dateApprovisionnement : Date
        -Quantite : int
        +MettreAJourStock() : int
    }
    class Livraison {
        -dateLivraison : Date
        -cheminOptimal : string
        +calculerCheminOptimal() : string
    }
    class Cabinet {
        -nomCabinet : string
        -adresseCabinet
        +RecevoirLivraison(livraison : Livraison)
    }
    class Utilisateur {
        <<Abstract>>
        -nomUtilisateur : string
        -emailUtilisateur : string
        +seConnecter() : bool
        +seDeconnecter() : bool
    }
    class Log {
        -nomSite : string
        -adresseSite : string
    }
    class LigneApprovisionnement {
        -quantite : int
        -prixUnitaire : float
        +verifierDisponibilite() bool
    }

    Categorie "1" -- "0..*" Medicament : -les categories
    Medicament "0..*" -- "0..*" LigneCommande : -les commandes
    LigneCommande "0..*" -- "0..*" Commande : -les commandes
    Commande "0..*" -- "0..*" Client : -les commandes
    Client "1" -- "0..*" Commande : -les commandes
    Gestionnaire "1" -- "0..*" Commande : -les commandes
    Administrateur "1" -- "0..*" Commande : -les commandes
    Medicament "0..*" -- "0..*" Approvisionnement : -les approvisionnements
    Approvisionnement "0..*" -- "0..*" Livraison : -les livraisons
    Livraison "0..*" -- "1" Cabinet : recevoir
    Cabinet "1" -- "0..*" Livraison : -les livraisons
    Utilisateur <|-- Gestionnaire
    Utilisateur <|-- Administrateur
    Utilisateur "0..*" -- "1" Log : Associer
    Log "1" -- "0..*" LigneApprovisionnement : < Gérer
    LigneApprovisionnement "0..*" -- "0..*" Medicament : -les medicaments
    LigneApprovisionnement "0..*" -- "0..*" Approvisionnement : -les approvisionnements
    
```

The diagram illustrates the structure of a pharmacy management system. It includes classes for categories, orders, clients, managers, administrators, medications, supplies, deliveries, cabinets, users, and logs. Relationships are defined through associations, generalizations, and compositions, with multiplicity and role names indicated on the association lines.

02 Novembre 2024

SAE - Rapport de Gestion de Projet

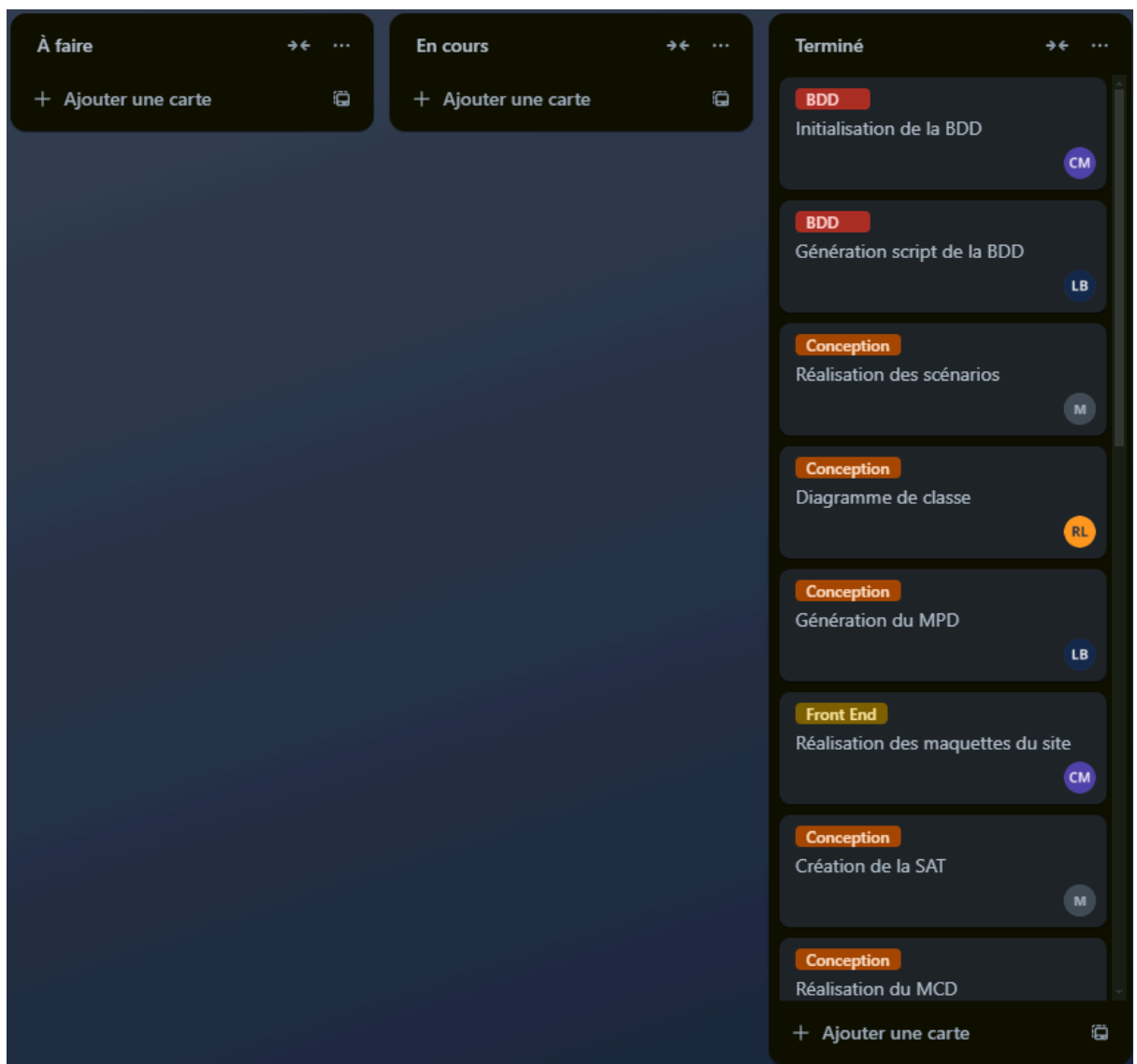
Objectif : Finaliser la préparation technique en rédigeant les scénarios et en générant le script SQL de la base de données.

Résumé

- **Création des scénarios**
- **Génération du script SQL**
- **Initialisation de la BDD**

Cette dernière mûlée a permis de préparer la base de données pour l'utilisation et de valider les parcours utilisateurs principaux.

Tasks Board



Rendu

Extraits du script de la BDD :

SAE - Rapport de Gestion de Projet

```
/*=====*/
/* Table : Administrateur */
/*=====*/
create table Administrateur
(
    idEmploye          INTEGER          not null,
    nomEmploye         VARCHAR2(30)     not null,
    pnomEmploye        VARCHAR2(30)     not null,
    telEmploye         CHAR(10)         not null,
    mailEmploye        VARCHAR2(30)     not null,
    loginEmploye       VARCHAR2(30)     not null,
    mdpEmploye         VARCHAR2(30)     not null,
    constraint PK_ADMINISTRATEUR primary key (idEmploye)
);

/*=====*/
/* Table : Approvisionnement */
/*=====*/
create table Approvisionnement
(
    idAppr            INTEGER          not null,
    idFour            INTEGER          not null,
    idEmploye         INTEGER          not null,
    dateAppr          DATE             not null,
    constraint PK_APPROVISIONNEMENT primary key (idAppr)
);
```

```
alter table LigneApprovisionnement
    add constraint FK_LIGNEAPP_LIGNEAPPR_APPROVIS foreign key (idAppr)
        references Approvisionnement (idAppr);

alter table LigneApprovisionnement
    add constraint FK_LIGNEAPP_LIGNEAPPR_PRODUIT foreign key (idProd)
        references Produit (idProd);

alter table LigneCommande
    add constraint FK_LIGNECOM_LIGNECOMM_COMMANDE foreign key (idCmde)
        references Commande (idCmde);

alter table LigneCommande
    add constraint FK_LIGNECOM_LIGNECOMM_PRODUIT foreign key (idProd)
        references Produit (idProd);

alter table Livraison
    add constraint FK_LIVRAISO_DESTINER__CABINETM foreign key (idCli)
        references CabinetMedical (idCli);

alter table Livraison
    add constraint FK_LIVRAISO_SUPERVISE_GESTIONN foreign key (idEmploye)
        references Gestionnaire (idEmploye);

alter table Produit
    add constraint FK_PRODUIT_APPARTENI_CATEGORI foreign key (idCat)
        references Categorie (idCat);

alter table Produit
    add constraint FK_PRODUIT_STOCKER_STOCK foreign key (idStock)
        references Stock (idStock);
```