GESTION D’UN STOCK À DISTANCE :

1.TRAVAIL COLLABORATIF (Une seule et même équipe)

2.Mise en place d’une stratégie Git.

* Création d’un nouveau repo GitHub ;
* Configuration d’un accès pour les différents collaborateurs.
* Installer Git Lens
* Test des fonctionnalités Git sur chaque poste.

3.Le scénario

Un client souhaite pouvoir consulter ses stocks en ligne. Il imagine une interface composée de **4 éléments principaux :**

1. Un volet d’administration pour :

-vider toute la collection (avec une demande de confirmation)

-avoir la possibilité d’afficher ou pas dans la table les champs suivants :

* Nom (obligatoire, sans possibilité de ne pas l’afficher ce champs)
* Référence (obligatoire, sans possibilité de ne pas l’afficher ce champs)
* Quantité (obligatoire, sans possibilité de ne pas l’afficher ce champs)
* Catégorie (optionnel)
* Sous-catégorie (optionnel)
* Prix (optionnel)
* Date de création (optionnel)

-

-créer un nouvel utilisateur (avec des droits d’accès)

-changer de collection

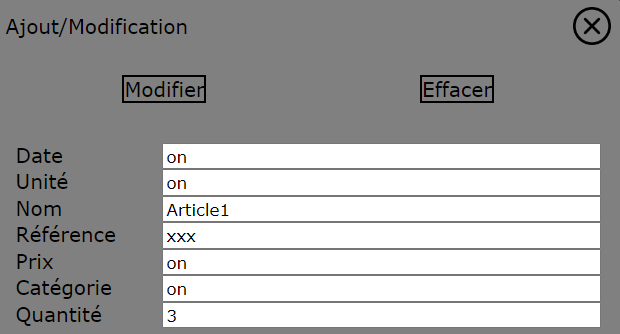
2. Un conteneur pour afficher les différents champs de saisie

* Ce conteneur n’affiche que les champs de saisie sélectionnés dans le volet d’administration.
* Par défaut seul Nom, Référence et Quantité sont affichés (obligatoires)
* Un bouton de validation permet d’inscrire les données saisies sur la base de données.
* Les données sont contrôlées avant envoi et un champ vide obligatoire empêche l’envoi.

3.Une table d’affichage.

* Par défaut la table n’affiche que l’entête définie par le volet administration.
* A la création d’un nouvel article (après avoir cliqué sur “valider”), une nouvelle ligne apparaît sur la table.
* La ligne créée est cliquable.
* Au clique, un panneau apparaît permettant de modifier ou effacer la ligne en cours.
* Même si la table est paramétrée pour n’afficher qu’un certain nombre de champs, le panneau doit afficher toutes les valeurs de toutes les propriétés de l’objet.

EXEMPLE …



* Il est possible de trier la table en cliquant sur un des champs

**IDENTIFICATION**

Les utilisateurs doivent pouvoir créer un compte, s’authentifier pour accéder à l’application.

4.MISE EN OEUVRE

1-Réaliser une première ébauche avec Figma

2-Définir les composants essentiels de l’application.

3-Définir les fonctionnalités de l’application.

4.Créer des fichiers distincts pour héberger un type de composant particulier :

* Un fichier pour les variables et les constantes globale
* Commenter chaque ligne de code importante
* Un fichier pour les fonctions de construction (création des conteneurs et sous conteneurs, ajout de ligne à la table etc.)
* Un fichier pour les fonctions de gestion (lecture, écriture, modification, suppression de documents sur la base de données, contrôle saisie utilisateurs, tri des champs, contrôle d’authentification etc…)

5.Répartir la création des composants entre les différents utilisateurs.

* Créer des fonctions qui renvoient un élément HTML paramétré et spécifique.

Exemple : fonction créerBouton = (nom,id,callback) {return élémentHtml} …

* Créer un conteneur pour les champs de saisi et y placer des composants input paramétrés (nombre ou texte, type de nombre etc…) avec la technique ci-dessus
* Utiliser git pour mettre le projet à jour
* Créer des fonctions spécifiques à un cas précis d’utilisation (isoler jusqu’au plus petit mécanisme et faire en sorte que les fonctions soient réutilisables
* Nommer les fonctions et les variables de la façon la plus intelligible possible.

Une variable globale de type tableau de caractère doit s'identifier telle que VGAS\_ (variable globale array string).

* Créer le fichier css, implémenter le responsive, créer dossier images
* Créer le composant table

Composants de construction :

-champ de saisie

-bouton

-bouton de fermeture

-conteneur principal

-conteneur authentification

-conteneur administration

-conteneur affichage info sur la ligne

-conteneur table

-conteneur message à l’utilisateur

Composants fonctionnels

-café Mathis

-authentification d’un utilisateur

-création de compte

-récupérer la configuration de l’affichage

-créer une configuration d’affichage

-afficher la collection dans la table

-mettre à jour la table

-vider ou détruire un conteneur

-vider un tableau

-vider une collection

-récupérer la valeur des champs de saisi

-construire un objet avec la valeur et l’id des champs de saisie

-sauvegarder un enregistrement sur la BDD

-trier les champs de saisie (clique sur les champs de la table)

-contrôler la saisie utilisateur

-remplir un conteneur d’élément

-forcer l’affichage que d’un seul panneau à la fois (info ou administration)

-changer les attributs d’un élément

-modifier les valeurs des propriétés dans un tableau d’objet

-afficher message d’erreur pour l’utilisateur

-réinitialiser l’application

-changer de collection

-déconnecter utilisateur

-mettre à jour les champs à afficher choisis par l’utilisateur

-permettre ou non d’accéder au panneau d’administration

-cacher/montrer un panneau (administration, info, log, message à l’utilisateur)