Guilliam MORISSET

Usages IA – Mini projet

Choix des IA

- **Gemini PRO** : je lui ai donné ce que je voulais dans mon app (specs + skills) et il m'a construit les contextes pour que ce soit clair et détaillé pour Copilot
  Pourquoi Gemini ? Parce que je trouve qu'il est efficace dans la génération de specs et ça permets d'avoir un assistant moins "focus code".

- **GitHub Copilot** : à chaque nouvelle fonctionnalité je crée un nouveau chat en redonnant tout le contexte (skills, fichiers html/css/js...) pour éviter la pollution du contexte et économiser les tokens
  Pourquoi Copilot ? Car il est directement intégré à VS Code, ça permet de gagner en productivité et permet d'utiliser **Claude Haïku 4.5**

Choix des skills

- web-design-guidelines by vercel-labs/agent-skills : ayant uniquement des bases faibles en design web je ne suis pas trop en capacité d'orienter Copilot, donc ce skill permet de faire ce job à ma place

npx skills add https://github.com/vercel-labs/agent-skills --skill web-design-guidelines

- code-reviewer by google-gemini/gemini-cli : étant donné que c'est un projet en solo, personne ne relit le code, donc ça permet qu'il applique directement les bonnes pratiques et que s'il y a un problème il puisse avoir des guidelines pour refactoriser/corriger

npx skills add https://github.com/google-gemini/gemini-cli --skill code-reviewer

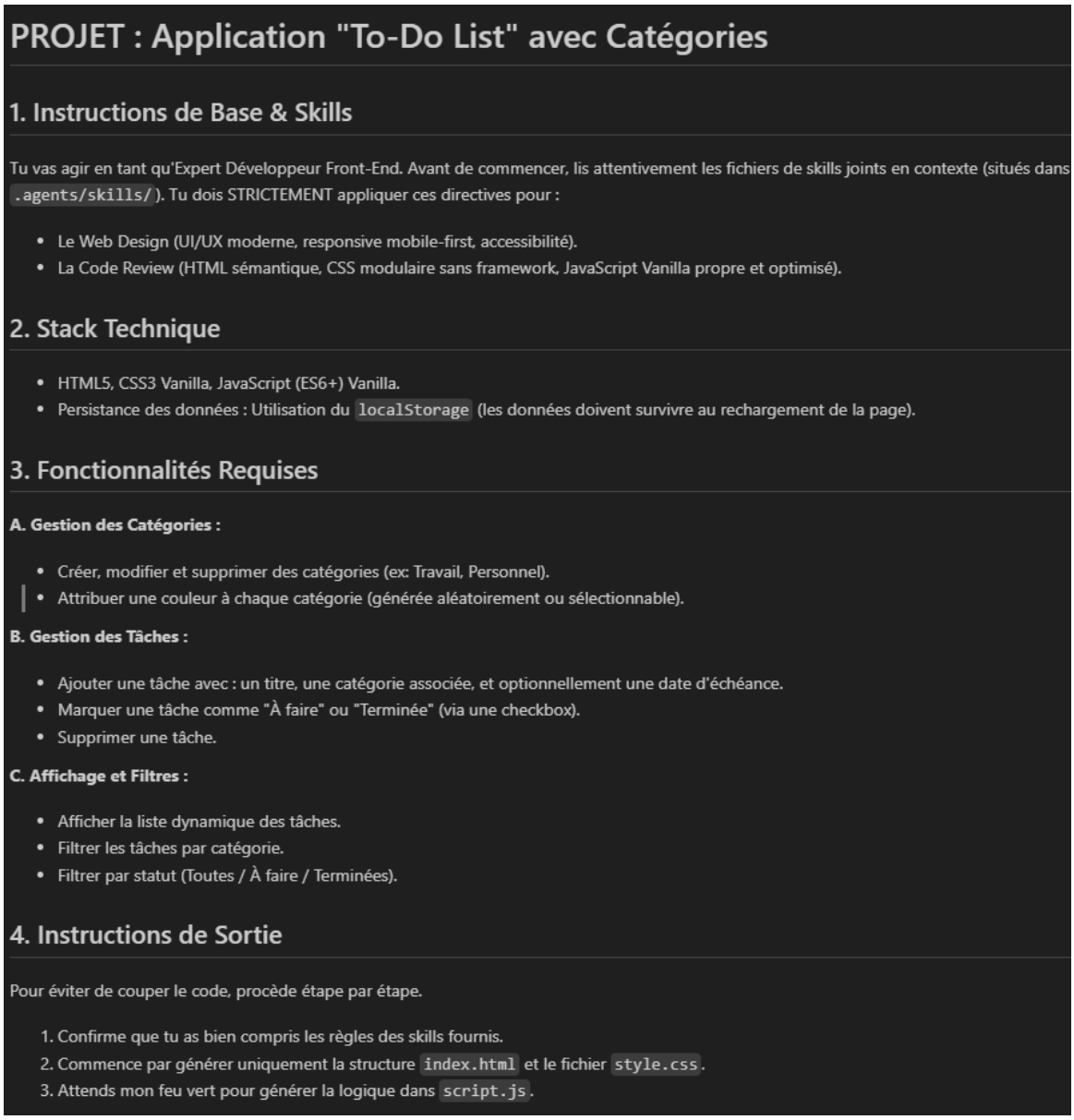
Démarrage du projet

La consigne étant uniquement de faire une app de TO-DO List, j'ai demandé à Gemini de me créer un fichier de contexte avec mes prérequis qui étaient :

"Je dois faire une application TO DO LIST (tâches/catégories) full IA, peux-tu me faire mon fichier de contexte à passer à une IA. Mon projet doit être en HTML/CSS/JS,

avec les skills web-design-guidelines et code-reviewer. Mon site doit être responsive et design”

Il m’a généré le fichier de contexte ci-dessous plutôt complet que j’ai joint à toutes mes requêtes Copilot.



PROJET : Application "To-Do List" avec Catégories

1. Instructions de Base & Skills

Tu vas agir en tant qu'Expert Développeur Front-End. Avant de commencer, lis attentivement les fichiers de skills joints en contexte (situés dans `.agents/skills/`). Tu dois STRICTEMENT appliquer ces directives pour :

- Le Web Design (UI/UX moderne, responsive mobile-first, accessibilité).
- La Code Review (HTML sémantique, CSS modulaire sans framework, JavaScript Vanilla propre et optimisé).

2. Stack Technique

- HTML5, CSS3 Vanilla, JavaScript (ES6+) Vanilla.
- Persistance des données : Utilisation du `localStorage` (les données doivent survivre au rechargement de la page).

3. Fonctionnalités Requises

A. Gestion des Catégories :

- Créer, modifier et supprimer des catégories (ex: Travail, Personnel).
- Attribuer une couleur à chaque catégorie (générée aléatoirement ou sélectionnable).

B. Gestion des Tâches :

- Ajouter une tâche avec : un titre, une catégorie associée, et optionnellement une date d'échéance.
- Marquer une tâche comme "À faire" ou "Terminée" (via une checkbox).
- Supprimer une tâche.

C. Affichage et Filtres :

- Afficher la liste dynamique des tâches.
- Filtrer les tâches par catégorie.
- Filtrer par statut (Toutes / À faire / Terminées).

4. Instructions de Sortie

Pour éviter de couper le code, procède étape par étape.

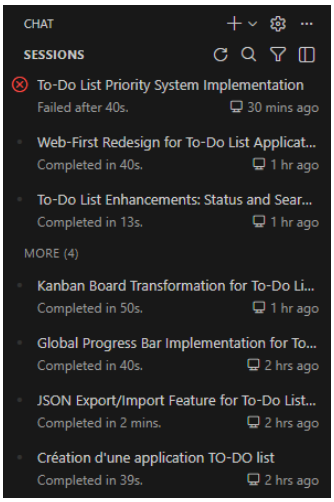
1. Confirme que tu as bien compris les règles des skills fournis.
2. Commence par générer uniquement la structure `index.html` et le fichier `style.css`.
3. Attends mon feu vert pour générer la logique dans `script.js`.

Fonctionnalités

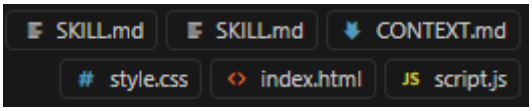
- V1.0.0 : Initialisation des skills, du contexte et MVP de l'app
- V1.1 : Export/import JSON
- V1.2 : Barre de progression pour gamification
- V1.3 : Ajout du status en cours et recherche textuelle
- V1.4 : Front en web first
- V1.5 : Ajout priorité sur les tâches

J’ai essayé de faire un tableau Kanban en v2.0.0 mais ma base de code étant trop solide Copilot n’a pas réussi à faire quelque chose de fonctionnel.

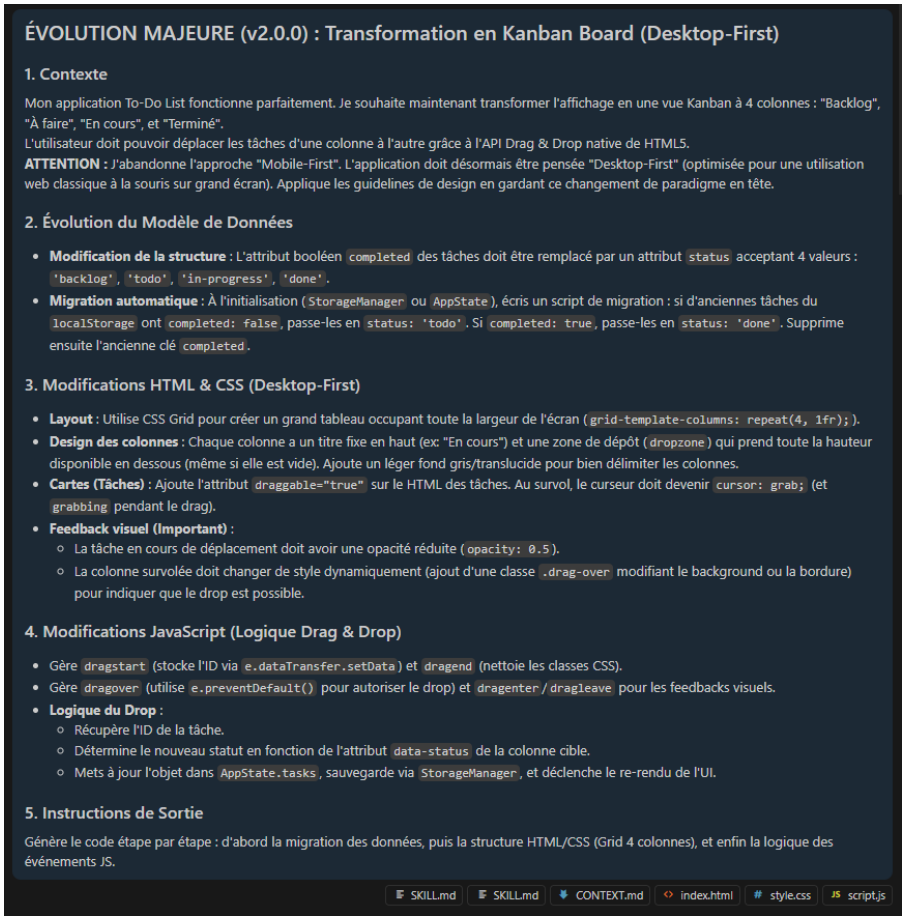
A chaque nouvelle fonctionnalité, je recréais un nouveau chat Copilot pour éviter la pollution du contexte et économiser les tokens :



Je joignais tous les fichiers de contexte pour qu’il puisse repartir sur de bonnes bases :



Sans oublier un contexte détaillé de la fonctionnalité attendue :



ÉVOLUTION MAJEURE (v2.0.0) : Transformation en Kanban Board (Desktop-First)

1. Contexte

Mon application To-Do List fonctionne parfaitement. Je souhaite maintenant transformer l'affichage en une vue Kanban à 4 colonnes : "Backlog", "À faire", "En cours", et "Terminé".
L'utilisateur doit pouvoir déplacer les tâches d'une colonne à l'autre grâce à l'API Drag & Drop native de HTML5.
ATTENTION : J'abandonne l'approche "Mobile-First". L'application doit désormais être pensée "Desktop-First" (optimisée pour une utilisation web classique à la souris sur grand écran). Applique les guidelines de design en gardant ce changement de paradigme en tête.

2. Évolution du Modèle de Données

- **Modification de la structure** : L'attribut booléen `completed` des tâches doit être remplacé par un attribut `status` acceptant 4 valeurs : `'backlog'`, `'todo'`, `'in-progress'`, `'done'`.
- **Migration automatique** : À l'initialisation (`StorageManager` ou `AppState`), écris un script de migration : si d'anciennes tâches du `localStorage` ont `completed: false`, passe-les en `status: 'todo'`. Si `completed: true`, passe-les en `status: 'done'`. Supprime ensuite l'ancienne clé `completed`.

3. Modifications HTML & CSS (Desktop-First)

- **Layout** : Utilise CSS Grid pour créer un grand tableau occupant toute la largeur de l'écran (`grid-template-columns: repeat(4, 1fr);`).
- **Design des colonnes** : Chaque colonne a un titre fixe en haut (ex: "En cours") et une zone de dépôt (`dropzone`) qui prend toute la hauteur disponible en dessous (même si elle est vide). Ajoute un léger fond gris/translucide pour bien délimiter les colonnes.
- **Cartes (Tâches)** : Ajoute l'attribut `draggable="true"` sur le HTML des tâches. Au survol, le curseur doit devenir `cursor: grab;` (et `grabbing` pendant le drag).
- **Feedback visuel (Important)** :
  - La tâche en cours de déplacement doit avoir une opacité réduite (`opacity: 0.5`).
  - La colonne survolée doit changer de style dynamiquement (ajout d'une classe `.drag-over` modifiant le background ou la bordure) pour indiquer que le drop est possible.

4. Modifications JavaScript (Logique Drag & Drop)

- Gère `dragstart` (stocke l'ID via `e.dataTransfer.setData`) et `dragend` (nettoie les classes CSS).
- Gère `dragover` (utilise `e.preventDefault()` pour autoriser le drop) et `dragenter` / `dragleave` pour les feedbacks visuels.
- **Logique du Drop** :
  - Récupère l'ID de la tâche.
  - Détermine le nouveau statut en fonction de l'attribut `data-status` de la colonne cible.
  - Mets à jour l'objet dans `AppState.tasks`, sauvegarde via `StorageManager`, et déclenche le re-rendu de l'UI.

5. Instructions de Sortie

Génère le code étape par étape : d'abord la migration des données, puis la structure HTML/CSS (Grid 4 colonnes), et enfin la logique des événements JS.

SKILL.md | SKILL.md | CONTEXT.md | index.html | style.css | script.js

Conclusion

Ce qui a bien marché

- Génération des contextes via Gemini
- Génération des fonctionnalités simples via Copilot
- Utilisation des skills
- Un chat unique par fonctionnalité

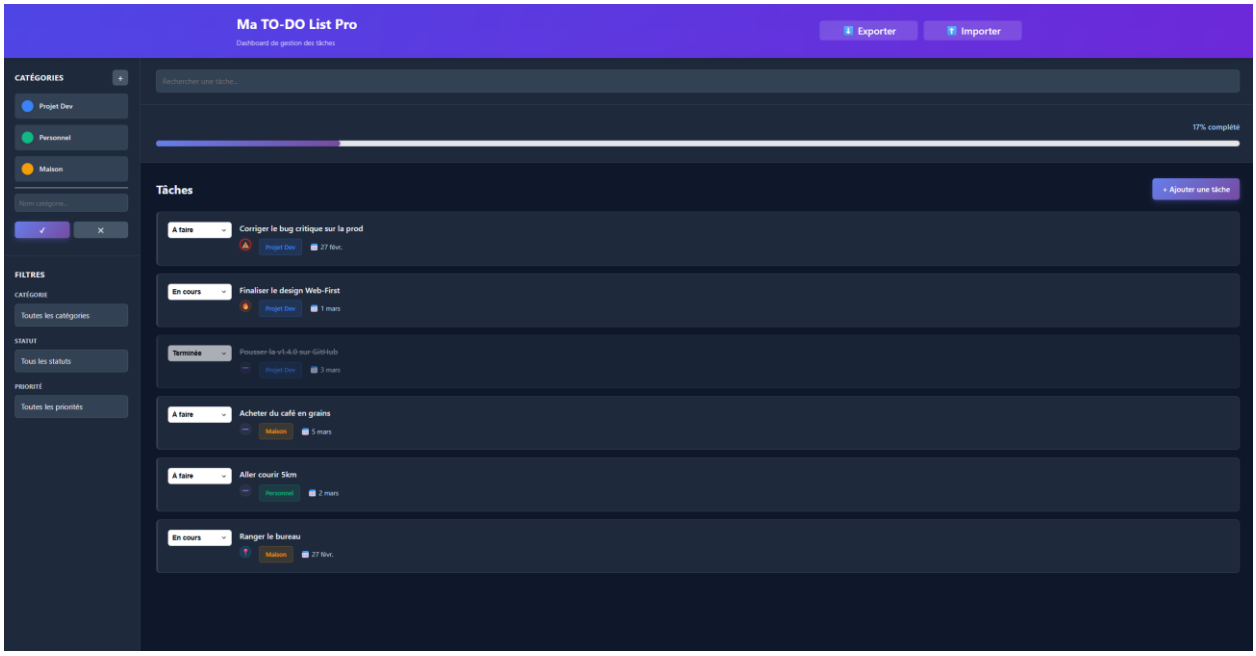
Ce qui a moins bien marché

- Génération des fonctionnalités devant refactoriser la quasi-totalité du code par Copilot (tableau Kanban)

Limites

Quand une fonctionnalité touche à trop de dépendances et de parties de code, la refactorisation est compliquée et peut engendrer une perte de temps plus importante que le gain de temps espéré.

Résultat


Ma TO-DO List Pro
Dashboard de gestion des tâches
Exporter
Importer
CATÉGORIES
Projet Dev
Personnel
Maison
Nom catégorie...
FILTRES
CATÉGORIE
Toutes les catégories
STATUT
Tous les statuts
PRIORITÉ
Toutes les priorités
Rechercher une tâche...
17% complété
Tâches
+ Ajouter une tâche
A faire
Corriger le bug critique sur la prod
Projet Dev
27 févr.
En cours
Finaliser le design Web-First
Projet Dev
1 mars
Terminée
Pousser la v1.4.0 sur GitHub
Projet Dev
3 mars
A faire
Acheter du café en grains
Maison
5 mars
A faire
Aller courir 5km
Personnel
2 mars
En cours
Ranger le bureau
Maison
27 févr.