Absolument. Voici la synthèse de notre session de travail, structurée selon vos instructions pour servir de "Document d'État du Projet".

Markdown

---

Projet: "Usine à Tests"

Date: "2508151000"

Version: "10.0"

---

### 1. Résumé des Échanges du Jour

La session d'aujourd'hui a marqué un tournant majeur, passant de la conception architecturale à l'industrialisation. Partant de la finalisation de la logique de lecture de la feuille `sys\_Composition\_Emails`, la session a évolué vers la création d'un \*\*générateur de structures de tests\*\*, accélérant considérablement le déploiement de nouveaux tests.

Les points clés sont :

1. \*\*Validation de la Logique de Données :\*\* Nous avons solidifié la convention pour la colonne `Code\_Niveau\_Email`, en adoptant une approche explicite (`"N1,N2,N3"`) pour gérer les contenus partagés entre différents niveaux d'e-mails, rejetant une approche de "fallback" implicite.

2. \*\*Clarification des `Code\_Profil` :\*\* Nous avons confirmé l'utilisation de codes spécifiques pour les tests "Couleurs" (R, J, V, B) et "ANCRES" (`EXPERTISE`, `MANAGEMENT`, etc.) afin d'assurer la compatibilité avec l'existant.

3. \*\*Création d'un Générateur de Structure :\*\* Le point central de la session a été la création et l'utilisation d'un script pour générer automatiquement le "boilerplate" (la structure de base) de la configuration des e-mails pour de nouveaux tests.

4. \*\*Génération des Premiers Tests :\*\* Nous avons appliqué ce générateur pour produire les structures complètes pour les tests \*\*MBTI\*\* (132 lignes), \*\*Couleurs\*\* (28 lignes), et \*\*ANCRES\*\* (52 lignes), les livrant sous forme de fichiers CSV prêts à l'import.

5. \*\*Gestion de la Complexité :\*\* Nous avons validé que la flexibilité de notre architecture implique qu'un contenu plus spécifique (par ex., des textes différents par niveau d'e-mail) se traduit logiquement par un plus grand nombre de lignes dans la base de données, ce qui est le comportement attendu.

### 2. État Actuel du Projet

Le projet a dépassé la simple conception pour entrer dans une phase d'outillage et de production de données.

\* \*\*Tâches réalisées :\*\*

\* Finalisation des règles de gestion pour la composition d'e-mails multi-niveaux et multi-profils.

\* Développement d'un processus automatisé pour générer les structures de données de base pour les tests.

\* Génération des fichiers de structure CSV pour 3 tests pilotes (MBTI, Couleurs, ANCRES).

\* \*\*Tâches en cours :\*\* Le projet est en attente du peuplement de la base de données à l'aide des fichiers CSV générés.

\* \*\*Prochaines étapes planifiées :\*\*

1. Importer les fichiers CSV dans l'onglet `sys\_Composition\_Emails`.

2. Remplir la colonne `Contenu / ID\_Document` pour au moins un test afin de disposer d'un cas d'usage complet.

3. Développer la fonction Google Apps Script qui lira et exploitera ces données.

### 3. Contexte Historique du Projet

Partant d'une vision de plateforme de tests entièrement pilotée par la donnée, les sessions précédentes avaient permis d'universaliser le calcul des scores et de concevoir l'architecture de composition des e-mails. La session actuelle a fait un bond en avant en \*\*automatisant la création de cette architecture de données\*\*. Nous ne nous contentons plus de définir comment la "machinerie" de l'usine fonctionne, nous avons commencé à construire les machines qui produisent les pièces elles-mêmes.

### 4. Orientations Stratégiques

Cette session a massivement renforcé les principes directeurs du projet : \*\*"La Base de Données est la Source Unique de Vérité"\*\* et \*\*"Zéro Convention Implicite"\*\*.

\* La décision d'utiliser des listes explicites (`"N1,N2,N3"`) dans la colonne `Code\_Niveau\_Email` est une application directe du principe "Zéro Convention Implicite".

\* Le "Journal des Décisions d'Architecture" s'enrichit d'un nouveau pattern majeur :

\* \*\*Pattern : Générateur de Structure (Boilerplate) :\*\* "Pour accélérer le déploiement d'entités complexes (comme un nouveau test), on ne les configure pas manuellement. On développe un script qui génère une structure de données de base ('boilerplate') valide et complète, en se basant sur un set de paramètres. L'utilisateur n'a plus qu'à remplir le contenu métier, garantissant la cohérence structurelle et éliminant les erreurs de saisie."

### 5. Architecture Technique & Ressources Clés

L'accent a été mis sur la production des ressources de données.

#### 5.1. État des Scripts et Fichiers

Aucun script du projet final n'a été modifié. Cependant, un script externe (Python) a été développé et utilisé pour générer les fichiers de données.

\* \*\*Script de Génération (Exemple pour "ANCRES") :\*\*

```python

import pandas as pd

# 1. Définition des paramètres du test

test\_name = "ANCRES"

profiles = [

"EXPERTISE", "MANAGEMENT", "AUTONOMIE", "SECURITE",

"CREATION", "SERVICE", "DEFI", "EQUILIBRE"

]

languages = ["FR", "EN"]

column\_names = [

"Type\_Test", "Code\_Langue", "Code\_Niveau\_Email", "Code\_Profil",

"Element", "Ordre", "Contenu / ID\_Document"

]

# 2. Logique de génération (hypothèse : 1 intro, 1 doc commun, 2 textes et 1 doc spécifique par profil)

data = []

# Contenus communs

for lang in languages:

data.append({"Type\_Test": test\_name, "Code\_Langue": lang, "Code\_Niveau\_Email": "N1,N2,N3", "Code\_Profil": "", "Element": "Introduction", "Ordre": 1, "Contenu / ID\_Document": ""})

data.append({"Type\_Test": test\_name, "Code\_Langue": lang, "Code\_Niveau\_Email": "N1,N2,N3", "Code\_Profil": "", "Element": "Document", "Ordre": 100, "Contenu / ID\_Document": ""})

# Contenus spécifiques

for lang in languages:

for profile in profiles:

data.append({"Type\_Test": test\_name, "Code\_Langue": lang, "Code\_Niveau\_Email": "N1,N2,N3", "Code\_Profil": profile, "Element": "Corps\_Texte", "Ordre": 10, "Contenu / ID\_Document": ""})

data.append({"Type\_Test": test\_name, "Code\_Langue": lang, "Code\_Niveau\_Email": "N1,N2,N3", "Code\_Profil": profile, "Element": "Corps\_Texte", "Ordre": 20, "Contenu / ID\_Document": ""})

data.append({"Type\_Test": test\_name, "Code\_Langue": lang, "Code\_Niveau\_Email": "N1,N2,N3", "Code\_Profil": profile, "Element": "Document", "Ordre": 110, "Contenu / ID\_Document": ""})

# 3. Création du fichier CSV

df = pd.DataFrame(data, columns=column\_names)

df.to\_csv("structure\_ANCRES\_corrigee.csv", index=False, sep=';')

print("Fichier généré avec succès.")

**5.2. Architecture Détaillée et Traçabilité**

Fichier : **[BDD] Tests & Profils**

* **MODIFIÉ (Méthodologie)** :
  + L'onglet **sys\_Composition\_Emails** n'est plus destiné à être rempli manuellement ligne par ligne, mais à être peuplé en masse via l'import de structures pré-générées.
* **AJOUTÉ (Ressources Externes)** :
  + Fichier structure\_MBTI.csv (132 lignes)
  + Fichier structure\_Couleurs.csv (28 lignes)
  + Fichier structure\_ANCRES\_corrigee.csv (52 lignes)

**5.3. Nomenclature des Variables**

* **Code\_Niveau\_Email (String) :**
  + **Contenu Attendu :** Une chaîne de caractères contenant un ou plusieurs niveaux d'e-mail, séparés par des virgules (ex: "N1", "N1,N2", "N1,N2,N3").
  + **Rôle :** Permet de lier une brique de contenu à un ou plusieurs niveaux d'e-mail de manière explicite.

**5.4. Structure des Données (Feuilles et Colonnes)**

* **Onglet : sys\_Composition\_Emails**
  + **Colonnes critiques :** Type\_Test, Code\_Langue, Code\_Niveau\_Email, Code\_Profil, Element, Ordre, Contenu / ID\_Document. (Structure inchangée mais règles de remplissage solidifiées).

**6. Actions Recommandées / Prochaines Étapes**

1. **Action Immédiate (Pour vous) :** Dans le Google Sheets [BDD] Tests & Profils, importer les données des trois fichiers CSV (structure\_MBTI.csv, structure\_Couleurs.csv, structure\_ANCRES\_corrigee.csv) dans l'onglet sys\_Composition\_Emails.
2. **Action Suivante (Pour vous) :** Choisir un des tests (par ex. "Couleurs") et commencer à remplir la colonne Contenu / ID\_Document pour disposer d'un cas pratique complet pour le développement.
3. **Question pour notre prochaine session :** Une fois les données d'au moins un test complétées, nous serons prêts à coder. La question sera : **Par quelle partie du script de lecture de sys\_Composition\_Emails souhaitez-vous commencer ? (Ex: la récupération et les filtres initiaux, ou l'assemblage final du contenu ?)**