Absolument. Voici le point d'étape complet, préparé selon votre prompt et intégrant la synthèse de notre session ainsi que les nouvelles informations pour la suite.

Markdown

---

Projet: Usine à Tests

Date: 2025-07-07

Version: 9.0 (MBTI Implémenté)

---

### 1. Résumé des Échanges du Jour

La session a été dédiée à l'implémentation et au débogage du module de logique pour le test \*\*MBTI\*\*, ce qui a permis de tester en conditions réelles la robustesse de l'architecture.

\* \*\*Développement Initial :\*\* Nous avons développé une première version du script `Logique\_MBTI.gs`.

\* \*\*Débogage en Plusieurs Étapes :\*\*

1. \*\*Bug de l'E-mail Vide :\*\* Le premier test a produit un e-mail de résultats vide. Le diagnostic a montré que la fonction `buildAndSendEmails` dans `Utilities.gs` ne possédait pas de cas pour le type `MBTI`. Nous avons corrigé `Utilities.gs` pour y ajouter la logique de mise en forme nécessaire.

2. \*\*Bug de Retraitement Multilingue :\*\* Nous avons découvert que le retraitement d'une réponse dans une autre langue échouait (scores à zéro, profil par défaut "ESTJ"). Le diagnostic a révélé que le script comparait des réponses textuelles dans une langue avec des options dans une autre.

3. \*\*Bug Final de Logique :\*\* Après plusieurs tentatives de correction, nous avons identifié la cause racine de l'échec de tous les calculs : une supposition erronée sur le format des titres de questions. Nous avons confirmé que le `MOTEUR` préfixe les titres avec leur ID (ex: `"Q01: Titre..."`). La version finale de `Logique\_MBTI.gs` a été corrigée pour parser ce format, ce qui a résolu le problème de manière définitive.

\* \*\*Documentation :\*\* Nous avons mis à jour et capitalisé sur le `Manuel Utilisateur` et la `Documentation de l'Architecture` pour y inclure les leçons apprises, les exemples et les sections de dépannage.

### 2. État Actuel du Projet

Le système a prouvé sa robustesse en intégrant un nouveau type de test complexe.

\* \*\*Tâches réalisées :\*\*

\* Développement, débogage et validation du module de logique `Logique\_MBTI.gs`.

\* Correction de la logique de retraitement multilingue au niveau des scripts "spécialistes".

\* Enrichissement et finalisation des documents de référence (`Manuel Utilisateur` et `Documentation de l'Architecture`).

\* \*\*Tâches en cours :\*\* Aucune. Le système est stable.

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

\* [cite\_start]Développer le script de logique pour le test des \*\*Ancres de Schein\*\* (`Logique\_Ancres.gs`). [cite: 35]

\* Préparer les données de questions en anglais pour le test "Ancres" (`Questions\_Ancres\_EN`).

\* Analyser et corriger le bug où les e-mails de type \*\*"simple" et "détaillé"\*\* sont pour l'instant identiques.

\* Implémenter la fonctionnalité de \*\*pièces jointes\*\* dans les e-mails.

\* [cite\_start]Aborder les \*\*nouvelles améliorations ergonomiques\*\* souhaitées. [cite: 36]

### 3. Contexte Historique du Projet

[cite\_start]Le projet sortait d'une phase de stabilisation majeure de son architecture, rendant le `[TEMPLATE] Kit de Traitement` autonome et entièrement internationalisé [cite: 17-22, 32]. La session d'aujourd'hui était le premier test grandeur nature de cette nouvelle architecture, consistant à y intégrer un nouveau module de logique (`MBTI`) de A à Z. Les bugs rencontrés, bien que complexes, ont validé la pertinence des choix d'architecture (découplage, ID invariants) car ils ont permis de circonscrire et de corriger les problèmes de manière ciblée.

### 4. Orientations Stratégiques

La session a validé et renforcé les orientations stratégiques du projet :

\* [cite\_start]\*\*Validation par le Test de Bout en Bout :\*\* Seuls des tests complets ont permis de révéler les failles de logique et les suppositions erronées sur le format des données. [cite: 50]

\* \*\*Importance de la Documentation d'Architecture :\*\* Le bug final a souligné l'importance de documenter non seulement le code, mais aussi le comportement des composants "externes" comme le `MOTEUR` (ex: le formatage `ID: Titre`).

\* [cite\_start]\*\*Découplage Logique/Données :\*\* Le principe des ID invariants a été la clé pour résoudre le bug de retraitement multilingue et s'est avéré être le pilier de la robustesse du système. [cite: 49]

### 5. Derniers Scripts et Ressources Clés

Le script le plus important de cette session est la version finale et fonctionnelle de `Logique\_MBTI.gs`.

#### Fichier `Logique\_MBTI.gs` (Version 3.0 - Définitive)

```javascript

// =================================================================================

// == FICHIER : Logique\_MBTI.gs (Le Spécialiste)

// == VERSION : 3.0 (DÉFINITIVE - Logique basée sur l'ID Invariant)

// == RÔLE : Calcule les résultats pour le test MBTI.

// =================================================================================

/\*\*

\* Fonction utilitaire pour charger et mapper les questions d'une langue donnée.

\* La clé du mappage est l'ID INVARIANT de la question.

\* @param {Object} bdd - L'objet Spreadsheet de la base de données.

\* @param {string} nomFeuille - Le nom de la feuille de questions à charger.

\* @returns {Object} Un mappage des questions par leur ID.

\*/

function mapQuestionsById(bdd, nomFeuille) {

const sheet = bdd.getSheetByName(nomFeuille);

if (!sheet) {

throw new Error(`Feuille de questions '${nomFeuille}' introuvable.`);

}

const data = sheet.getDataRange().getValues();

const headers = data.shift();

const idCol = headers.indexOf('ID');

const typeCol = headers.indexOf('TypeQuestion');

const optionsCol = headers.indexOf('Options');

const logiqueCol = headers.indexOf('Logique');

const mapById = {};

data.forEach(row => {

const qId = row[idCol];

if (qId) {

mapById[qId] = {

id: qId,

type: row[typeCol],

options: row[optionsCol].split(';').map(item => item.trim()),

logique: row[logiqueCol].split(';').map(item => item.trim())

};

}

});

return mapById;

}

function calculerResultatMBTI(reponses, langue, config) {

Logger.log(`Début du calcul pour le test MBTI en langue CIBLE : ${langue}.`);

const scores = { 'E': 0, 'I': 0, 'S': 0, 'N': 0, 'T': 0, 'F': 0, 'J': 0, 'P': 0 };

const systemIds = getSystemIds();

const bdd = SpreadsheetApp.openById(systemIds.ID\_BDD);

const langueRepondantBrute = reponses['Langue / Language'][0];

const mapLangue = { 'Français': 'FR', 'English': 'EN', 'Español': 'ES', 'Deutsch': 'DE' };

const langueOrigine = mapLangue[langueRepondantBrute] || 'FR';

Logger.log(`Langue d'ORIGINE de la réponse détectée : ${langueOrigine}`);

const nomFeuilleCible = `Questions\_${config['Type\_Test']}\_${langue}`;

const questionsCibleMap = mapQuestionsById(bdd, nomFeuilleCible);

const nomFeuilleOrigine = `Questions\_${config['Type\_Test']}\_${langueOrigine}`;

const questionsOrigineMap = mapQuestionsById(bdd, nomFeuilleOrigine);

for (const questionTitre in reponses) {

const questionId = questionTitre.split(':')[0].trim();

const questionOrigineInfo = questionsOrigineMap[questionId];

if (!questionOrigineInfo) continue;

const questionCibleInfo = questionsCibleMap[questionId];

if (!questionCibleInfo) continue;

switch (questionOrigineInfo.type) {

case 'CHOIX\_BINAIRE':

const reponseUtilisateur = reponses[questionTitre][0];

const reponseIndex = questionOrigineInfo.options.indexOf(reponseUtilisateur);

if (reponseIndex !== -1) {

const codeLogique = questionCibleInfo.logique[reponseIndex];

if (scores.hasOwnProperty(codeLogique)) {

scores[codeLogique]++;

}

}

break;

default:

break;

}

}

Logger.log("Scores MBTI calculés : " + JSON.stringify(scores));

const profilFinalCode =

(scores['E'] >= scores['I'] ? 'E' : 'I') +

(scores['S'] >= scores['N'] ? 'S' : 'N') +

(scores['T'] >= scores['F'] ? 'T' : 'F') +

(scores['J'] >= scores['P'] ? 'J' : 'P');

const nomFeuilleProfils = `Profils\_MBTI\_${langue}`;

const profilsSheet = bdd.getSheetByName(nomFeuilleProfils);

if (!profilsSheet) {

throw new Error(`L'onglet de profils '${nomFeuilleProfils}' est introuvable.`);

}

const profilsData = profilsSheet.getDataRange().getValues();

const pHeaders = profilsData.shift();

const pCodeCol = pHeaders.indexOf('Type MBTI');

const pTitreCol = pHeaders.indexOf('Titre');

const pDescCol = pHeaders.indexOf('Description');

const profilInfoRow = profilsData.find(row => row[pCodeCol] === profilFinalCode);

const resultats = {

profilFinal: profilFinalCode,

titreProfil: profilInfoRow ? profilInfoRow[pTitreCol] : "Profil Inconnu",

descriptionProfil: profilInfoRow ? profilInfoRow[pDescCol] : "Description non disponible.",

scoresData: scores

};

return resultats;

}

**6. Actions Recommandées / Prochaines Étapes**

* **Actions Immédiates :**
  + Valider que ce document de synthèse est complet et correct.
* **Questions pour la Prochaine Session :**
  + Par quelle tâche prioritaire souhaitons-nous commencer ?
    1. Le développement de Logique\_Ancres.gs ?
    2. L'analyse du bug des e-mails "simples/détaillés" ?
    3. La discussion sur l'implémentation des pièces jointes ?