Teo BASOL
BTS SIO 2

Automatisation des Relances et Suivi Qualité via Apps Script

I. Contexte

Ce script Google Apps Script a pour objectif d'automatiser le processus de relance concernant les Tableaux de Bords Projets. L'automatisation vise à améliorer le suivi de l'avancement des projets en s'assurant que les informations clés sont renseignées, ce qui peut également contribuer à l'évaluation de critères liés à l’intéressement annuel.

II. Prérequis

- Environnement Google Workspace (Google Sheets, Apps Script).
- Accès au fichier Google Sheets contenant les données de suivi des projets.
- Autorisations nécessaires pour que le script puisse lire la feuille de calcul et envoyer des emails via MailApp ou GmailApp.

III. Démarche et Structure de la Mise en Place

A. Récupération et Formatage des Données

```
function getSheetNames() {
  var spreadsheet = SpreadsheetApp.getActiveSpreadsheet();
  var sheets = spreadsheet.getSheets();
  sheets.map(function(sheet) {
    feuille.push(sheet.getName());
  });
}

getSheetNames();

function getLine(sheetName, array){
  var spreadsheet = SpreadsheetApp.getActiveSpreadsheet();
  var sheet = spreadsheet.getSheetByName(sheetName);
  var dataRange = sheet.getDataRange();
  var data = dataRange.getValues();
  data.map(function(row) {
    array.push(row);
  });
}

getLine(feuille[4], suiviAffaire)


function formatedArray (array){
  const headers = array[0]; // Première ligne
  const rows = array.slice(1); // Lignes restantes

  const formatedArray = rows.map(row => {
    return row.reduce((acc, value, index) => {
        acc[headers[index]] = value;
        return acc;
    }, {});
  });
  return formatedArray;
}
```

- getSheetNames() : Récupère les noms de toutes les feuilles du tableur actif.
- getLine(sheetName, array) : Lit toutes les données d'une feuille spécifiée et les stocke dans un tableau.
- formatedArray(array) : Transforme le tableau de données brutes en un tableau d'objets, où chaque objet représente une ligne et utilise les en-têtes de colonnes comme clés. Ceci facilite l'accès aux données par nom de colonne.

B. Filtrage des Affaires Non Remplies

- nonRempli(array, arrayNR) : Parcourt les affaires formatées et identifie celles qui nécessitent une relance. Le filtrage se base sur des conditions spécifiques liées aux colonnes "Code état démarrage", "Statut", "Tableau de bord", et "Phase en cours". Les affaires filtrées sont ajoutées à un tableau séparé (affaireNR)

C. Envoi des Emails de Relance

```
function envoieMail (){
  const codeToDetails = {};

  // Remplir le dictionnaire avec les détails
  for (const item of affaireNR) {
    if (item.Code && item["Mail groupe projet"] && item.Nom) {
      codeToDetails[item['Code']] = {
      code: item['Code'],
      projet: item['Nom'],
      origine: item['Origine'],
      codeEtat: item['Code état démarrage'],
      email: item["Mail groupe projet"],
      url: item['Tableau de bord']
    };
    } else {
      const mailAddress = "";
      const code = item['Code'];
      const name = item['Nom'];
      const mailGroup = item["Mail groupe projet"];
      const sujet = `Elements manquants pour les projets ${cod
      let corps = ""
      if (code !== 988){
        corps = `Bonjour,\n\nIl semblerait qu'il n'y ait pas d
        bord projet de ${code} - ${name}.\n\nPouvez-vous le re
        d'avance. \nCordialement.`;
        // console.log(mailAddress, sujet, corps)
        MailApp.sendEmail(mailAddress, sujet, corps);
      }
    }
  }
```

- Crée un dictionnaire (codeToDetails) pour stocker les informations pertinentes des affaires non remplies (code, nom, origine, email groupe projet, URL TBP).
- Vérifie si les informations essentielles sont présentes pour chaque affaire. Si l'email manque, un email est envoyé à des adresses administratives prédéfinies pour signaler le manque d'information.

```
// Parcourez le tableau des codes à relancer
for (const code of codeNR) {

  const details = codeToDetails[code];
  if (details) {
    if (details.origine === "France" || details.origine === ""){
      const mailAddress = details.email;
      const projet = details.projet;
      const url = details.url;
      const sujet = `Votre tableau ${code} - ${projet} ne semble pas rempli`;

      let corps =

      // Envoi de l'email avec le contenu HTML
      GmailApp.sendEmail(mailAddress, sujet, "", {
        htmlBody: corps
      });
      // console.log(mailAddress, sujet, corps)
    }
  }
}
```

- Boucle sur les codes des affaires non remplies.
- Pour chaque affaire ayant des détails valides et une origine "France" ou vide, un email de relance formaté en HTML est envoyé à l'adresse "Mail groupe projet".

D. Traitement de l’Intéressement

(script non indiqué ici, car il contient des données sensibles)

- Crée un dictionnaire (recapAffaire) contenant des informations plus détaillées sur toutes les affaires lues initialement.
- Filtre les affaires pour ne garder que celles pertinentes pour le calcul de l'intéressement, en excluant certains statuts. Les affaires retenues sont stockées dans arrayInteressement.
- Calcule deux taux de remplissage (txcode, txrevue) pour chaque affaire en convertissant l'état de remplissage de différents champs en valeurs numériques (0 ou 1) et en appliquant des moyennes. Le calcul de txrevue dépend de la phase du projet.
- Calcule un taux global (total) basé sur les deux taux précédents.
- Boucle sur les affaires retenues pour l'intéressement. Si le taux total est inférieur à un seuil (90%) et que l'affaire est "en cours", un email est envoyé au "Mail groupe projet" pour notifier du faible taux de remplissage et encourager la mise à jour du TBP.

IV. Remarques Techniques

- Gestion des Données Sensibles : Le script manipule des données de projets et des adresses email. Il est important que l'accès au script et à la feuille de calcul sous-jacente soit restreint aux personnes autorisées. Les adresses email administratives pour les notifications sont codées en dur dans le script.
- Dépendance à la Structure de la Feuille : Le script dépend fortement des noms exacts des colonnes et des valeurs spécifiques utilisées dans la feuille Google Sheets. Toute modification de la structure de la feuille nécessitera une mise à jour du script.
- Logique de Filtrage : La complexité des conditions dans les fonctions nonRempli et traitementInteressement (notamment les filtres sur les statuts, phases, origines) nécessite une bonne compréhension des processus métier pour être maintenue ou adaptée.

V. Évolutions Possibles

- Externaliser les configurations (seuils, adresses email admin, textes des emails, critères de filtrage) dans une feuille de configuration dédiée pour faciliter les ajustements sans modifier le code.
- Implémenter un système de logging plus détaillé pour suivre les exécutions, les affaires traitées et les erreurs éventuelles.

VI. Bilan

Ce script Google Apps Script fournit une solution d'automatisation efficace pour gérer les relances des tableaux de bord projets et effectuer un suivi qualité lié à l’intéressement. En lisant et traitant les données d'une feuille Google Sheets, il identifie les projets nécessitant une action, notifie les équipes concernées et alerte l'administration en cas de données manquantes. La gestion des accès et des données sensibles est un point clé.