Bugfix Guide

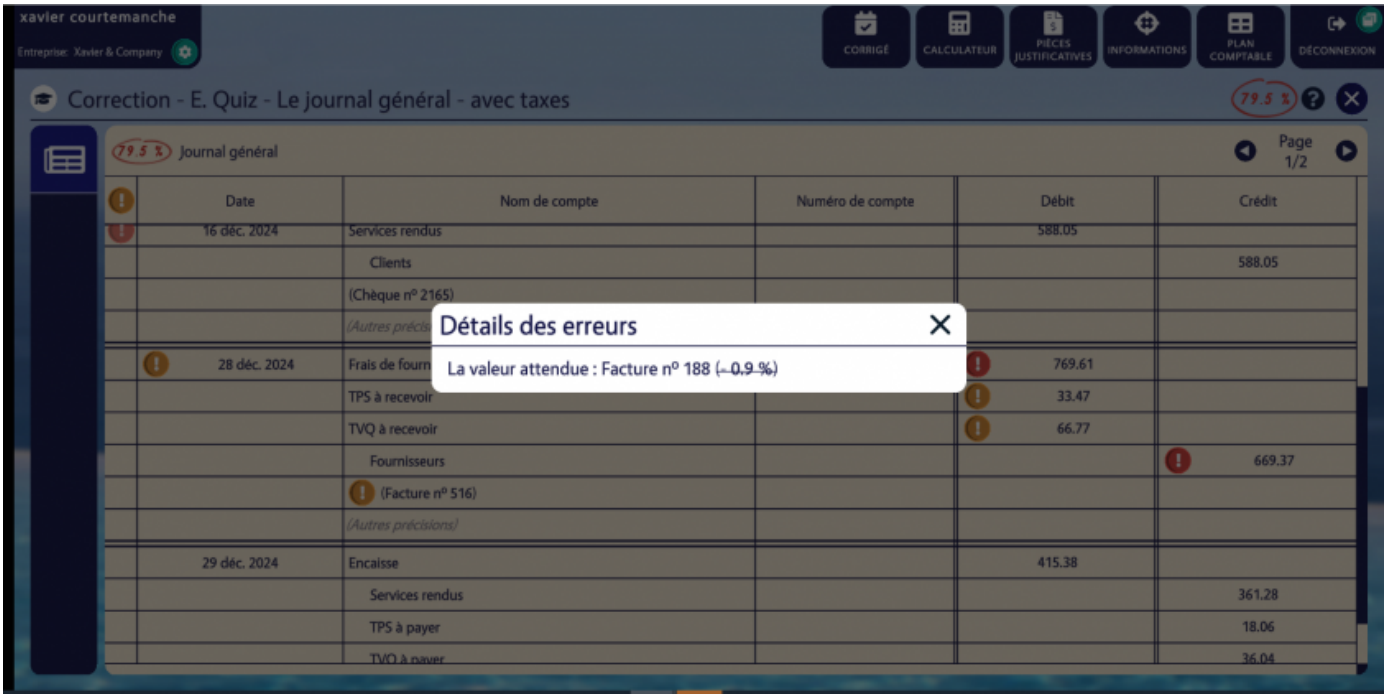
- Xavier courtemanche - bug de currection

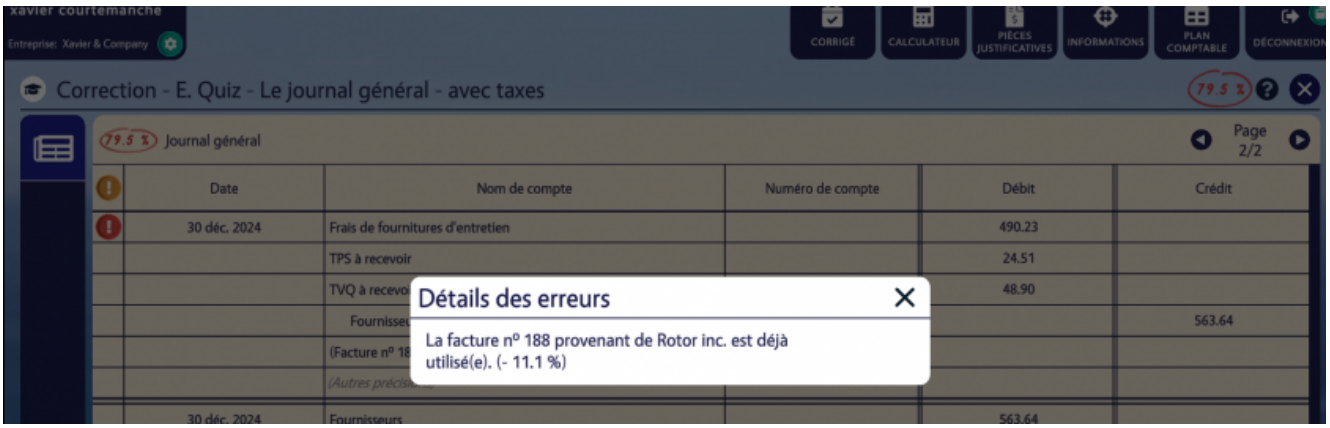
Xavier courtemanche - bug de currection

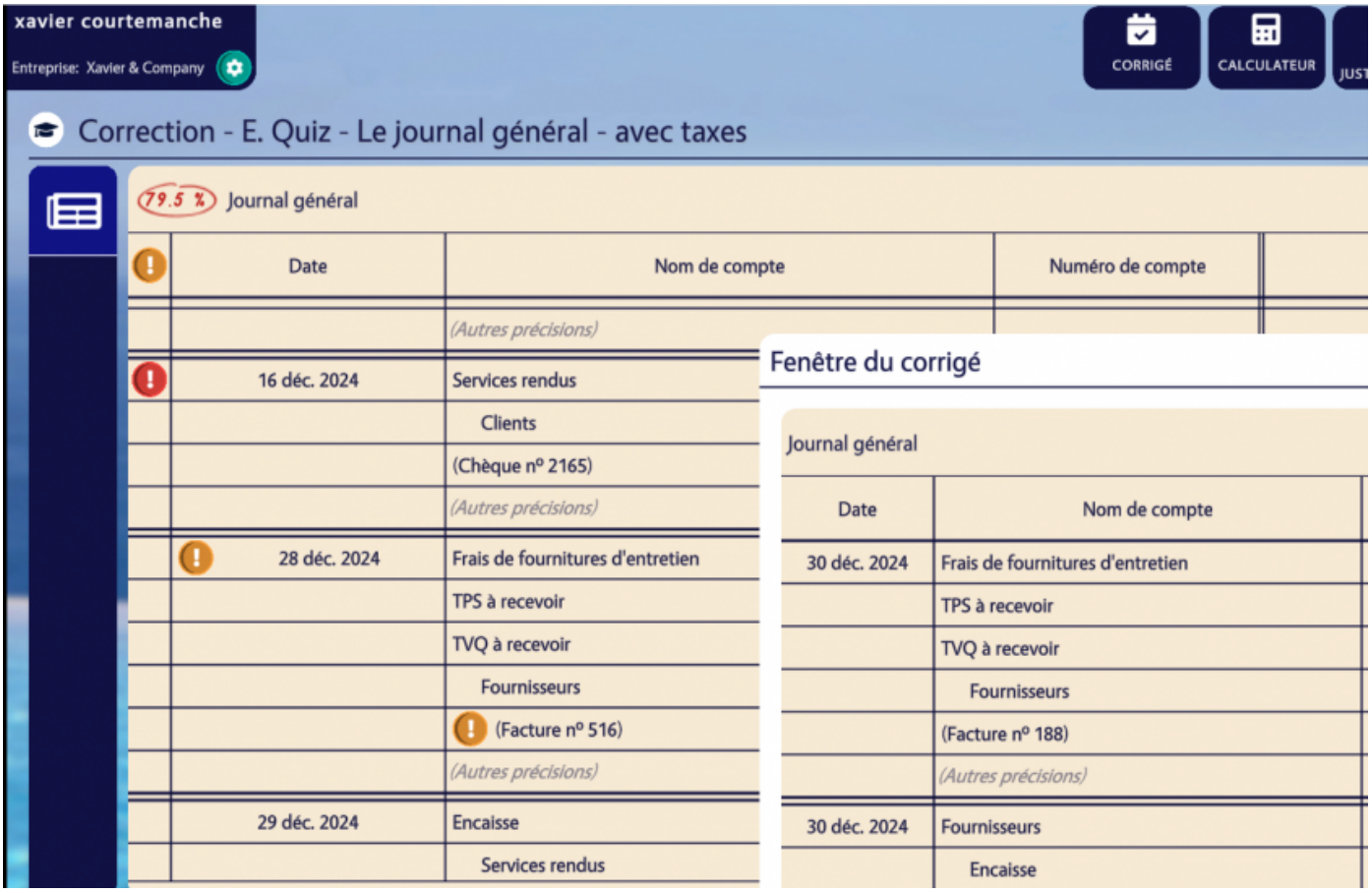
Journal Général

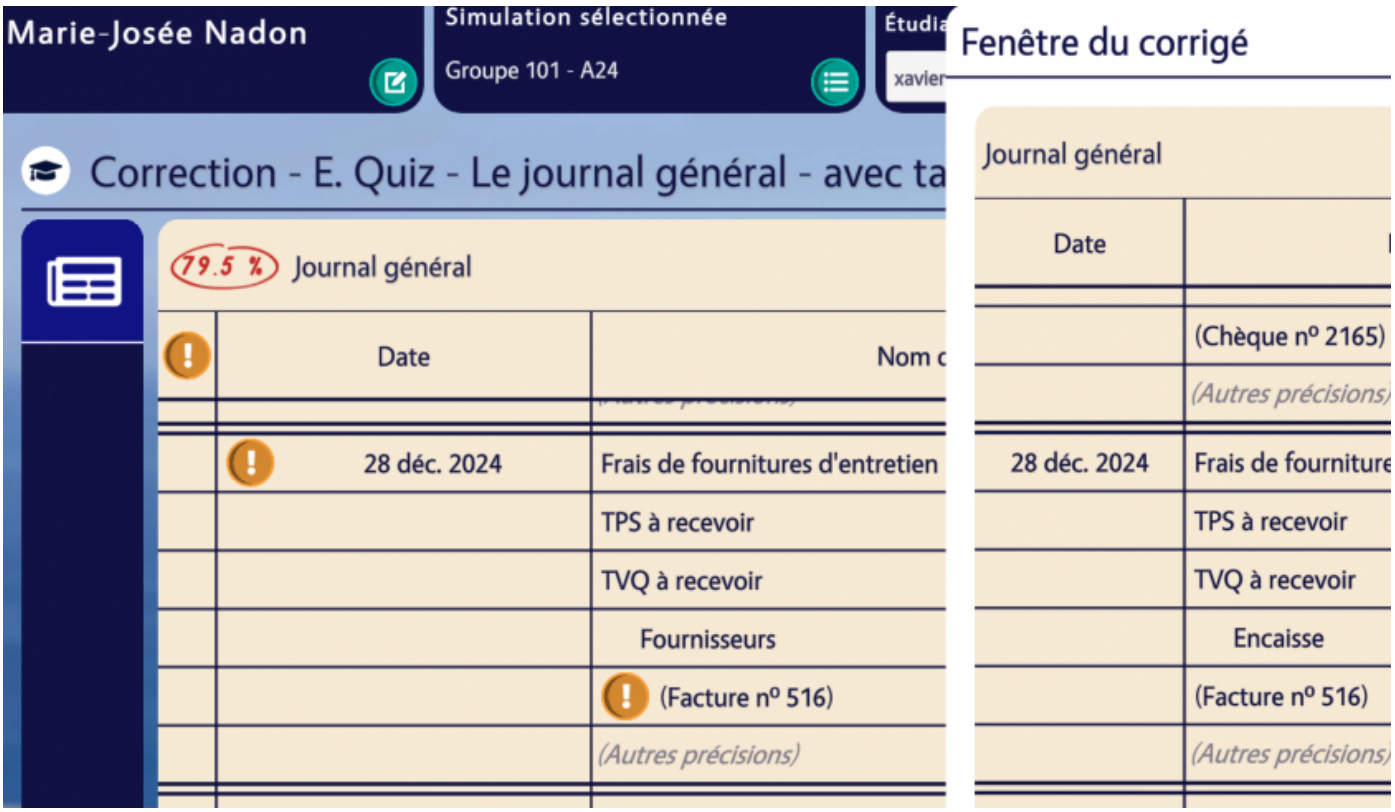
Description du problème: les identifiants des factures sont données en odres d'apparitions des écritures dans le J.G. Le système detecte que la "bonne" écriture est la facture n. 188 puisque les comptes sont un match parfait avec ce que l'étudiant à écrit. Ce problème origine de l'erreur de l'étudiant (n'a pas écris le bon compte pour une écriture) et du fait que nous ne faisont pas multiples tentatives de trouvé la bonne facture associé à l'écriture.

Solution: Faire multiple tentatives d'associations.

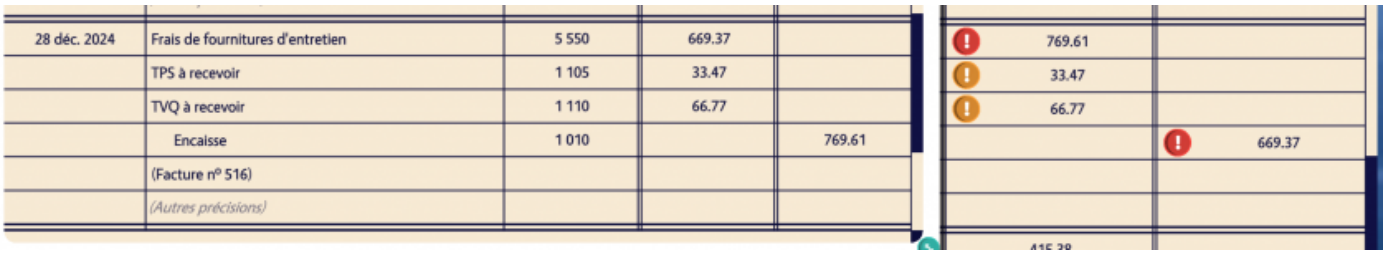

xavier courtemanche
Entreprise: Xavier & Company
CORRIGÉ
CALCULATEUR
PIÈCES JUSTIFICATIVES
INFORMATIONS
PLAN COMPTABLE
DÉCONNEXION
Correction - E. Quiz - Le journal général - avec taxes
79.5 %
Journal général
Page 1/2
Détails des erreurs
La valeur attendue : Facture nº 188 (-0.9 %)
Date
Nom de compte
Numéro de compte
Débit
Crédit
16 déc. 2024
Services rendus
588.05
Clients
588.05
(Chèque nº 2165)
(Autres précisions)
28 déc. 2024
Frais de fourn
769.61
TPS à recevoir
33.47
TVQ à recevoir
66.77
Fournisseurs
669.37
(Facture nº 516)
(Autres précisions)
29 déc. 2024
Encaisse
415.38
Services rendus
361.28
TPS à payer
18.06
TVQ à payer
36.04


xavier courtemanche
Entreprise: Xavier & Company
CORRIGÉ
CALCULATEUR
PIÈCES JUSTIFICATIVES
INFORMATIONS
PLAN COMPTABLE
DÉCONNEXION
Correction - E. Quiz - Le journal général - avec taxes
79.5 %
Journal général
Page 2/2
Détails des erreurs
La facture nº 188 provenant de Rotor inc. est déjà utilisé(e). (- 11.1 %)
Date
Nom de compte
Numéro de compte
Débit
Crédit
30 déc. 2024
Frais de fournitures d'entretien
490.23
TPS à recevoir
24.51
TVQ à recevo
48.90
Fournisseu
563.64
(Facture nº 18
(Autres précisions)
30 déc. 2024
Fournisseurs
563.64


xavier courtemanche
Entreprise: Xavier & Company
CORRIGÉ
CALCULATEUR
Correction - E. Quiz - Le journal général - avec taxes
79.5 % Journal général
Date
Nom de compte
Numéro de compte
(Autres précisions)
16 déc. 2024
Services rendus
Clients
(Chèque nº 2165)
(Autres précisions)
28 déc. 2024
Frais de fournitures d'entretien
TPS à recevoir
TVQ à recevoir
Fournisseurs
(Facture nº 516)
(Autres précisions)
29 déc. 2024
Encaisse
Services rendus
Fenêtre du corrigé
Journal général
Date
Nom de compte
30 déc. 2024
Frais de fournitures d'entretien
TPS à recevoir
TVQ à recevoir
Fournisseurs
(Facture nº 188)
(Autres précisions)
30 déc. 2024
Fournisseurs
Encaisse


Marie-Josée Nadon
Simulation sélectionnée
Groupe 101 - A24
Correction - E. Quiz - Le journal général - avec ta
79.5 % Journal général
Date
28 déc. 2024
Frais de fournitures d'entretien
TPS à recevoir
TVQ à recevoir
Fournisseurs
(Facture nº 516)
(Autres précisions)
Fenêtre du corrigé
Journal général
Date
(Chèque nº 2165)
(Autres précisions)
28 déc. 2024
Frais de fourniture
TPS à recevoir
TVQ à recevoir
Encaisse
(Facture nº 516)
(Autres précisions)

Les montants ne sont pas bon aussi, ce qui peut être une autre cause du problème.

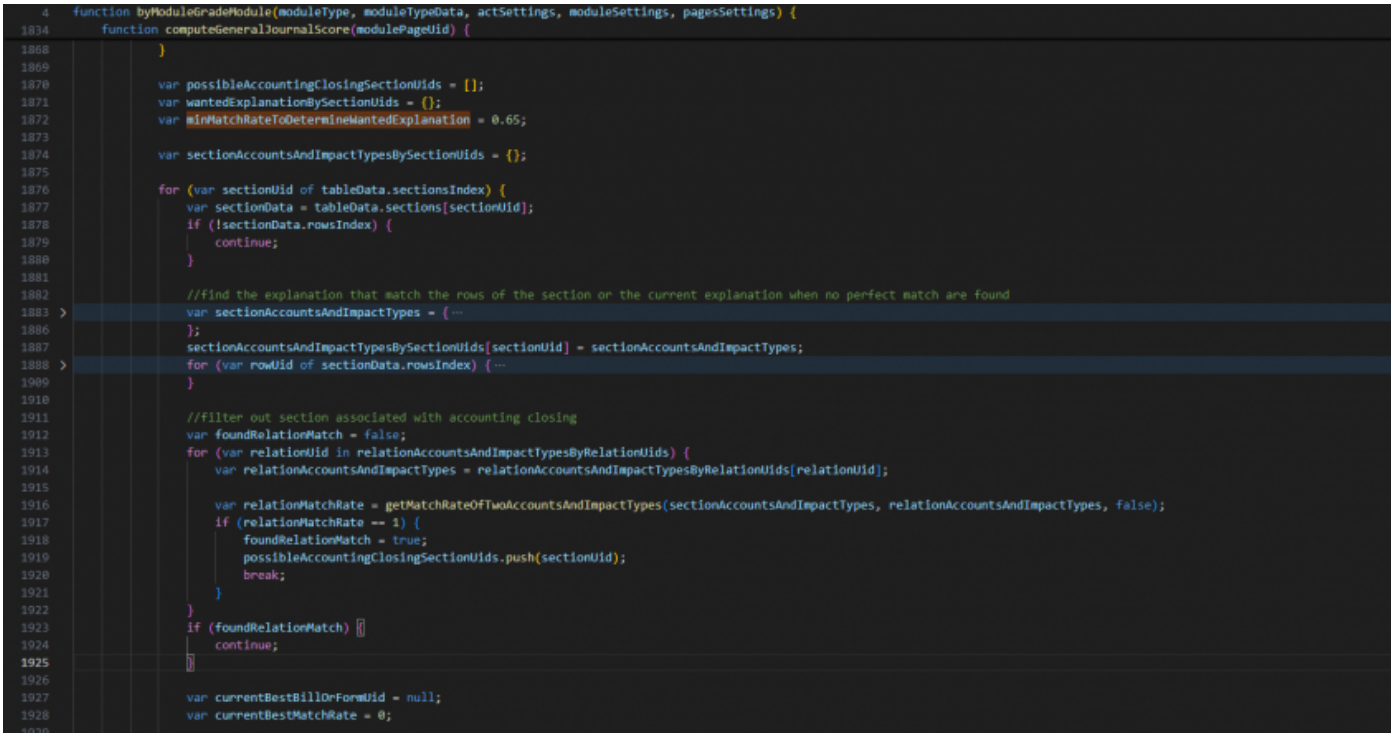


28 déc. 2024	Frais de fournitures d'entretien	5 550	669.37	
	TPS à recevoir	1 105	33.47	
	TVQ à recevoir	1 110	66.77	
	Encaisse	1 010		769.61
	(Facture nº 516)			
	(Autres précisions)			

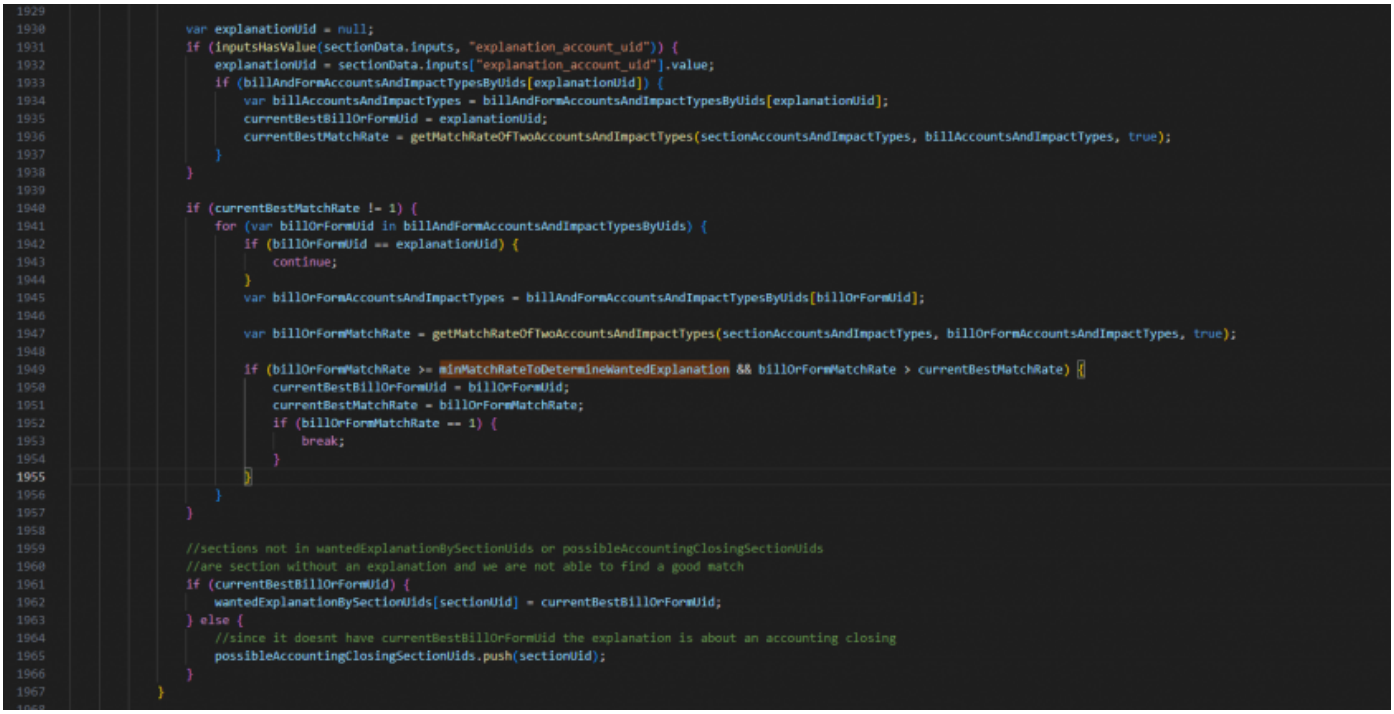
769.61		
33.47		
66.77		
	669.37	

Description detaillé de la solution proposé

L'algo actuel se fait en une seul passe pour l'association des écritures avec factures (si la facture ne match pas parfaitement la facture indiqué par l'étudiant, prend la prochaine meilleur qui à un taux de match supérieur à 65%). Ce qui peut causé une double erreur puisqu'une écriture peut "voler" une bonne facture à une autre écriture



```
function byModuleGradeModule(moduleType, moduleTypeData, actSettings, moduleSettings, pagesSettings) {
    function computeGeneralJournalScore(modulePageUid) {
        }

        var possibleAccountingClosingSectionUids = [];
        var wantedExplanationBySectionUids = {};
        var minMatchRateToDetermineWantedExplanation = 0.65;

        var sectionAccountsAndImpactTypesBySectionUids = {};

        for (var sectionUid of tableData.sectionsIndex) {
            var sectionData = tableData.sections[sectionUid];
            if (!sectionData.rowsIndex) {
                continue;
            }

            //find the explanation that match the rows of the section or the current explanation when no perfect match are found
            var sectionAccountsAndImpactTypes = {…
            };
            sectionAccountsAndImpactTypesBySectionUids[sectionUid] = sectionAccountsAndImpactTypes;
            for (var rowUid of sectionData.rowsIndex) {…
            }

            //filter out section associated with accounting closing
            var foundRelationMatch = false;
            for (var relationUid in relationAccountsAndImpactTypesByRelationUids) {
                var relationAccountsAndImpactTypes = relationAccountsAndImpactTypesByRelationUids[relationUid];

                var relationMatchRate = getMatchRateOfTwoAccountsAndImpactTypes(sectionAccountsAndImpactTypes, relationAccountsAndImpactTypes, false);
                if (relationMatchRate == 1) {
                    foundRelationMatch = true;
                    possibleAccountingClosingSectionUids.push(sectionUid);
                    break;
                }
            }
            if (foundRelationMatch) {
                continue;
            }

            var currentBestBillOrFormUid = null;
            var currentBestMatchRate = 0;
```



```
        var explanationUid = null;
        if (inputsHasValue(sectionData.inputs, "explanation_account_uid")) {
            explanationUid = sectionData.inputs["explanation_account_uid"].value;
            if (billAndFormAccountsAndImpactTypesByUids[explanationUid]) {
                var billAccountsAndImpactTypes = billAndFormAccountsAndImpactTypesByUids[explanationUid];
                currentBestBillOrFormUid = explanationUid;
                currentBestMatchRate = getMatchRateOfTwoAccountsAndImpactTypes(sectionAccountsAndImpactTypes, billAccountsAndImpactTypes, true);
            }
        }

        if (currentBestMatchRate != 1) {
            for (var billOrFormUid in billAndFormAccountsAndImpactTypesByUids) {
                if (billOrFormUid == explanationUid) {
                    continue;
                }
                var billOrFormAccountsAndImpactTypes = billAndFormAccountsAndImpactTypesByUids[billOrFormUid];

                var billOrFormMatchRate = getMatchRateOfTwoAccountsAndImpactTypes(sectionAccountsAndImpactTypes, billOrFormAccountsAndImpactTypes, true);

                if (billOrFormMatchRate >= minMatchRateToDetermineWantedExplanation && billOrFormMatchRate > currentBestMatchRate) {
                    currentBestBillOrFormUid = billOrFormUid;
                    currentBestMatchRate = billOrFormMatchRate;
                    if (billOrFormMatchRate == 1) {
                        break;
                    }
                }
            }
        }

        //sections not in wantedExplanationBySectionUids or possibleAccountingClosingSectionUids
        //are section without an explanation and we are not able to find a good match
        if (currentBestBillOrFormUid) {
            wantedExplanationBySectionUids[sectionUid] = currentBestBillOrFormUid;
        } else {
            //since it doesnt have currentBestBillOrFormUid the explanation is about an accounting closing
            possibleAccountingClosingSectionUids.push(sectionUid);
        }
    }
```

La solution que je propose afin de résoudre ce problème va comme suit (en pseudo code):

```
currentThreshold = 1
nextThreshold = 0
wantedExplanationBySectionUids = {}listOfSectionUids = copyOf(tableData.sectionsIndex)
billUidRemainings = copyOf(billUids)
tant que listOfSectionUids est pas vide
for sectionUid of listOfSectionUids    for billUid of billUidRemainings (ce ne sera pas
exactement représenté comme cela dans le code)threshold = getMatchRate(sectionUid,
billUid)
            if (threshold == currentThreshold)
wantedExplanationBySectionUids[sectionUid] = billUid
listOfSectionUids.remove(sectionUids)
billUidRemainings.remove(billUid)                break billUidRemaining
loop

            nextThreshold = max(threshold, nextThreshold)

currentThreshold = nextThreshold
```

Ce pseudo-algorithme devrait réduire les cas d'erreurs.

Afin d'implémenté ce pseudo code une bonne partie du code restera parreil.