

Grading, reassemble et calcul des scores

Trois étapes distinctes

La correction By Module n'est pas un seul calcul. Elle sépare : **grading brut**, **réassemblage/pondération**, puis **publication du résultat**.

```
flowchart LR
    DEC[Décisions étudiant] --> GRADE[byModuleGradeModule]
    AK[Answer Key] --> GRADE
    GRADE --> RAW[scores bruts + errors + expectedValues + ignoredPageUids]
    RAW --> RE[byModuleReassembleAnswer]
    RE --> FINAL[scores finaux + errorScores]
    FINAL --> GEN[GenerateTeamModuleResults]
```

byModule_utils_grade_module.js

`byModuleGradeModule()` initialise le contexte de grading et compare la donnée étudiante aux valeurs attendues. Le moteur route chaque `modulePageUid` vers son correcteur spécialisé et retourne :

- `scores` bruts;
- `errors` structurées;
- `expectedValues`;
- `ignoredPageUids`.

Si les formules sont actives, leurs valeurs peuvent être pré-calculées avant la comparaison.

byModule_utils_reassemble_answers.js

`byModuleReassembleAnswer()` transforme les scores bruts selon la configuration de `pages_settings`. Il charge notamment `inputs_ponderation`, `scores_ponderation`, `scores_details` et les colonnes ignorées.

Le moteur supporte des chaînes de pondération et des pénalités. Il recalcule aussi un arbre `errorScores` qui permet au frontend d'associer la perte de points aux cellules/rows concernées.

Pages ignorées

Une page dans `ignoredPageUids` reçoit un score `-1` au réassemblage. Cela distingue une page non évaluée d'une page évaluée à 0.

Arrondis

Le réassemblage contient une logique explicite de correction des erreurs d'arrondi autour de 0 afin que la somme des pénalités ne produise pas des scores légèrement négatifs ou des reliquats artificiels.

Configuration

Les pondérations sont data-driven. Modifier un score n'implique donc pas nécessairement de changer le JavaScript : commencer par vérifier `pages_settings.json` et ses `scores_details/scores_ponderation`.

Tests

Les tests de `grade_module_tests/` couvrent grading générique, balance sheet, income taxes et petty cash. Lorsqu'un changement d'answer key modifie une valeur attendue, exécuter également les tests de grading correspondants.

Index détaillé du moteur de grading

`byModule_utils_grade_module.js` est encore plus volumineux que l'Answer Key. Il combine routeurs de pages, matching structurel et scoring générique des inputs/options.

Orchestration

- `byModuleGradeModule()` — entrée publique;
- `gradeModule()` — orchestration;
- `initGradeModuleVariables()` — contexte de correction;
- `trimData()` — normalisation;
- `preComputeFormulaValues()` — calcul des formules avant comparaison;
- `computeModulePageScore()` — routeur de page.

Correcteurs spécialisés

Le moteur possède des fonctions distinctes pour purchases/sales/withdrawals/cashings/payroll/petty cash/general journal, taxes forms, subledgers, ledger, bank reconciliation, trial balance, statements et accounting journal. Lorsqu'une seule page échoue, commencer par son correcteur spécialisé avant les helpers génériques.

Matching structurel : pourquoi le grading n'est pas une simple égalité JSON

Le General Journal et plusieurs tableaux doivent accepter des rows dans un ordre ou une structure qui peut différer de l'Answer Key. Le moteur utilise notamment :

- `getMatchRateOfTwoAccountsAndImpactTypes()` ;
- `compareAccounts()` ;
- `getMatchRateForGeneralJournalSection()` ;
- `getTypeAndExplanationOfSection()` ;
- `billBelongsInGeneralJournal()` .

Avant de “simplifier” ce code en comparaison directe, vérifier les cas où l'étudiant peut produire une écriture équivalente mais ordonnée différemment.

Scoring générique des inputs

- `computeInputsGoodAnswers()` prépare les valeurs attendues;
- `inputHasGoodAnswer()` applique valeurs/tolérances;
- `computeRowInputsValueScore()`, section/table/page remontent le score;
- `computeInputsOptionsScore()` corrige les options de format/choix;
- `addError()`, `addExpectedValue()`, `addAnswerScore()` alimentent les sorties.

Ajouter le grading d'une nouvelle page

1. Déterminer si la page peut utiliser le grading générique table/inputs.
2. Sinon créer `compute<Page>Score()`.
3. Brancher la page dans `computeModulePageScore()`.
4. Réutiliser les helpers de score/tolérance/options.
5. Ajouter les `expectedValues` utiles à la correction visuelle.
6. Ajouter les erreurs avec un chemin suffisamment précis pour retrouver la cellule.
7. Vérifier les `ignored_grading_*` preferences.
8. Ajouter le score dans `pages_settings` / `scores_details` si nécessaire.
9. Tester une bonne réponse = 100 %, puis plusieurs mauvaises réponses indépendantes.

Account State côté grading

Le grading reconstruit lui aussi l'état des comptes. Certaines fonctions reçoivent `useUserRegularizations` : cela signifie que la page doit parfois être évaluée à partir des régularisations effectivement saisies par l'étudiant plutôt qu'à partir de la bonne réponse. Ce détail est critique pour les pages dépendantes.

Grading vs Reassemble : frontière à respecter

Problème	Modifier
Une réponse est reconnue bonne/mauvaise incorrectement	<code>grade_module</code> .
La bonne cellule perd le mauvais nombre de points	grading ou score detail selon le raw score.
Les raw scores sont bons mais le total final est faux	<code>reassemble_answers</code> / pondérations.
Une page ne devrait pas être évaluée	ignored pages/settings.

Test de régression minimal

Pour chaque bug de grading : ajouter un test qui reproduit exactement la structure étudiante fautive, vérifier le score attendu, les errors et les expectedValues. Ensuite lancer la suite complète du type de page pour éviter de casser les équivalences existantes.

Matching centralisé et ledger — septembre 2026

`matchUidsToAnswers()` centralise la priorité des identifiants stricts puis les passages fuzzy jusqu'au seuil inclusif de 0.5 pour les journaux auxiliaires, la petite caisse, les subledgers clients/fournisseurs et les écritures admissibles du ledger. Le score de matching sélectionne une réponse attendue; il ne remplace pas le scoring des cellules. Le General Journal conserve son algorithme distinct.

Voir [Fuzzy matching — logique de correspondance du grading Accounting](#) pour les poids, seuils, exemples, limites et le changement du ledger du 30 septembre.

Revision #4

Created 2026-09-18 19:16:31 UTC by Mathis

Updated 2026-09-30 14:36:29 UTC by Mathis