UQÀM | Département d'informatique
FACULTÉ DES SCIENCES
Université du Québec à Montréal

UNICAL - Assistant d'étude et de budget intelligent pour étudiants

RAPPORT DU SPRINT 2

Présenté à :

Gnagnely Serge Dogny

Dans le cadre du cours :

INM5151- Projet d'analyse et de modélisation:

Automne 2025 - Gr. 020

Travail réalisé par :

ÉQUIPE # 3, composée de :

KENFACK NANDJOU, Steve Marvyn

KUBARIEVA, Oryna

MUNYANEZA, Ursule Joana

SOKOUDJOU SOKAMTÉ Romuald

THIAM, Demba Aziz

Dimanche 23 novembre 2025

GABARIT DU RAPPORT DU SPRINT 2

TABLE DES MATIÈRES

TABLES DES MATIÈRES

1. INTRODUCTION

Ce document présente une vue d'ensemble du projet UNICAL. Il débute par une introduction du système, accompagnée d'une description générale qui expose le public cible, les contraintes ainsi que le contexte d'utilisation de l'application.
Par la suite, le document aborde la présentation du Sprint 1, qui avait été préalablement planifié avant le développement. Cette section inclut la comparaison entre les estimations planifiées et les résultats réels, le contenu développé, ainsi que les fonctionnalités réalisées, tout en mettant en évidence les contraintes et les difficultés rencontrées par l'équipe durant le processus de développement.
Les cas d'utilisation sont ensuite présentés afin d'illustrer le fonctionnement du système dans sa version actuelle et de montrer concrètement comment les différentes fonctionnalités peuvent être utilisées par les utilisateurs.
Enfin, le document se conclut par une brève présentation du prochain sprint, décrivant les estimations prévues, les fonctionnalités qui seront ajoutées ou améliorées, ainsi que les objectifs à atteindre lors de la prochaine phase de développement.

1.1 Objectifs

L'objectif de ce document est de présenter l'ensemble des fonctionnalités préétablies qui ont été réalisées au cours du développement du Sprint 1, tout en décrivant les démarches entreprises ainsi que les difficultés rencontrées par chaque membre de l'équipe. De plus, il offre un premier aperçu du logiciel à travers son interface utilisateur, ses principales pages, et le fonctionnement général des modules déjà implémentés.

1.2 Vue d'ensemble du produit

Description générale

UNICAL est une application conçue pour centraliser la gestion du temps et du budget des étudiants.
Elle regroupe dans un même espace :

- Le calendrier académique (cours, travaux, examens),
- Les créneaux d'étude personnalisables (minuteur de 25, 45, 60 ou 90 minutes, avec pause et reprise),
- Et un outil de gestion budgétaire simplifié (suivi des revenus et dépenses, alertes en cas de dépassement).
- L'application intègre une intelligence artificielle (IA) qui :
- Propose des priorités de base (version initiale),
- Effectue une replanification automatique en cas de conflit dans l'emploi du temps,
- Fournit des repères budgétaires simples pour aider à mieux gérer les finances étudiantes.

UNICAL offre également des vues synthétiques (par semaine, « prévu vs réalisé », barres de progression) ainsi que des notifications cohérentes pour les échéances académiques et les rappels budgétaires.

Limites et bordures

Le système :

- Ne remplace pas les plateformes institutionnelles officielles (ex. Moodle, Omnivox, Teams, etc.),
- Ne gère pas les communications entre étudiants ou enseignants,
- Ne vise pas la comptabilité avancée ni la gestion de dettes complexes.

Il agit comme un complément intelligent et pratique à la planification académique et financière.

Contexte et objectifs

UNICAL s'inscrit dans un contexte où de nombreux étudiants doivent jongler entre études, projets et emploi, tout en respectant un budget limité.

Les objectifs principaux du projet sont :

- d'optimiser la gestion du temps et du budget au quotidien,
- de réduire la charge mentale liée à la planification,
- et de favoriser l'autonomie et la productivité des étudiants.
- Les bénéfices attendus incluent :
- une meilleure organisation académique,
- une vision claire du progrès personnel,
- et une meilleure maîtrise des finances étudiantes.

Les retombées du projet visent à améliorer la réussite académique et le bien-être global des étudiants.

1.3 Définitions, acronymes et abréviations

Non applicable

1.4 Documents de références

Voici la liste des applications qui ont été utilisé comme la source de l'inspiration pour notre système :

- Pomodoro Tracker

  Pomodoro Tracker. Pomodoro Tracker (Version web) [Application web de gestion du temps]. Pomodoro-tracker.com. https://pomodoro-tracker.com/

- YEolPumTa (YPT)

  YeolPumTa. YeolPumTa (Version mobile) [Application de productivité et d'étude]. YeolPumTa. (Disponible sur Google Play et App Store)

- Todoist

  Doist. Todoist (Version web et mobile) [Application de gestion de tâches]. Doist Ltd. https://www.todoist.com/

- Notion

  Notion Labs Inc. Notion (Version web et mobile) [Application de prise de notes et de gestion de projets]. Notion Labs Inc. https://www.notion.com/

- Google Tasks

  Google LLC. Google Tasks (Version web) [Application de gestion de tâches]. Google Workspace. https://workspace.google.com/products/tasks/

- Trello

  Atlassian. Trello (Version web et mobile) [Application de gestion de projets et d'organisation visuelle]. Atlassian. https://trello.com/

- Clockify

  COING Inc. Clockify (Version web et mobile) [Application de suivi du temps et de productivité]. COING Inc. https://clockify.me/

- Microsoft To Do

  Microsoft Corporation. Microsoft To Do (Version web et mobile) [Application de gestion de tâches et rappels]. Microsoft 365. https://to-do.microsoft.com/

- Open source icons. https://ionic.io/ionicons
- CSS for buttons. https://getcssscan.com/css-buttons-examples

2. DESCRIPTION GÉNÉRALE DU LOGICIEL

2.1 Vue d'ensemble des fonctions du produit

Le système UNICAL est une application web et mobile destinée aux étudiant·e·s souhaitant planifier efficacement leur temps d'étude, suivre leur progression et gérer leur budget personnel.
Les principales fonctionnalités sont regroupées selon les modules suivants :

1. Calcul de priorité d'événement

Description : Une fonctionnalité de calcul de priorité basique a été développée et intégrée au système. Elle repose sur une formule pondérée qui prend en compte plusieurs paramètres clés liés à l'événement, tels que l'urgence, l'impact, la probabilité, l'ancienneté ou encore la criticité métier.

Les étapes réalisées sont les suivantes :

- Identification et définition des paramètres nécessaires ainsi que de leurs coefficients de pondération.
- Conception de la formule de calcul de l'indice de priorité
- Implémentation du calcul en code
- Intégration complète dans le système existant
- Validation fonctionnelle et tests unitaires/end-to-end pour garantir la fiabilité des résultats.
- Ajout d'une visualisation claire de l'indice de priorité dans l'interface utilisateur

2. Barre de progression dynamique

Description : Une barre de progression visuelle et précise a été ajoutée pour refléter en temps réel l'état d'avancement des événements ou dossiers.

Les travaux réalisés sont les suivants :

- Identification des éléments mesurant l'avancement
- Développement du calcul du pourcentage d'avancement
- Conception et styling de la barre de progression
- Intégration dynamique dans l'interface
- Tests et validation approfondis : exactitude du pourcentage dans tous les cas et rendu visuel cohérent sur tous les états, toutes tailles d'écran et thèmes

3. Vue "Plan d'étude" – Vue quotidienne minimaliste

Fonctionnalités implémentées :

- Affichage quotidien automatique des blocs d'étude et tâches prévues pour la journée en cours
- Design minimaliste et épuré : liste verticale avec heure, matière, intitulé de la tâche/bloc et durée.
- Filtre en temps réel en haut de page : Tous / À faire / Terminés (avec compteur de tâches restantes).
- États visuels différenciés (tâche terminée barrée et grisée, tâche en cours/en retard surlignée).

4. Fonctionnalité Chronomètre Pomodoro / Temps de bloc (v1)

Travaux réalisés :

- Allocation automatique du temps défini pour chaque bloc (ex. 25 min, 50 min ou durée personnalisée).
- Lancement du chronomètre au clic sur « Démarrer le bloc » avec compte à rebours précis (mm:ss).
- Boutons Pause / Reprise fluides
- Arrêt automatique et marquage « Terminé » dès que le temps alloué est écoulé
- Gestion de fin de session : à l'expiration du bloc, une notification modale apparaît avec :
- Message de félicitations et résumé du bloc accompli
- Options : « Marquer comme terminé », « Prolonger de 5 / 10 / 15 min », ou « Passer au bloc suivant »
- Possibilité de désactiver les notifications sonores.

5. Vue quotidienne avec bouton « Commencer »

Travaux réalisés :

- Récupération et filtrage des tâches du jour : interrogation de Supabase (table tasks) ou fallback localStorage, filtrage précis sur les tâches dont la date prévue correspond à la journée actuelle (timezone-safe).
- Création de l'interface quotidienne : liste verticale épurée affichant pour chaque tâche :
- Catégorie / Matière (pastille colorée)
- Durée estimée
- Statut visuel clair (« À faire », « En cours », « Terminé ») avec code couleur et icône.
- Ajout du bouton « Commencer » :
- Visible uniquement sur les tâches « À faire »
- Au clic → mise à jour instantanée du statut en « En cours » dans Supabase
- Retour visuel immédiat : bouton devient « En cours… », ligne surlignée, icône de chronomètre activé
- Une seule tâche « En cours » autorisée à la fois (les autres boutons « Commencer » se grisent si une tâche est déjà en cours).

6. Module Gestion des revenus et dépenses (v1)

Fonctionnalités implémentées :

- Ajout de revenus : formulaire rapide pour saisir source (bourse, job, aide familiale, etc.), montant, date et récurrence éventuelle.
- Modification & suppression des revenus : édition ou suppression en un clic depuis la liste (avec confirmation).
- Ajout de dépenses liées aux études : catégories pré-remplies (frais de scolarité, trimestre, livres, transport, logement étudiant, matériel informatique, etc.) + possibilité d'ajouter des catégories personnalisées.
- Modification & suppression des dépenses : édition ou suppression sécurisée de chaque entrée.
- Vue tableau récapitulatif clair :
- Liste chronologique ou par catégorie
- Total revenus / total dépenses / solde actuel
- Filtre par mois ou par trimestre scolaire
- Indicateur visuel (vert/rouge) quand le solde devient négatif
- Stockage sécurisé dans Supabase (table finances) avec synchronisation offline.

7. Système d'alertes budgétaires intelligentes

Fonctionnalités implémentées :

- Paramétrage des limites: seuils personnalisables par mois, trimestre ou année scolaire (
- Surveillance en temps réel du total des dépenses et calcul automatique du solde restant.
- Détection immédiate des seuils critiques :

    - ≥ 90 % du budget → alerte orange
    - ≥ 100 % → alerte rouge (dépassement)
- Affichage prioritaire et non intrusif :
- Barre fixe en haut de l'écran Finances (persiste jusqu'à action de l'utilisateur)
- Boutons d'action rapides : « Voir le détail », « Ajuster le budget » ou « Masquer 48 h »
- Notifications complémentaires : push browser, notification mobile et (optionnel) rappel par courriel

2.2 Contenu des sprints

Sprint 2 – Discipline, finances et IA de base

Le deuxième sprint s'est concentré sur les modules de discipline d'étude, de gestion budgétaire et sur une première version de l'intelligence artificielle.

Il incluait :

- Le calcul de priorité des événements et la barre de progression du travail réalisé.
- Le placement automatique de blocs d'étude et la vue quotidienne « Plan d'étude ».
- Le chronomètre d'étude (démarrage, pause, reprise, fin de session, prolongation).
- Le suivi de progression académique par graphiques.
- La gestion complète des revenus et dépenses, le suivi budgétaire mensuel et les alertes en cas de dépassement de budget.

Sprint 3 – IA avancée et intégration complète

Le dernier sprint portait sur l'enrichissement des fonctionnalités d'IA et sur l'intégration globale avec des tests.

Les développements prévus comprenaient :

- La re-priorisation intelligente des tâches et l'apprentissage adaptatif (ajustement des durées selon la performance).
- Le placement optimal des blocs d'étude en fonction des contraintes personnelles (emploi, fatigue, etc.).
- L'ajout de conseils de bien-être, de notifications motivationnelles et de questions de suivi personnel.
- L'analyse avancée des finances par l'IA et la génération de rapports globaux de fin de semestre, combinant progression académique, situation financière et recommandations personnalisées.

3. CONTENU DU SPRINT ACTUEL

3.1 Cas d'utilisation implantés dans le sprint (30%)

Cas d'utilisation – Ajouter revenus
Acteurs principaux : les utilisateurs (étudiants)
Brève description : Ce cas d'utilisation décrit le processus par lequel un utilisateur enregistre une nouvelle transaction de revenu (entrée d'argent) dans le système.
Préconditions : L'utilisateur est connecté à son compte.
P**ostconditions** :

- Un nouvel enregistrement de revenu est créé et stocké dans la base de données
- La liste des revenus est mise à jour sur l'interface.
- Le total des revenus et le solde global sont recalculés et affichés.

Scénario Principal : Ajout d'un nouveau revenu

1. L'utilisateur lance le cas d'utilisation (se positionne sur l'onglet "Revenus").
2. Le système affiche le formulaire de saisie des transactions.
3. L'utilisateur saisit les informations requises dans le formulaire:
   a. Type de revenu (ex. Salaire, Bourse)
   b. Montant ($)
   c. Date
   d. Récurrence (ex. unique, hebdomadaire, mensuel)
   e. Description (optionnel)
4. L'utilisateur clique sur le bouton **« AJOUTER »**.
5. Le système vérifie la validité des données saisies, notamment que le montant est valide et positif.
6. Le système appelle la fonction d'API pour créer l'enregistrement.
7. Le système met à jour la liste des revenus affichée et réinitialise le formulaire.
8. Le système met à jour les totaux affichés dans le tableau de bord (Revenus, Dépenses, Solde).
9. Le système confirme l'ajout du revenu et revient à l'état initial.

Scénarios Alternatifs
Champ obligatoire manquant ou invalide :

1. Si le Montant n'est pas saisi ou est invalide (<= 0).
2. Le système stoppe l'enregistrement et affiche un message d'alerte : « Montant invalide ».
3. Le scénario reprend à l'étape 3 (saisie de l'utilisateur).

Modification d'un revenu existant (édition) :

1. Si l'utilisateur a cliqué sur **Modifier** un élément existant (editingId est actif).
2. Le système appelle la fonction d'API pour mettre à jour l'enregistrement existant au lieu de le créer.
3. Le système met à jour la liste des revenus (revenues.map) avec les données modifiées.
4. Le scénario reprend à l'étape 7 (réinitialisation et mise à jour des totaux).

Cas d'utilisation – Ajouter dépenses
Acteurs principaux : L'utilisateur (Gestionnaire de finances, Particulier, etc.)
Brève description : Ce cas d'utilisation décrit le processus par lequel un utilisateur enregistre une nouvelle transaction de dépense (sortie d'argent) dans le système.
Préconditions :

- L'utilisateur est connecté à son compte.
- L'onglet **Dépenses** est actif.

Postconditions :

- Un nouvel enregistrement de dépense est créé et stocké dans la base de données.
- La liste des dépenses (expenses) est mise à jour sur l'interface.
- Le total des dépenses (totalDepenses) et le solde global (solde) sont recalculés et affichés.

Scénario Principal : Ajout d'une nouvelle dépense

1. L'utilisateur se positionne sur l'onglet **"DÉPENSES"**.

2. Le système affiche le formulaire de saisie des transactions, adapté pour les dépenses.
3. L'utilisateur saisit les informations requises dans le formulaire :
   a. Type de dépense (ex. Loyer, Alimentation)
   b. Montant ($)
   c. Date
   d. Récurrence (ex. unique, mensuel)
   e. Description (optionnel)
4. L'utilisateur clique sur le bouton **« AJOUTER »**
5. Le système vérifie la validité des données saisies, notamment que le montant est valide et positif.
6. Le système appelle la fonction d'API pour créer l'enregistrement de dépense (createDepense).
7. Le système met à jour la liste des dépenses affichée et réinitialise le formulaire.
8. Le système met à jour les totaux affichés dans le tableau de bord (Revenus, Dépenses, Solde).
9. Le système confirme l'ajout de la dépense et revient à l'état initial.

Scénarios Alternatifs

Champ obligatoire manquant ou invalide :

1. Si le Montant n'est pas saisi ou est invalide (<= 0).
2. Le système stoppe l'enregistrement et affiche un message d'alerte : « Montant invalide ».
3. Le scénario reprend à l'étape 3 (saisie de l'utilisateur).

Modification d'une dépense existante (édition) :

1. Si l'utilisateur a cliqué sur **Modifier** un élément existant (editingId est actif).
2. Le système appelle la fonction d'API pour **mettre à jour** l'enregistrement existant (updateDepense) au lieu de le créer.
3. Le système met à jour la liste des dépenses (expenses.map) avec les données modifiées.
4. Le scénario reprend à l'étape 7 (réinitialisation et mise à jour des totaux).

Voici l'ensemble des fonctions système utilisées pour le cas d'utilisation : ajouter les revenus

createRevenue () Crée un nouvel enregistrement de revenu dans la base de données via l'interface de l'API**.**
updateRevenue() Met à jour un enregistrement de revenu existant (lorsque l'utilisateur est en mode édition).
listRevenues() Récupère tous les enregistrements de revenus au lancement de la page pour les afficher dans la liste.
parseFloat() Convertit la chaîne de caractères du montant saisi par l'utilisateur en un nombre à virgule pour le calcul et la validation.
resetForm() Réinitialise tous les champs du formulaire de saisie à leurs valeurs initiales après un enregistrement réussi ou lors d'une annulation.
toFixed() Formate les montants numériques pour qu'ils s'affichent avec exactement deux décimales, représentant la valeur monétaire.
getLabel() Traduit les valeurs techniques (ex. 'salaire', 'unique') en labels lisibles par l'utilisateur

Signature	Promise<Revenue> createRevenue(payload: RevenueInput)
Description	Crée un nouvel enregistrement de revenu dans la base de données via l'API.
Références	Cas d'utilisation : Ajout d'un revenu dans la page "Revenus".

Entrées	• **payload**
	  ○ Type : RevenueInput
	  ○ Signification : Données saisies par l'utilisateur (montant, catégorie, type, date, etc.)

Sorties	**Revenue** • Revenu nouvellement créé et renvoyé par la base.
Exceptions	1. Erreur si les données sont invalides. 2. Erreur si l'enregistrement échoue (connexion, contrainte DB).
Pré-conditions	Les champs obligatoires doivent être remplis et valides.
Post-conditions	Le revenu est enregistré et peut être ajouté à la liste affichée.
Notes	Utilisé lorsque l'utilisateur valide un nouveau revenu.

Signature	Promise<Revenue> updateRevenue(id: string, payload: Partial<Revenue>)
Description	Met à jour un revenu existant lorsque l'utilisateur est en mode édition.
Références	Cas d'utilisation : Modification d'un revenu existant dans "Revenus".
Entrées	**id** : Identifiant du revenu **payload** : Champs à mettre à jour
Sorties	• **Revenue** mis à jour
Exceptions	1. Erreur si l'id est introuvable. 1. Erreur si la mise à jour échoue.
Pré-conditions	L'id doit correspondre à un revenu existant.
Post-conditions	Le revenu est modifié dans la base et renvoyé à l'UI.
Notes	Utilisé lors de l'édition d'un revenu dans le formulaire.

Signature	Promise<Revenue[]> listRevenues()
Description	Récupère tous les revenus existants pour affichage.
Références	Cas d'utilisation : Chargement initial de la page "Revenus".
Entrées	aucune
Sorties	• **Revenue[]** Liste complète des revenus.
Exceptions	1. Erreur de connexion ou de récuperation
Pré-conditions	Base accessible.
Post-conditions	Les revenus sont affichés dans la liste de l'UI.
Notes	Appelé automatiquement au chargement de la page.

3.2 Modèles de classes

3.2.1

Nom	Evaluation
Description	Représente une évaluation ou une note obtenue dans un cours spécifique (ex. : devoir, examen partiel, quiz, travail final, etc.). Permet le calcul automatique de la moyenne pondérée et le suivi de la progression académique.
Attributs	
	<uuid> <id> : Identifiant unique de l'évaluation
	<uuid> <user_id> : Identifiant de l'utilisateur propriétaire de l'évaluation (clé étrangère vers la table users)
	<uuid> <course_id> : Identifiant du cours auquel appartient cette évaluation (clé étrangère vers la table courses)
	<text> <title> : Titre ou nom de l'évaluation (ex. : « Midterm Exam », « Devoir 2 », « Quiz Semaine 4 »)
	<numeric> <weight> : Poids/pondération de l'évaluation dans la note finale du cours (ex. : 0.25 pour 25 %, 30 pour 30 %, etc.)
	<numeric> <score> : Note obtenue par l'étudiant sur cette évaluation (ex. : 87.5, 14/20, null si pas encore rendue)
	<timestamptz> <created_at> : Date et heure de création de l'entrée (automatique)
	<timestamptz> <updated_at> : Date et heure de la dernière modification (automatique)

Nom	Depences
Description	Enregistre chaque mouvement financier (revenu ou dépense) de l'étudiant dans le cadre de ses études ou de sa vie quotidienne. Utilisée pour le suivi budgétaire, les alertes de dépassement et les rapports semestriels (en CAD $).
Attributs	
	<uuid> <id> : Identifiant unique de la transaction
	<uuid> <user_id> : Identifiant de l'utilisateur propriétaire (clé étrangère vers users)
	<text> <type> : Type de transaction (« revenue » ou « expense »)
	<numeric> <montant> : Montant de la transaction en dollars canadiens (CAD $), positif pour les revenus, positif également pour les dépenses (la distinction se fait via le champ type)
	<date> <date> : Date effective de la transaction (ex. : date de réception de bourse, date de paiement des frais)
	<text> <description> : Libellé détaillé (ex. : « Bourse d'excellence trimestrielle », « Frais de scolarité Winter 2026 », « Achat manuel CPSC 413 »)
	<text> <recurrence> : Indique si la transaction est ponctuelle ou récurrente (valeurs possibles : « none », « monthly », « term », « yearly ») – utilisé pour les prévisions budgétaires
	<timestamp> <created_at> : Date et heure de saisie de la transaction dans l'application

Nom	Revenues
Description	Table centrale du module de gestion budgétaire. Elle regroupe tous les mouvements d'argent (revenus et dépenses) de l'étudiant, permettant le calcul du solde, les alertes de

	dépassement et les rapports financiers en CAD $.
Attributs	
	<uuid> <id> : Identifiant unique de la transaction
	<uuid> <user_id> : Identifiant de l'étudiant propriétaire (clé étrangère → users)
	<text> <type> : Type de mouvement (« revenue » ou « expense »)
	<numeric> <montant> : Montant en dollars canadiens (toujours positif ; le sens est donné par le champ type)
	<date> <date> : Date effective de la transaction (ex. : 2025-11-15)
	<text> <description> : Libellé complet (ex. : « Bourse Entrance Scholarship », « Paiement frais de scolarité Fall 2025 », « Remboursement prêt étudiant »)
	<text> <recurrence> : Fréquence de récurrence (« none », « monthly », « quarterly », « term », « yearly ») – utilisée pour les prévisions et rappels automatiques
	<timestamp> <created_at> : Date et heure de saisie dans l'application (sans fuseau horaire – heure locale de l'utilisateur)

3.2.2

Nom	Ajouter / Modifier un revenu
Description	Composant modal réutilisable (Ionic + React) pour l'ajout ou la modification d'un revenu dans le module Finances. Gère la validation, l'appel Supabase et la mise à jour en temps réel des totaux.
Attributs	
	open (boolean) : Contrôle l'ouverture/fermeture du modal
	onClose (function)Callback exécutée à la fermeture du modal
	editingRevenue (objectnull)
	userId (uuid)Identifiant de l'utilisateur connecté
	type (string)Valeur fixe "revenue" (le même composant gère aussi les dépenses avec "expense")
	amount (string) : Montant saisi (en CAD $) – converti en number avant insertion
	revenueType (string) : Type de revenu sélectionné (ex. Bourse, Job, Aide familiale…)
	date (string) : Date effective de la transaction (format YYYY-MM-DD)
	recurrence (string) : Fréquence : "none"
	description (string) : Champ libre pour détails supplémentaires
Méthodes	
	handleAmountChange() : Synopsis : formate et valide le montant (accepte uniquement chiffres et décimaux, refuse ≤ 0)
	validateForm() : Synopsis : vérifie que tous les champs obligatoires sont remplis et valides ; retourne true/false
	handleSubmit() : Synopsis : si editingRevenue → appel Supabase update, sinon insert ; recalcule les totaux via callback parent ; toast succès/échec ; ferme le modal
	resetForm() : Synopsis : réinitialise tous les champs et erreurs après ajout/modification réussie
	handleClose() : Synopsis : annule les modifications en cours, réinitialise le formulaire et appelle onClose()

4. PROCHAIN SPRINT (1 À 2 PAGES, MAX) (35%)

4.1 Revue du sprint (30%)

4.1.1 Répartition des tâches

La répartition des tâches du Sprint 2 s'est faite selon les trois piliers fonctionnels du projet : discipline d'étude, gestion financière étudiante et intelligence artificielle de base. Toutes les fonctionnalités prévues ont été livrées et sont pleinement opérationnelles.

- **KUBARIEVA, Oryna**
  - **Issue #12** – **Calcul de priorité d'événement basique (IA v1)** : conception et intégration complète de la formule de priorité (pondération de l'évaluation × coefficient + jours restants avant échéance), avec affichage clair de l'indice et tri automatique des tâches.
  - **Issue #13 – Barre de progression :** implémentation d'une barre dynamique et responsive affichant en temps réel le pourcentage d'avancement (travail réalisé / travail total), avec code couleur adapté.
- **MUNYANEZA, Ursule Joana**
  - Issue #14 – Placement automatique de blocs d'étude (25/45/60/90 min) : algorithme de génération intelligente qui insère automatiquement les blocs dans les créneaux libres du calendrier.
  - Issue #15 – Vue "Plan d'étude" : développement d'une vue quotidienne minimaliste listant les tâches du jour avec titre, catégorie, durée et bouton « Commencer » (passage instantané en « En cours », une seule tâche active à la fois).
- **KENFACK NANDJOU, Steve Marvyn**
  - Module complet – Gestion du chronomètre d'étude (Issues #16 à #19) :
  - Allocation précise du temps par bloc d'étude
  - Fonctions Pause/Reprise fluides et persistantes
  - Arrêt automatique à la fin du temps imparti avec signal sonore/vibration
  - Gestion avancée de fin de session : notification modale avec message de félicitations et options de prolongation (+5, +10, +15 min ou temps personnalisé).
- **THIAM, Demba Aziz**
  - Issue #20 – Suivi de progression académique : création de graphiques journaliers et hebdomadaires (camemberts et courbes) illustrant le pourcentage de tâches complétées.
  - Issue #25 – Suivi budgétaire mensuel : tableau récapitulatif avec agrégation automatique des revenus/dépenses par catégorie et affichage du solde en CAD $.
- **SOKOUDJOU SOKAMTÉ, Romuald**
  - Module – Gestion des revenus et dépenses :
  - Issues #21 & #22 : ajout, édition et suppression des revenus (bourse, job, aide familiale, etc.)
  - Issues #23 & #24 : ajout, édition et suppression des dépenses liées aux études (frais de scolarité UNICAL, livres, UPass, logement, etc.)
  - Issue #26 – Alerte de dépassement de budget : système de seuils (90 % et 100 %), notification prioritaire fixe en haut d'écran avec message clair en CAD $ et actions rapides (voir détail, ajuster budget, masquer temporairement).

4.1.2 Revue technique

Le Sprint 2 a porté sur le cœur fonctionnel de la discipline d'étude et du suivi financier : **calcul intelligent de priorité (IA v1)**, **barre de progression**, **placement automatique des blocs d'étude**, **vue « Plan d'étude » quotidienne** avec bouton Commencer, **chronomètre** complet avec pause/reprise et prolongation, ainsi que **le module complet** de gestion des revenus/dépenses (en CAD $) avec alertes de dépassement budgétaire.

Fonctionnalités complétées :

- **Calcul de priorité d'évènement basique (IA v1)** : indice automatique calculé selon l'importance et l'urgence.
- **Barre de progression** : affichage visuel en % de l'avancement global
- **Placement automatique de blocs d'étude (25/45/60/90 min)** : insertion intelligente de sessions 25/45/60/90 min dans les créneaux libres du calendrier.
- **Vue "Plan d'étude" avec liste des tâches du jour** : interface quotidienne épurée listant les tâches du jour avec bouton « Commencer » et suivi instantané du statut
- **Module : Gestion chronomètre d'étude (timer par bloc)** : démarrage, pause/reprise, arrêt automatique, notification de fin et options de prolongation (+5/+10/+15 min).
- **Module : Gestion des revenus et dépenses** : ajout/édition/suppression des entrées

Fonctionnalités partiellement complétées ou non terminées :

- **Suivi de progression académique – Issue #20**

Non réalisé dans ce sprint, car :

- Priorité donnée aux fonctionnalités essentielles (chronomètre, vue quotidienne).
- Les données de complétion des tâches devaient être stabilisées avant de pouvoir générer des graphiques fiables.

- **Module Notifications – Issue #26**

La partie « alerte visuelle » a été livrée, mais les **notifications push** n'ont pas été développées dans ce sprint :

- Configuration et tests supplémentaires nécessaires
- Travail prévu pour le prochain sprint.

Ce qui a bien fonctionné :

- La possibilité de corriger les erreurs identifiées dans le Sprint 1 et qui n'avaient pas pu être corrigées avant la remise.
- L'ajout des pages qui n'avaient pas été réalisées entièrement dans le Sprint 1 et qui ont pu être finalisées durant le Sprint 2.

- L'uniformisation du style entre les pages afin de rendre l'application plus cohérente et esthétiquement agréable.

Ce qui a moins bien fonctionné :

- La gestion du temps, même si globalement bien réalisée, a été impactée par deux fonctionnalités qui n'ont pas pu être suffisamment élaborées et retravaillées.
- Des difficultés lors de la fusion des branches ont ralenti les tests d'intégration du système de notifications.
- De légers retards de développement ont été causés par la dépendance entre les branches : il fallait s'assurer que les branches et les pages concernées soient correctement liées avant de poursuivre le développement.

Obstacles rencontrés et solutions adoptées :

- **Obstacle :** réussir les tests afin de s'assurer que toutes les fonctionnalités soient correctement liées entre elles avant de pouvoir avancer efficacement.

- **Solution :** mettre en place une stratégie de fusion plus structurée (pull requests, revue de code, tests automatisés), coordonner les branches en définissant un ordre clair de développement pour éviter les dépendances bloquantes, et communiquer régulièrement au sein de l'équipe afin d'identifier rapidement les conflits et de les résoudre avant qu'ils ne ralentissent le sprint.

4.1.3 Revue gestion

Le Sprint 2 a permis d'enrichir l'application avec des fonctionnalités plus avancées, notamment la gestion des rappels, l'amélioration du calendrier, l'affichage du budget. Cette rétrospective vise à évaluer le travail réalisé, la collaboration, les processus et les outils, afin de préparer un **Sprint 3** plus fluide et plus efficace.

Évaluation des personnes et relations

- La collaboration entre les membres a été **globalement bonne**, avec une coordination efficace malgré la complexité des nouvelles fonctionnalités.
- La communication a permis de **résoudre plusieurs blocages**, même si certains retards sont survenus à cause de dépendances entre branches.
- Les membres ont montré une **bonne capacité d'adaptation** : entraide lors des tests, échanges réguliers pour résoudre les conflits de code et ajustements rapides pour finaliser les pages incomplètes du Sprint 1.

Évaluation des processus

- La planification du Sprint 2 a permis de prioriser les fonctionnalités essentielles de productivité

(chronomètre, vue quotidienne, rappels, alertes).

- Le suivi des issues et du backlog a facilité l'organisation des tâches, même si certaines dépendances ont retardé l'intégration finale.
- Les tests d'intégration ont été ralentis par des conflits de branches et par la nécessité d'assurer la cohérence entre les modules avant la fusion.

Évaluation des outils

- Git, GitHub et la gestion des branches ont bien soutenu le développement, mais les conflits répétés ont montré la nécessité d'un processus de fusion plus strict.
- Les tests locaux ont permis de valider l'essentiel des nouvelles fonctionnalités avant intégration, mais les tests liés aux notifications push n'ont pas pu être finalisés faute de configuration backend complète.
- L'amélioration du style et de l'uniformité visuelle a été facilitée par les outils de prévisualisation, rendant l'application plus cohérente.

Points positifs

- Livraison des fonctionnalités clés du Sprint 2 : rappels, affichage du budget, barre d'alerte de dépassement, corrections issues du Sprint 1.
- Meilleure cohérence visuelle entre les pages grâce au travail sur l'interface.
- Bonne répartition des tâches et capacité de l'équipe à résoudre rapidement les problèmes rencontrés.

Points à améliorer

- Finaliser le système complet de notifications (push navigateur + mobile), non réalisé dans ce sprint.
- Améliorer la gestion des dépendances entre branches pour limiter les retards d'intégration.
- Optimiser la cohérence des données (statuts des tâches, complétion) pour préparer le module de suivi académique du prochain sprint.
- Continuer d'améliorer l'affichage mobile, où certains composants débordent encore ou manquent de responsivité.

Plan d'action pour le Sprint suivant

- Finaliser le système de notifications et terminer la configuration backend (Web Push API + Edge Functions).
- Mettre en place une procédure stricte de fusion : pull requests, revue systématique de code, tests obligatoires avant intégration.
- Stabiliser totalement les données de tâches afin de permettre le développement du module de progression académique (graphiques).
- Poursuivre l'amélioration visuelle et corriger les problèmes d'affichage mobile.
- Finaliser les pages partiellement réalisées et vérifier la cohérence complète entre les modules avant

le passage à des fonctionnalités avancées.

4.2 Plan pour prochain sprint (5%)

Le Sprint 3 se concentre sur l'intégration de modules avancés d'aide à l'étude, d'intelligence adaptative et d'analyse financière. L'objectif est d'améliorer la personnalisation de l'expérience étudiante, d'optimiser la gestion du temps et de générer des recommandations ciblées basées sur le comportement et les progrès de l'utilisateur.

Le Sprint 3 se concentre sur l'intégration de modules avancés d'aide à l'étude, d'intelligence adaptative et d'analyse financière. L'objectif est d'améliorer la personnalisation de l'expérience étudiante, d'optimiser la gestion du temps et de générer des recommandations ciblées basées sur le comportement et les progrès de l'utilisateur.

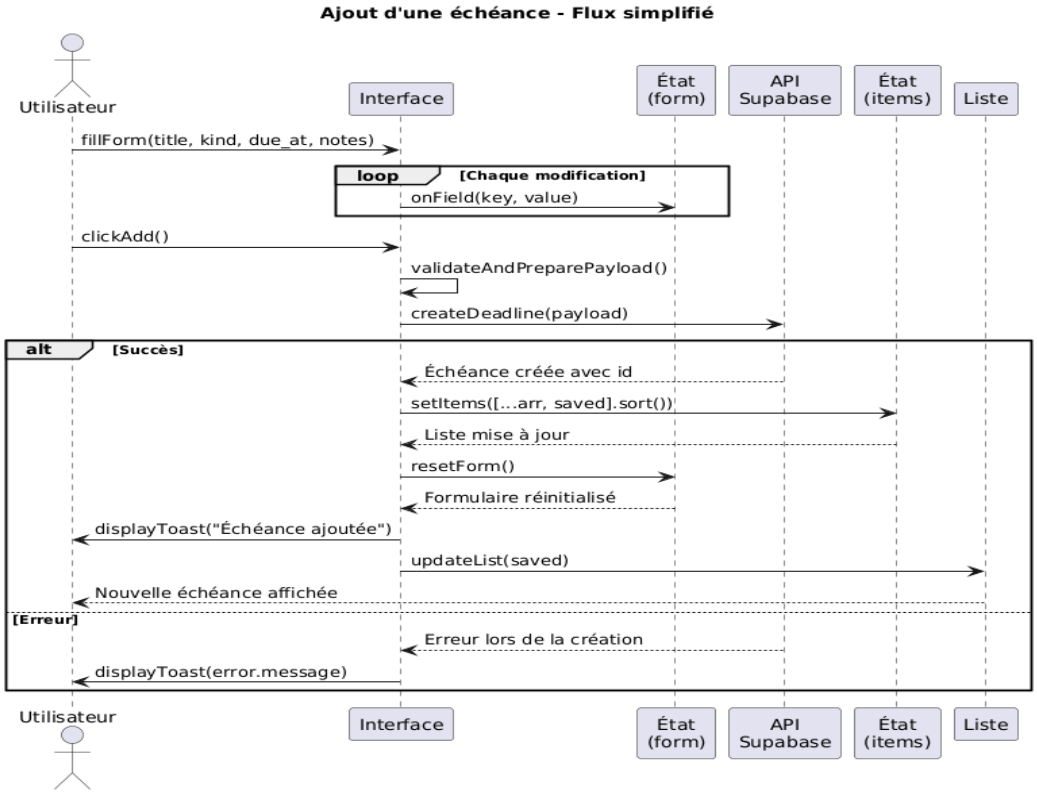
Cas d'utilisation prévus pour le Sprint 3

1. **Re-priorisation automatique des tâches (IA – niveau 2)**
    - Ajuster l'ordre des tâches selon leur importance, leur urgence et les retards accumulés.
    - Recalculer automatiquement les priorités lorsque certaines tâches ne sont pas exécutées à temps.
2. **Apprentissage adaptatif des sessions d'étude**
    - Adapter la durée des blocs d'étude selon les performances de l'utilisateur.
    - Allonger les sessions en cas de retard fréquent, ou les réduire en cas de surcharge ou d'efficacité élevée.
3. **Placement intelligent avancé des blocs d'étude**
    - Générer des blocs optimisés en tenant compte des contraintes personnelles (emploi, sport, fatigue, disponibilités).
    - Réorganiser automatiquement les blocs lorsque le calendrier change ou lorsqu'un bloc est manqué.
4. **Conseils personnalisés pour l'équilibre de vie et le bien-être**
    - Proposer des recommandations ciblées (révision d'un chapitre clé, pause active, méditation, hydratation).
    - Détecter les périodes de surcharge et suggérer des ajustements.
5. **Notifications POP-UP motivantes**
    - Envoyer des messages d'encouragement lors des sessions d'étude ou en cas de baisse d'activité.
    - Rappeler les bonnes pratiques (respirer, boire de l'eau, courte pause).
6. **Mini-questions de bien-être**
    - Poser de courtes questions quotidiennes (ex. as-tu bu suffisamment d'eau ? as-tu fait tes pas ?).
    - Adapter les recommandations selon les réponses reçues.
7. **Analyse financière intelligente (IA – niveau 2)**
    - Examiner les dépenses et le budget pour générer des suggestions adaptées.
    - Identifier les catégories où les dépenses sont trop élevées ou irrégulières.
8. **Rapport global académique de fin de semestre**
    - Générer un résumé incluant le pourcentage total de complétion des cours et révisions.
    - Présenter les tendances d'étude (heures, cohérence, progrès).
9. **Rapport global financier de fin de semestre**
    - Régrouper revenus, dépenses, catégories, dépassements et tendances.
    - Fournir une vue synthétique et structurée du comportement financier mensuel ou semestriel.
10. **Recommandations académiques et financières pour les prochaines sessions**

- Proposer des stratégies pour mieux gérer les études, éviter la surcharge et améliorer la productivité.
- Offrir des conseils budgétaires personnalisés pour optimiser les dépenses futures.

5. ANNEXES (10%)

Inclure :



1) le product backlog en date d'aujourd'hui (fin du sprint courant/début du prochain sprint)
2) Le sprint backlog du sprint 2
3) Le sprint backlog pour le sprint 3