**Collège de Rosemont**

**PROJET  
Le Cahier des Charges**

**Par  
  
gr. 445**

**Travail remis à M. Sylvain Labranche dans le cadre du cours  
DÉVELOPPEMENT DES APPLICATIONS INFORMATIQUES - PROJET  
420-B34-RO**

**Pour le sprint 2, à remettre le 9 décembre**

# Présentation du projet

## Demande du client

Pour ce sprint, Le client voulait l’intégration d’une interface graphique pour pouvoir visualiser les feuilles de temps des employés et le faire valider, et il veut que les employés puissent entrer les temps de travail directement sur l’application. De plus il veut un login pour la fenêtre du Patron.

Le projet utilise toujours des fichier JSON pour gérer les feuilles de temps. Il veut diviser les employés normaux dans 2 catégories, employés de production et employés d’exploitation. Il veut que les employés d’exploitations travaillent au moins 38 heures et veux aussi que les employés de l’administrations travaillent 36 heures au bureau minimum.

Le client veut qu’une journée de travail ne dépasse pas 24 heures, et qu’on gère les journées de maladie et les journées fériés comme une journée de 7 heures payés soit 420 minutes.

## Explication du projet

Pour notre projet, on choisit de faire une application web contenant les interfaces graphiques demandés, d’abord on a le menu qui permet de choisir si l’utilisateurs est un patron ou employé, dans le cas du patron l’application mène à une fenêtre de connexion ou il devra entrer son mot de passe et son nom d’utilisateur.

Après le login du Patron, on ouvre l’interface graphique qui permet de charger les feuilles de temps (Fichiers.JSON) et de les valider, après les avoir validé, le patron peut les accepter ou refuser, les feuilles sont envoyé à son dossier correspondant. Si on a choisi la partie employée dans le menu principal, l’interface qui permet à l’employé d’entrer ses heures de travail s’affiche.

Dans l’interface employée, on a des champs de saisie pour entrer les informations tel que son numéro d’employé, l’année, le numéro de la semaine. Une espace pour entrer le code de projet et le temps accordé à ce projet pour chaque jour de la semaine ainsi que la possibilité de cocher la case d’une journée de maladie s’il y a lieu, ceci empêche la saisie du code de projet et le temps de cette journée et le mets à 999 et 420 respectivement.

Pour gérer une journée férié, après la saisie de l’année et le numéro de la semaine, l’application auto rempli le jour férié correspondant au numéro de la semaine Ex : Semaine 51 = Noel (code = 998, minutes = 420).

Le système empêche aussi la saisie lorsqu’on saisie plus de 24 heures (1440 minutes) dans une journée.

On ajoute à notre projet la gestion de 2 types d’employés différents ceux d’exploitation et ceux de production, tous les employés sont différentiés à partir de leurs codes :

Les employés de l’administration, code < 1000;

Les employés de production, code >= 1000 mais code <2000;

Les employés d’exploitation, code >= 2000;

De plus, les employés d’exploitation doivent désormais, travailler au moins 38 heures par semaine soit bureau soit télétravail, et on ajoute aussi la nouvelle validation qui vérifie si les employés de l’administrations font maintenant 36 heures de travail au bureau par semaine.

## Explication des fonctionnalités souhaitées

* La gestion de congés de maladie;
  + Lorsque l’employé utilise l’interface pour entrer son temps de travail, il peut cocher la case de journée de maladie ou appuyer sur le bouton, qui rends automatiquement le code projet à 999 et le temps de travail à 7 heures, soit 420 minutes pour cette journée et rends impossible d’ajouter plus des projets et du temps de travail.
  + Le temps d’une journée de maladie va être considéré comme du temps au bureau.
  + L’employé n’aura pas la possibilité de prendre un congé de maladie la fin de semaine et pourtant cette option n’est pas présente dans l’interface.
* La gestion de congés fériés;
  + Lorsque l’employé saisie l’année et le numéro de la semaine, l’application va automatiquement déterminer si c’est une semaine dont il y a un jour férié officiel du Canada et auto rempli la journée et l’employé peut charger la journée comme une journée férié (code projet = 998 et temps = 420 minutes).
  + L’employé aura la possibilité d’ajouter plus de temps de travail seulement si c’est du télétravail (Code projet > 900) dans une journée.
  + Le temps d’une journée férié sera considéré comme du temps au bureau.
  + L’employé ne pourra pas charger une journée férié la fin de semaine.
* La gestion de deux nouveaux types d’employés;
  + Les employés normaux sont divisés en 2, employés de production et employés d’exploitation.
  + Les employés de production sont distingués avec leur code d’employé qui serait entre1000 et 1999.
  + Les employés d’exploitation sont distingués avec leur code d’employé qui serait 2000 et plus.
  + Les employés d’exploitation doivent travailler un minimum de 38 heures par semaine soit bureau soit télétravail.
* La gestion des heures de travail des employés de l’administration;
  + Les employés de l’administration doivent désormais faire 36 heures au bureau par semaine.
* La gestion de la restriction des heures de travail par jour;
* Si l’employé fait une erreur lorsqu’il rempli sa feuille de temps il peut recommencer;
  + Lorsque l’employé saisie le temps et code de projet par jour, l’application l’empêche de continuer à en ajouter lorsque le temps dépasse 24 heures soit 1440 minutes.
* La gestion de feuilles de temps.
  + Dès l’interface patron, il est possible de charger une feuille de temps, sauvegardé dans le pc ou faite par l’application elle même,en format JSON une fois elle se trouve dans le dossier C:\FeuillesTemps, qui est créer automatiquement lorsqu’on utilise l’application.
  + Une fois la feuille de temps validée, les informations sont affichés à l’écran pour faciliter la lecture.
  + Fonctionnalité bonus : On a le choix d’accepter une feuille de temps ou de la refuser, ceci va trier les feuilles de temps en deux différents dossier.

Répartition des tâches à chacun des membres de l’équipe

## Product owner

Sebastian est la Product Owner. C’est lui qui a préparé le cahier de charges, et conçu la vision globale du projet à partir du document du sprint 2. Il a préparé les UML et programmé l’interface graphique, Soit le login, la page employé et la page manager, et le menu principale, Soit le front-end de l’application à fin donner une structure de base au projet à partir de sa vision, et il a collaboré avec le reste de l’équipe avec le développement du projet.

## Scrum master

Anthony est la Scrum master. C’est lui qui a établi les user stories avec le Product Owner, il a aussi divisé les tâches pour le projet, il organise et prends en note la date des mêlées Scrum ainsi que les points importants abordés dans ces rencontres. Il a géré le backlog tout au long du sprint 2 et a collaboré avec le reste de l’équipe pour le développement du projet. Il a identifié les bugs qui se sont produit au cours du développement du projet et les a documentés pour que l’équipe SCRUM puissent travailler là-dedans. Il a implémenté la fonctionnalité bonus du triage des feuilles de temps acceptés et non acceptés.

## Équipe de développement

Pour le sprint 2, l’équipe de développement est composée par Guillaume Tremblay, Martin Forget et Martin Vézina. Tout d’abord, pour ce sprint, Martin Vézina et Martin Forget ont cédé leur poste de Product Owner et Scrum Master à Sebastian Toledo et Anthony Ménard-Boucher. À l’aide de Guillaume ils ont tous implémenté les nouvelles fonctionnalités que le client nous a présenté. Ils se sont occupés de transcrire les validations déjà existantes dans le sprint 1, dans l’interface employé, qui a pour but de restreindre les feuilles de temps erronées. De plus, Martin Vézina s’est occupé de vérifier et merger le code sur GitHub.

Langages et logiciels utilisés

On s’est tous mis d’accord d’utiliser le langage C# pour tous les sprints à date de ce projet car c’est le langage dont on est le plus à l’aise. On utilise I’IDE Visual Studio et on se sert du Framework .net plus précisément ASP.NET pour faire une application web avec des interfaces graphiques. Au cas où on aurait besoin d’implémenter une base de données relationnelle, on pourrait se servir de l’Entity Framework qui s’intègre facilement avec le projet développé en ASP.NET, ceci a été notre raisonnement lorsqu’on a choisi les technologies .Net.

Pareil comme dans le sprint précédant, on continue à utiliser Jira pour la gestion du projet et pour entreposer notre code on utilise GitHub

* C#
  + C’est un langage avec lequel nous sommes familiers
* ASP.NET
  + C’est un Framework qui Inclus HTML, CSS et Javascript et facilite la conception d’une application web
* GitHub et Git
  + C’est le logiciel qu’on utilise pour la gestion de versions et pour collaborer tous dans le même projet.
* Jira
  + Utile pour la gestion du projet.

Échéancier du sprint

On fait nos mêlés Scrum en moyenne de 3 fois par semaine, 2 fois dans le temps alloué au cours du projet soit les mercredis et les vendredis, et une autre fois souvent les Lundis, pour discuter sur notre progrès faite durant la fin de semaine, quand on a besoin de rencontres additionnels le Scrum master organise une rencontre quand l’équipe au complet pourra être présent.

Chacun d’entre nous travail sur le projet quand on peut vu qu’on a un horaire d’étude très chargé et condensé, on a une bonne communication. Quand quelqu’un fini une tâche en particulier on se tient au courant à travers notre groupe de Discord, pour que le reste du groupe puisse analyser le code ou contribuer ou ajouter une modification avant de considérer la tâche comme faite.

Diagramme de fenêtres de l’application

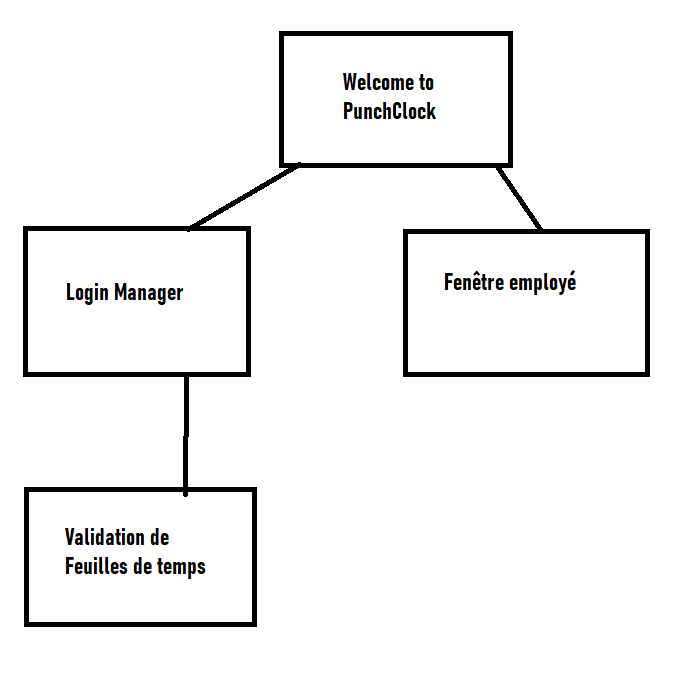
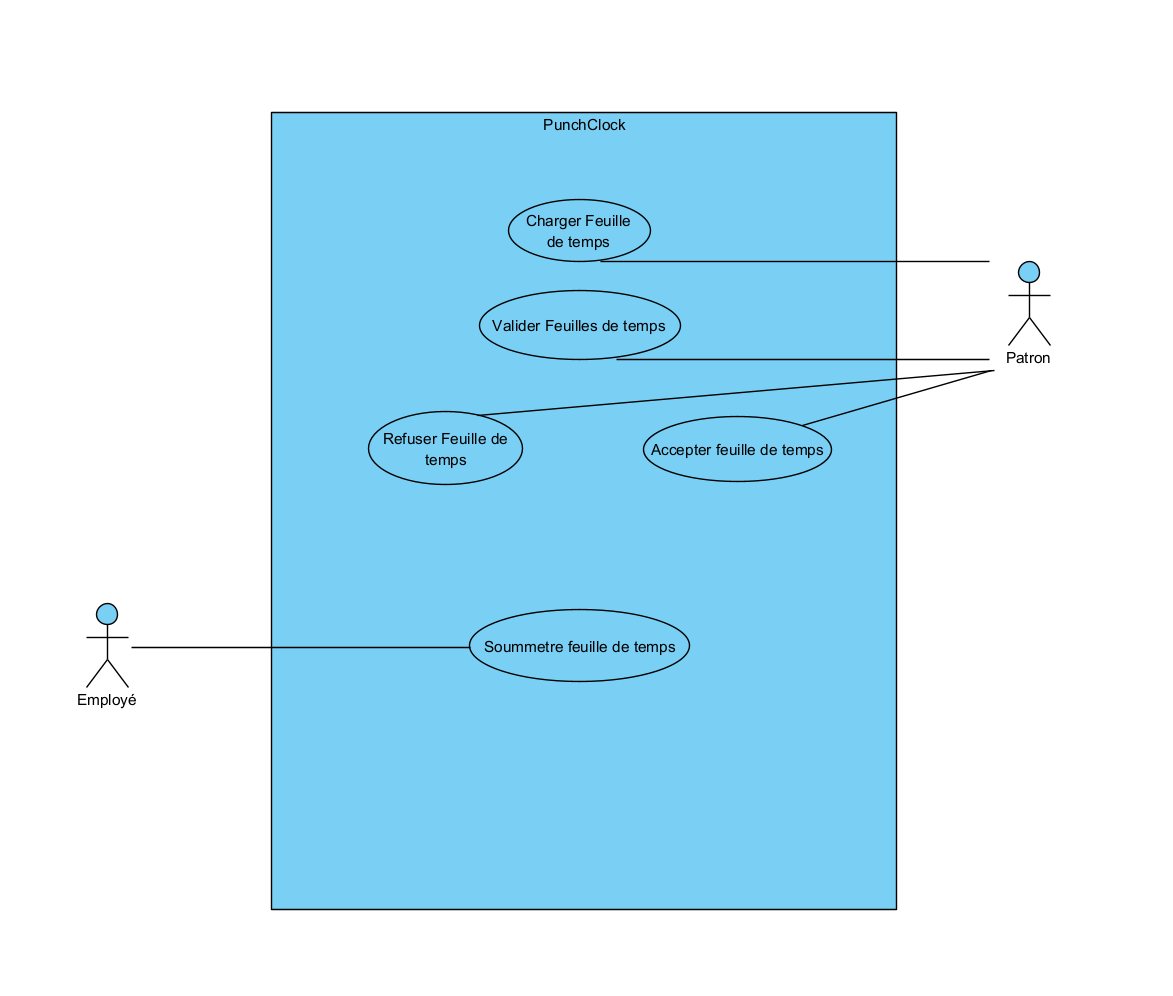


Diagram de cas d’utilisation et description narrative



**Nom :** Charger la feuille de temps(1).

**Acteurs :** Patron

**Objectif :** Charger une feuille de temps dans l’application provenant des documents sauvegardés dans l’ordinateur de l’utilisateur

**Règle d’initialisation :** Le patron ouvre son profil de ‘*Manager*’ dans l’application

**Scénario nominal :**

1. Le patron appui sur le bouton de chargement des feuilles de temps
2. Le patron choisi la feuille de temps qu’il souhaite faire valider

**Règle de terminaison :**La feuille est chargé dans l’application et est prête à être validée.

**Scénarios d’exception :**

1. Le système n’est pas fonctionnelle pour une raison donné Il n’ya pas de feuilles de temps sauvegardés dans l’ordinateur du patron
2. Interruption soudaine d’internet
3. Le patron ferme l’application abruptement pour une raison donnée

**Nom :** Valider feuille de temps (2).

**Acteurs :** Patron

**Objectif :** Valider la feuille de temps

**Règle d’initialisation :** Une fois la feuille de temps chargée, le patron appui sur le bouton valider

**Scénario nominal :**

1. L’application affiche toutes les informations de la feuilles de temps de l’employé
2. Le patron la lit et valide s’il respect les conditions du travail

**Règle de terminaison :** Le patron décide s’il accepte ou refuse la feuille de temps

**Scénarios d’exception :**

1. Le système n’est pas fonctionnel pour une raison donné

2. La feuille de temps ne respecte pas le format établi.

3. Le patron ferme l’application abruptement pour une raison donnée.

4. Interruption soudaine d’internet

**Nom :** Accepter la feuille de temps(3).

**Acteurs :** Patron

**Objectif :** Accepter la feuille de temps d’un employé

**Règle d’initialisation :** Le patron a pris une décision concernant la feuille de temps chargé préalablement.

**Scénario nominal :**

1. Le patron appui sur le bouton d’accepter la feuille de temps
2. S’il n’existe pas encore l’application crée un nouveau dossier à l’intérieur du dossier FeuillesTemps appelée FeuillesAccepte
3. L’application sauvegarde la feuille de temps chargé dans ce dossier.

**Règle de terminaison :** Le patron a sauvegardé la feuille de temps avec succés

**Scénarios d’exception :**

1. Le système n’est pas fonctionnel pour une raison donné

2. Le dossier FeuillesTemps n’existe pas encore sur le disque c.

3. Le patron ferme l’application abruptement pour une raison donnée.

4. Interruption soudaine d’internet

**Nom :** Refuser la feuille de temps(4).

**Acteurs :** Patron

**Objectif :** Refuser la feuille de temps d’un employé

**Règle d’initialisation :** Le patron a pris une décision concernant la feuille de temps chargé préalablement.

**Scénario nominal :**

1. Le patron appui sur le bouton refuser feuille de temps
2. S’il n’existe pas encore l’application crée un nouveau dossier à l’intérieur du dossier FeuillesTemps appelée FeuillesRefuser
3. L’application sauvegarde la feuille de temps chargé dans ce dossier.

**Règle de terminaison :** Le patron a sauvegardé la feuille de temps avec succés

**Scénarios d’exception :**

1. Le système n’est pas fonctionnel pour une raison donnée.

2. Le dossier FeuillesTemps n’existe pas encore sur le disque c.

3. Le patron ferme l’application abruptement pour une raison donnée.

4. Interruption soudaine d’internet.

**Nom :** Soumettre une feuille de temps(5).

**Acteurs :** Employé

**Objectif :** Soumettre une feuille de temps de travail pour une semaine de l’année

**Règle d’initialisation :** L’employé a accédé à la fenêtre de remplissage de feuille de temps de l’application

**Scénario nominal :**

1. L’employé choisi premièrement le numéro de la semaine dont il souhaite remplir la feuille de temps.
2. L’application rempli automatiquement les champs des dates exactes des jours de la semaine que l’employé a choisi
3. S’il y a un jour férié dans cette semaine, le système rempli automatiquement le temps de cette journée à 420 minutes et le projet à 998.
4. L’employé entre son numéro d’employé.
5. L’employé entre ses minutes par projet, dans la section de chaque jour de la semaine et il les ajoute au système
6. S’il y a lieu l’employé peut cocher la case de journée de maladie et le système rempli le projet à 999 et les minutes à 420.
7. Une fois toutes les informations remplis, l’employé appui sur le bouton Subit, pour soumettre la feuille de temps
8. Le système va déterminer s’il l’employé a respecté toutes les normes concernant les politiques des heures de travail de l’entreprise
9. Si la feuille de temps n’est pas conforme l’application averti l’employé
10. Une fois toutes les champs requis soient conformes et remplis, l’application sauvegarde la feuille de temps sous forme de fichier JSON dans le dossier FeuillesTemps.
11. Si le dossier FeuillesTemps n’existe pas, l’application le crée automatiquement au moment d’appuyer sur le bouton de soumission.

**Règle de terminaison :** La feuille de temps a été créé avec succès.

**Scénarios d’exception :**

1. Le système n’est pas fonctionnel pour une raison donnée

3. Le patron ferme l’application abruptement pour une raison donnée.

4. Interruption soudaine d’internet