***Cahier des charges***

1. ***Introduction :***

Développement d’une application qui effectuera la validation du temps travaillé par des employés d’une entreprise dans le respect des règles donnée par l’entreprise. Chaque employer doit remplir une feuille de temps hebdomadaire.

L’application doit avoir un menu qui permet de choisir entre l'interface patron ou employé. Le patron ne peut accéder qu’avec un mot de passe pour charger une feuille de temps.

Dans l'interface employé, l'utilisateur pourra entrer sa semaine de travail.

***Répartition des tâches à chacun des membres de l'équipe.***

|  |  |
| --- | --- |
| ***Le membre*** | ***Sa Tache*** |
| Belil Zakaria | * Responsable de projet « product owner ». * Répartition des taches pour les membres d’équipe. * Vérification d’évolution de travaille de chaque membre à chaque Daily Scrum Meeting. * Création du projet sur visuel studio * Vérification de diagramme de classe réaliser par l’équipe * Programmation |
| Chabane Rima | * « Scrum Master » aide les autres membres d’équipe dans la résolution des problèmes durant le travail sur le projet. * Modification de l’interface employé. * Programmation |
| Alimiraoussaid Samira | * Création des interfaces. * Faire le cahier des charges * Programmation |
| Lamouchi Mohamed Melek | * Programmation * Participation dans la création des interfaces. |

1. ***Technologies choisies pour le projet :***

Pour le projet on a choisi d’utiliser le langage de programmation C#.

1. ***Échéancier du sprint :***

***Le 18-04-2023(***durée de la rencontre 1h)

Première rencontre de l’équipe : explication de projet et répartition des taches pour chaque membre de l’équipe.

***Le 19-04-2023(***durée de la rencontre 1h :30mn***)***

Établissement des Exigences Fonctionnelles et non fonctionnelles,

Modification de diagramme de classe.

***Le 21-04-2023(***durée de la rencontre 20mn***)***

Vérification d’état d’avancement des taches reparti précédemment.

***Le 24-04-2023(***durée de la rencontre 1h***)***

Vérification d’état d’avancement des taches reparti précédemment. Et répartition des nouvelles taches.

***Le 28-04-2023(***durée de la rencontre 1h***)***

Vérification d’état d’avancement des taches reparti précédemment. Plus un travail de groupe sur le code.

***Le 01-05-2023(***durée de la rencontre 2h***)***

Rencontre pour vérification

***Le 04-05-2023(***durée de la rencontre 2h***)***

Rencontre pour vérification

1. ***Exigences Fonctionnelles***

Les employés doivent pouvoir saisir leur temps de travail pour différents projets pour chaque jour de la semaine.  
Les employés doivent pouvoir enregistrer leurs données de temps de travail.  
Les employés doivent pouvoir accéder et modifier  leur feuille de temps.  
Les patrons doivent pouvoir se connecter à l'interface spéciale en saisissant un mot de passe.  
Les données de temps de travail doivent être stockées localement sous format json.  
Les employés doivent pouvoir modifier les données de temps de travail précédemment enregistrées.  
Les employés doivent pouvoir visualiser leurs données de temps de travail précédemment enregistrées.  
Les patrons doivent pouvoir visualiser les données de temps de travail de tous les employés.  
Les employés et les patrons doivent pouvoir naviguer entre les différentes fenêtres de l'application.  
Les employés et les patrons doivent pouvoir fermer l'application.  
Les données de temps de travail doivent être sauvegardées en temps réel.  
L'interface utilisateur doit être conviviale et intuitive.  
Les messages d'erreur doivent être clairs et précis.  
Les informations sensibles, telles que les mots de passe, doivent être stockées de manière sécurisée.

1. ***Exigences non fonctionnelles :***
   1. Performance : L'application doit être capable de valider les feuilles de temps des employés en un temps raisonnable, même pour de grandes quantités de données.
   2. Sécurité : Les données des employés doivent être protégées contre les accès non autorisés.
   3. Fiabilité : L'application doit être robuste et capable de traiter les erreurs de manière efficace pour minimiser les temps d'arrêt et les pertes de données.
   4. Convivialité : L'application doit être facile à utiliser pour les utilisateurs, avec une interface utilisateur intuitive et conviviale.

Dispose d’un menu contextuel

1. ***Liste des diagrammes UML :***
   * + - ***Le diagramme de cas d’utilisation***

***Une image contenant diagramme

Description générée automatiquement***

* ***Le Diagramme de classe***

Une image contenant diagramme

Description générée automatiquement