**LOIC MARTIN**

**Epreuve PPE – LOIC MARTIN – MISSION 1 et MISSION 2**

**DEMARCHE**

**DEMARCHE**

**Table matière**

|  |  |  |
| --- | --- | --- |
|  |  | **Page** |
| **Mission 1 – Ajout de la partie Comptable** |  |  |
|  | **Description** | **2** |
|  | **Développement** | **4** |
|  | **Résultat** | **7** |
| **Mission 2 – Gestion de la clôture** |  |  |
|  | **Description** | **8** |
|  | **Développement** | **9** |

1. **Mission 1 – Ajout de la partie Comptable**
   1. **Description**

   **T1 - Coder la page de validation d’une fiche de frais en respectant le cas d’utilisation « Valider la fiche de frais »**

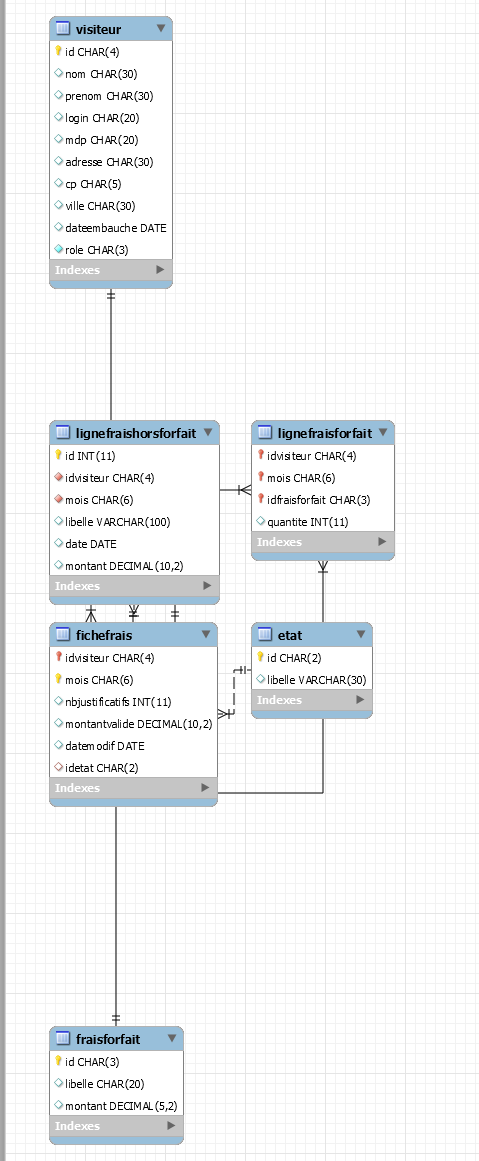
|  |
| --- |
| Nom cas d’utilisation :**Valider fiche de frais** |
| **Acteur déclencheur :** Comptable |
| **Pré conditions :** Utilisateur Comptable authentifié  Toutes les fiches du mois qui vient de s’achever sont clôturées par un script  Le comptable dispose de tous les éléments pour évaluer la conformité des frais forfaitisés |
| Scénario nominal :  1.       L’utilisateur demande à valider les fiches de frais  2.       Le système propose de choisir le visiteur et le mois concernés  3.       L’utilisateur sélectionne les informations et valide  4.       Le système affiche le détail de la fiche de frais –frais forfaitisés et hors forfait  5.       L’utilisateur actualise les informations des frais forfaitisés.  6.       Le système indique que la modification a été prise en compte et affiche les informations actualisées.  7.       L’utilisateur demande la suppression des lignes de frais hors forfait non valides – Ajouter un statut «  Refusé et/ou reporté dans table lignefraishorsforfait ou utiliser libellé si plus simple  8.       Le système modifie le libellé en ajoutant en début le texte « REFUSE : ».  9.       L'utilisateur valide la fiche.  10.    Le système passe la fiche à l’état «Validée» et met à jour la date de modification de la fiche. |
| **Exceptions :**  ·        4-a : Aucune fiche de frais n’existe, le système affiche 'Pas de fiche de frais pour ce visiteur ce mois'. Retour au 2  ·        7.a : L'utilisateur demande le report des frais hors forfait pour lesquels une facture acquittée n’a pas été reçue dans les temps.  ·        8.a : Le système ajoute la ligne hors forfait dans la fiche du mois suivant et la supprime de la fiche courante. Si la fiche du mois suivant n’existe pas, le système génère une nouvelle fiche pour le visiteur en cours de traitement et pour le mois suivant. Cette nouvelle fiche a des valeurs à 0 pour les frais forfaitisés et est dans l’état « Saisie en cours ». Retour au 9  ·        8.b : le texte ainsi complété dépasse la taille maximale du champ « libelle » : le texte est tronqué par la fin au nombre de caractères du champ « libelle » |

**T2 - Coder la page de suivi de paiement en respectant le cas d’utilisation "Suivre le paiement fiche de frais".**

|  |
| --- |
| Nom cas d’utilisation :**Suivre le paiement fiche de frais** |
| **Acteur déclencheur :** Comptable |
| **Pré conditions :** Utilisateur Comptable authentifié |
| Scénario nominal :  1.       L’utilisateur demande à suivre le paiement les fiches de frais.  2.       Le système propose de choisir une fiche de frais parmi celles à valider et mises en paiement  3.       L’utilisateur sélectionne les informations et valide  4.       Le système affiche le détail de la fiche de frais –frais forfaitisés et hors forfait  5.       L’utilisateur « Met en paiement » la fiche de frais  6.       Le système modifie l’état de la fiche à « Mise en paiement » et met à jour la date de modification. |
| **Exceptions :**  ·        4-a : Aucune fiche de frais n’existe, le système affiche 'Pas de fiche de frais pour ce visiteur ce mois'. Retour au 2   * 5.a : L’utilisateur indique que la fiche a été effectivement payée * 6.a : Le système modifie l’état de la fiche à « Remboursée » et enregistre la date de modification |

* 1. **Développement**

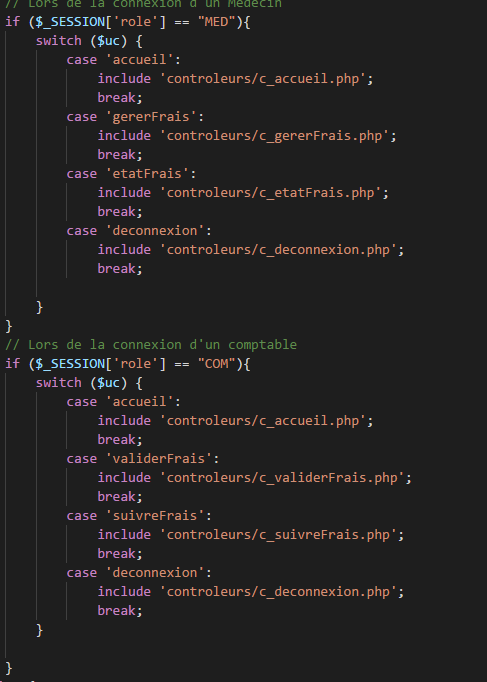
**Création d’un profil comptable en base dans la table visiteur.**



**Création de deux nouvelles pages**

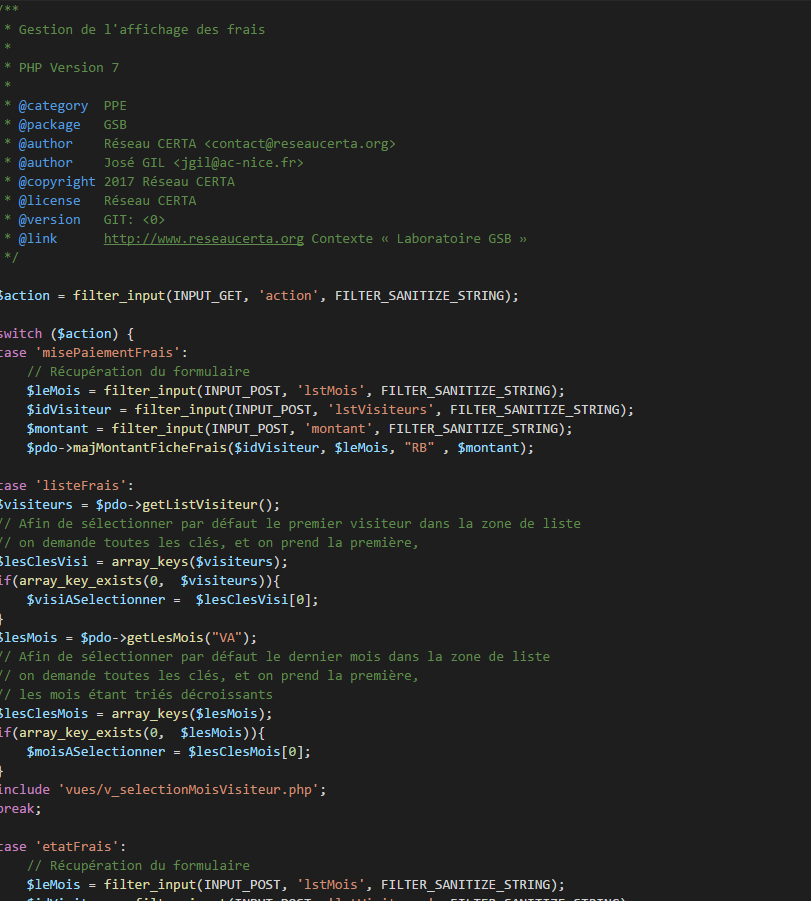
* Validation d’une fiche
* Suivi de fiche de frais

**Ajout des routes pour accéder aux pages du comptable dans le fichier index.php ont été rajoutées pour le profil comptable :**



**Création des contrôleurs**

* + C\_suivreFrais
  + C\_validerFrais

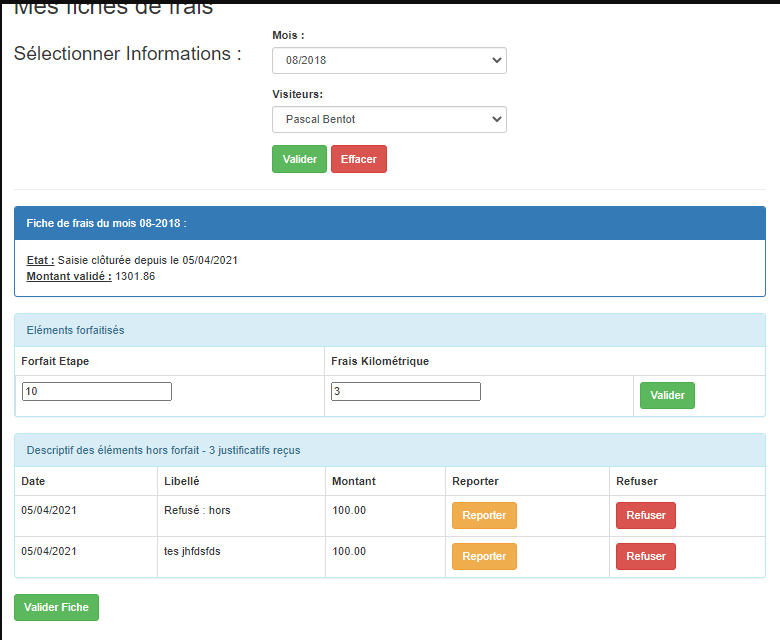


**Création des vues associées aux actions des comptables**

* + V\_valider\_frais
  + V\_selectionMoisVisiteur
  + V\_suivreFrais
  + V\_listeFraisHorsForfait

**Génération de la documentation technique**

* 1. **Résultat**

****

1. **Mission 2 – Gestion de la clôture**
   1. **Description**

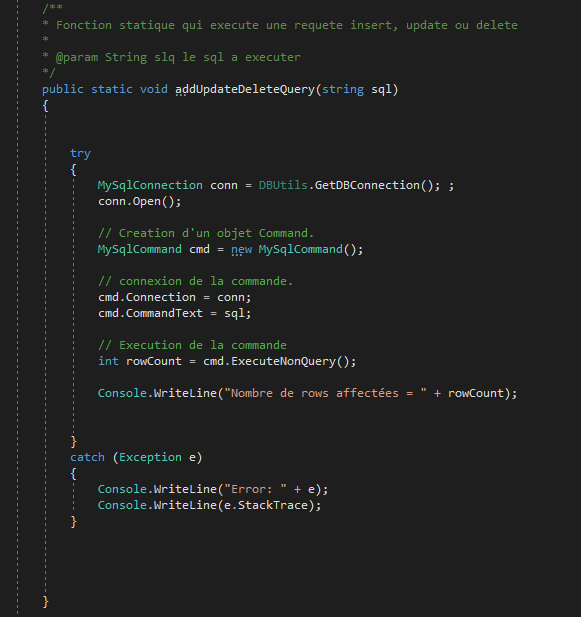
* Fiche visiteur clôturée au dernier jour du mois par l’application

1. À la première saisie pour le mois N par le visiteur, sa fiche du mois précédent est clôturée si elle ne l’est pas.

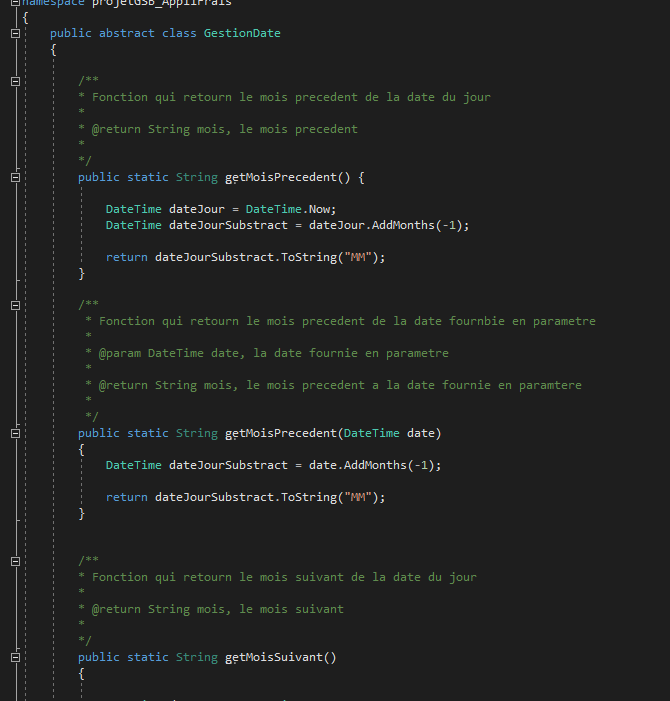
* Au début de la campagne de validation des fiches par le service comptable, un script est lancé qui clô­ture toutes les fiches non clôturées du mois qui va être traité
* Développement d’une nouvelle application C# avec VS.Net.
* Clôture de toutes les fiches créées le mois N-1 au début de la campagne de validation, c’est-à-dire à partir du 1er jour du mois N
* Mise en remboursement à partir du 20e jour du mois N des fiches créées le mois N-1.
* **T1 : Création de la classe d’accès aux données**
* Dans la nouvelle application, création d’une classe réutilisable d’accès aux données, permettant les fonc­tionnalités classiques d’accès aux données
  + Connexion à la base MySQL,
  + Exé­cution d’une requête d’administration (insert, update, delete...),
  + Gestion d’un curseur (exécution d’une requête type select et gestion du résultat avec passage à la ligne suivante, récupération d’un champ, gestion de la fin du curseur...).
* **T2 : Création d’une classe de gestion de dates**
* Abstraite avec des méthodes statiques exclusivement statiques
  + getMoisPrecedent : ne reçoit aucun paramètre et permet de retourner sous forme d’une chaîne de 2 chiffres le numéro du mois précédent par rapport à la date d’aujourd’hui (atten­tion, il faut forcément 2 chiffres : "01" pour janvier, "10" pour octobre...) ;
  + getMoisPrecedent : surcharge de la méthode précédente. Cette fois elle reçoit un objet de type DateTime en paramètre et retourne le mois précédent de cette date ;
  + getMoisSuivant : suivant la même logique que les 2 méthodes précédentes, écrire les 2 méthodes pour le mois suivant ;
  + entre : reçoit en paramètre deux numéros de jours dans le mois, et retourne vrai si la date actuelle se situe entre ces deux jours ;
  + entre : surcharge de la méthode précédente. Cette fois un troisième paramètre est reçu de type DateTime et c’est le jour de cette date qui est testée.
* **T3 : Création de l’application**
* Exploitation des 2 classes précédentes pour à intervalle régulier via Timer
  + Récupération des fiches créées du mois N-1 et leur mise à jour, en les mettant à l’état ‘CL’ ; en supposant que la campagne de validation va se passer entre le 1er et le 10 du mois courant, comparer les dates et s’assurer que l’on se trouve bien dans cet intervalle-là ;
  + de la même manière, à partir du 20e jour du mois, mettre à jour les fiches validées du mois précédent en les passant à l’état ‘RB’ ;
  + Aucun affichage
* Générer la documentation technique sous Visual Studio.
* **T4 : Création d’une service Windows**
* Exécution en tâche de fond comme un service Win­dows.

* 1. **Développement**

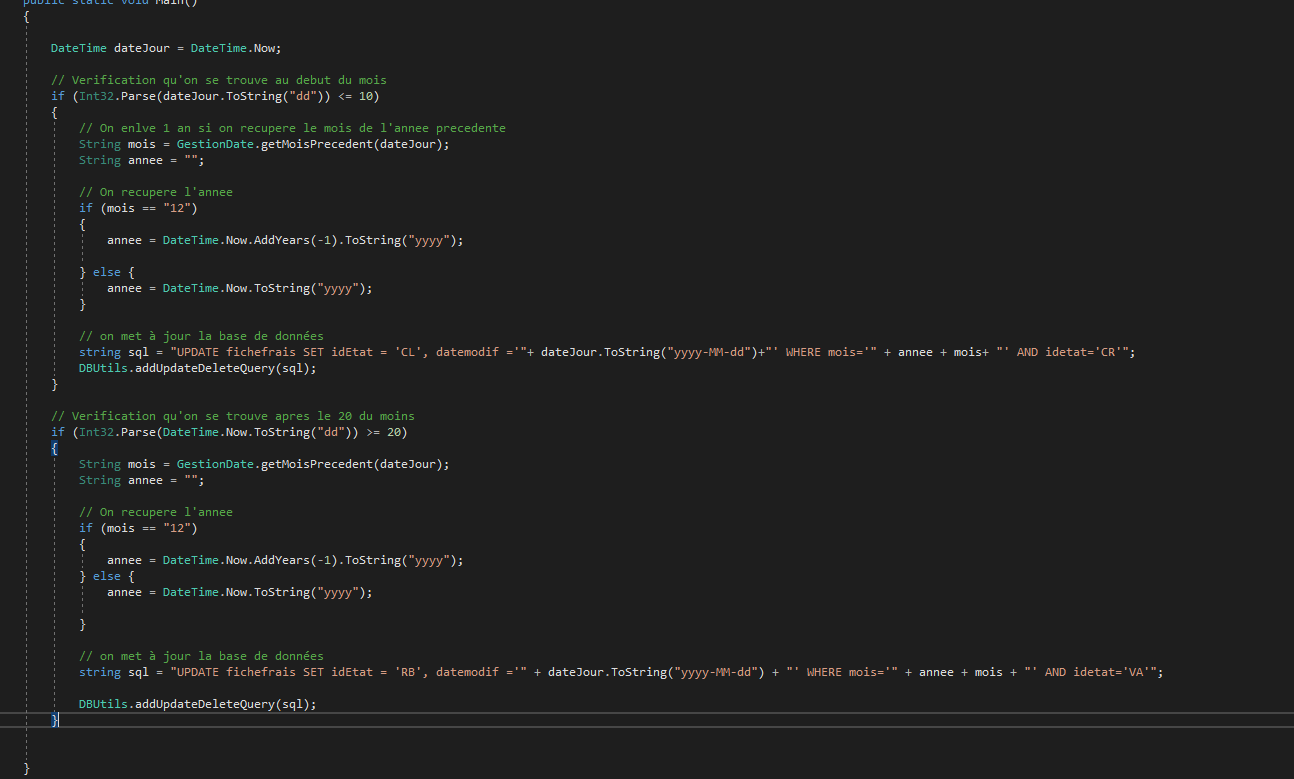
**T1 - Création de la classe d’accès aux données**



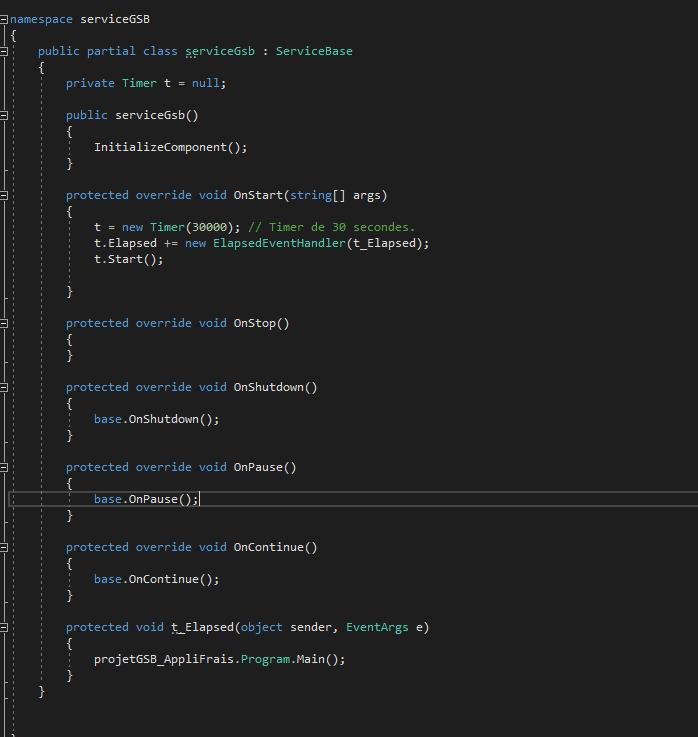
**T2 - Création d’une classe de gestion de dates**



**T3 - Création de l’application**



**T4 - Création d’un service Windows**



**Tn - Tests unitaires réalisés et lancement du service sur Windows**