|  |
| --- |
| Projet Pré-TPI : Application de gestion d’employés |

Auteur : Nithujan Jegathees waran

Date : 01.04.22

Table des matières

[1 Analyse préliminaire 3](#_Toc99718049)

[1.1 Introduction 3](#_Toc99718050)

[1.2 Objectifs 3](#_Toc99718051)

[1.3 Planification initiale 4](#_Toc99718052)

[2 Analyse / Conception 5](#_Toc99718053)

[2.1 Concept 5](#_Toc99718054)

[2.1.1 Use cases et scénarios 5](#_Toc99718055)

[2.1.2 Base de données 11](#_Toc99718056)

[2.1.3 Maquettes des pages web 13](#_Toc99718057)

[2.2 Stratégie de test 17](#_Toc99718058)

[2.3 Risques techniques 17](#_Toc99718059)

[2.4 Planification 18](#_Toc99718060)

[2.5 Dossier de conception 19](#_Toc99718061)

[3 Réalisation 21](#_Toc99718062)

[3.1 Dossier de réalisation 21](#_Toc99718063)

[3.2 Description des tests effectués 23](#_Toc99718064)

[3.3 Travail restant 24](#_Toc99718065)

[3.4 Liste des documents fournis 24](#_Toc99718066)

[4 Conclusions 25](#_Toc99718067)

[5 Annexes 26](#_Toc99718068)

[5.1 Résumé du rapport du TPI / version succincte de la documentation 26](#_Toc99718069)

[5.2 Sources – Bibliographie 26](#_Toc99718070)

[5.3 Manuel d'Installation 26](#_Toc99718071)

[5.4 Manuel d'Utilisation 26](#_Toc99718072)

# Analyse préliminaire

## Introduction

Ce projet a pour but de créer une application permettant à un département RH d’une petite ou moyenne entreprise de gérer ses employés. Elle permettra également à ces derniers d’accéder à certaines informations les concernant.

L’application sera réalisée en Python car ce langage est polyvalent et sa simplicité permet de réduire le temps d’apprentissage. Sa polyvalence permettra en outre de réaliser ce projet via une interface web qui communique avec une base de données.

Ce projet est réalisé dans le cadre du cours de Pré-TPI. Aucun travail n’a été effectué avant le début de celui-ci.

## Objectifs

L’application devra permettre aux employé/es du département RH de :

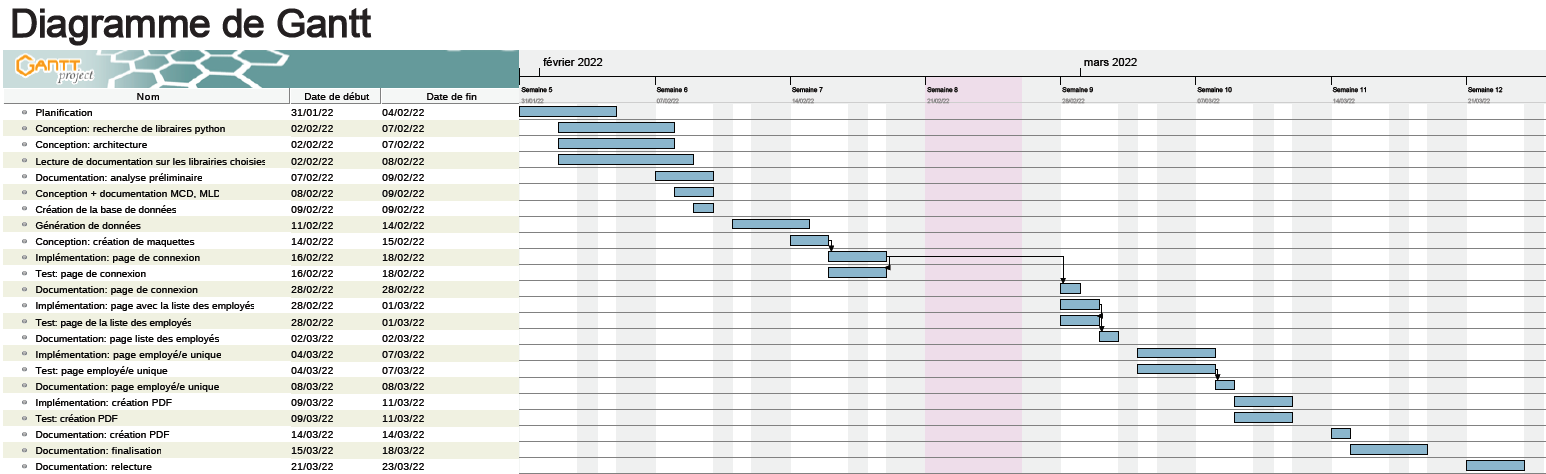
* Afficher la liste des employés
* Ajouter un employé
* Historiser un employé
* Modifier un employé
* Générer au format PDF le bulletin de salaire de l’employé
* Vérifier le temps de travail par mois de l’employé. S’il a effectué moins d’heures que son contrat ne le précise, un email d’avertissement lui est envoyé ainsi qu’à son supérieur hiérarchique.

Les employés ne faisant pas partie du département des RH (Ressources Humaines) pourront :

* Voir les informations les concernant
* Indiquer leurs heures de travail pour chaque jour de présence

## Planification initiale

La méthode de gestion choisie pour ce pré-TPI est la planification en cascade. Le cahier des charges définit de manière claire et précise les fonctionnalités de l’application à concevoir. L’avantage principal de la méthode AGILE est de fournir un produit qui répond aux attentes du client si ce dernier ne sait pas précisément ce qu’il désire or dans le cas de ce projet ce n’est pas le cas.

Voici donc le diagramme de Gantt qui en ressort :

# Analyse / Conception

## Concept

### Use cases et scénarios

*Use cases :*

1. L’application permet de se connecter avec un compte utilisateur
2. L’application permet d’accéder à nos informations personnelles relatives à l’entreprise
3. L’application permet d’entrer les tâches effectuées durant une journée de travail ainsi que le temps qu’elles ont pris
4. L’application permet de voir les tâches précédemment entrées et de les modifier avant la fin du mois
5. L’application permet au département RH de gérer les employés (voir la liste complète, ajout, modification, « suppression)
6. L’application permet de générer automatiquement des fiches de salaires à la fin d’un mois tout en vérifiant le nombre d’heures de travail effectué

*Scénarios :*

Connexion :

|  |  |
| --- | --- |
| ***Identifiant*** | AW[[1]](#footnote-1)-01, connexion au site |
| ***En tant que*** | Employé/e |
| ***Je veux*** | Me connecter au site |
| ***Pour*** | Voir des informations sur mon emploi |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Ouvrir la page du site |  | La page de connexion s’affiche |
| Entrer les identifiants et valider | Les identifiants sont corrects | La page d’accueil s’affiche |
| Les identifiants sont erronés | Un message d’erreur s’affiche |

Informations personnelles :

|  |  |
| --- | --- |
| ***Identifiant*** | AW-02, Voir mes informations personnelles |
| ***En tant que*** | Employé/e |
| ***Je veux*** | Accéder à mes informations personnelles |
| ***Pour*** | Vérifier leur exactitude |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte |  | La page d’accueil contenant diverses options s’affiche |
| Cliquer sur le menu pour voir mes informations personnelles |  | La page avec mes informations s’affiche |
| Lecture des différentes informations |  |  |

Tâches :

|  |  |
| --- | --- |
| ***Identifiant*** | AW-03, Ajouter les tâches effectuées |
| ***En tant que*** | Employé/e |
| ***Je veux*** | Entrer les tâches que j’ai effectuées |
| ***Pour*** | Mettre à jour mes heures de travail du mois |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte |  | La page d’accueil contenant diverses options s’affiche |
| Cliquer sur le menu pour communiquer le travail effectuée |  | La page du formulaire permettant d’ajouter une nouvelle tâche effectuée s’affiche |
| Remplissage des différents champs du formulaire | Au moins un des champs contient une erreur de format (date, heure) |  |
| Les champs obligatoires sont remplis : nom du projet, titre de la tâche ainsi que les heures de début et de fin | La page avec la liste des tâches effectuées durant le mois en cours s’affiche et est à jour |
| Un des champs obligatoire n’est pas rempli | Un message d’erreur s’affiche et demande de remplir les champs obligatoires |

Modifier les tâches :

|  |  |
| --- | --- |
| ***Identifiant*** | AW-04, Editer les tâches précédemment entrées |
| ***En tant que*** | Employé/e |
| ***Je veux*** | Modifier un des champs d’une tâche précédemment entrées |
| ***Pour*** | Corriger une erreur |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte |  | La page d’accueil contenant diverses options s’affiche |
| Cliquer sur le menu pour voir les tâches effectuées |  | La page avec les tâches effectuées durant le mois en cours s’affiche |
| Cliquer sur l’option permettant de modifier une des tâches |  | Le formulaire pour modifier une tâche s’affiche |
| Modification du/des champs désiré/s |  |  |
| Validation de la mise à jour des informations |  | La page avec les tâches s’affiche et est à jour |

Gestion de employés :

|  |  |
| --- | --- |
| ***Identifiant*** | AW-05A, Ajouter un/e employé/ |
| ***En tant que*** | Employé/e RH |
| ***Je veux*** | Ajouter un/e employé/e |
| ***Pour*** | Mettre à jour la liste des employés |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte | Être du département RH | La page d’accueil standard mais avec des options en plus pour gérer les employés s’affiche |
| Cliquer sur l’option pour ajouter un/e employé/e |  | Le formulaire pour ajouter un/e employé/e s’affiche |
| Insertion des valeurs dans les champs du formulaire |  |  |
| Validation du formulaire | Les champs obligatoires, c’est à dire tous sauf le numéro de téléphone et le nom du supérieur sont remplis | La page avec la liste des employés mise à jour s’affiche |
| Il manque un des champs requis | Un message d’erreur expliquant le problème s’affiche |

|  |  |
| --- | --- |
| ***Identifiant*** | AW-05B, Historiser un/e employé/ |
| ***En tant que*** | Employé/e RH |
| ***Je veux*** | Enlever un/e employé/e de la liste des employés |
| ***Pour*** | Mettre à jour la liste des employés |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte | Être du département RH | La page d’accueil standard mais avec des options en plus pour gérer les employés s’affiche |
| Cliquer sur l’option pour afficher la liste des employés |  | La page avec la liste des employés s’affiche |
| Cliquer sur l’option pour historiser un/e employé/e précis/e |  | Un message d’avertissement s’affiche |
| Validation de l’archivage |  | La page avec la liste des employés est mise à jour |

|  |  |
| --- | --- |
| ***Identifiant*** | AW-05C, Modifier un/e employé/ |
| ***En tant que*** | Employé/e RH |
| ***Je veux*** | Modifier les informations d’un/e employé/e |
| ***Pour*** | Mettre à jour ses informations |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| Connexion au compte | Être du département RH | La page d’accueil standard mais avec des options en plus pour gérer les employés s’affiche |
| Cliquer sur l’option du menu pour afficher la liste des employés |  | La page avec la liste des employés s’affiche |
| Cliquer sur l’option pour modifier un/e employé/e précis/e |  | La page de modification d’un/e employé/e s’affiche |
| Modification des champs désirés |  |  |
| Validation du formulaire | Tous les champs modifiables obligatoires sont remplis : nom, adresse complète, fonction, taux de travail | Les données sont mise à jour dans la base de données |
| Certains champs modifiables obligatoires sont laissés vides | Un message d’erreur demandant de ne pas laissé les champs obligatoires vides s’affiche |

Fiches de salaire :

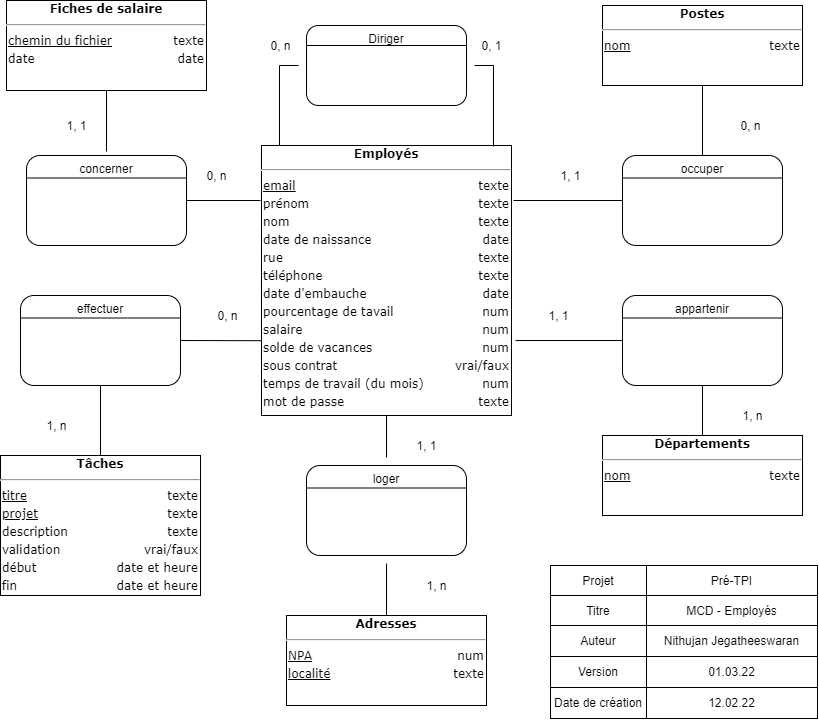
|  |  |
| --- | --- |
| ***Identifiant*** | AW-06, Générer des fiches de salaire |
| ***En tant que*** | Employé/e RH |
| ***Je veux*** | Que les fiches de salaire soient créées automatiquement |
| ***Pour*** | Les mettre à disposition des employés |
| ***Priorité*** | M |

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Condition*** | ***Réaction*** |
| On passe au premier jour du mois |  | La tâche planifiée pour générer les fiches de salaires se lance |
| Le serveur vérifie les données de travail d’un/e employé/e | A travaillé moins d’heures que précisé dans le contrat | Un mail est envoyé à l’employé/e et à son/sa supérieur/e hiérarchique |
| A travaillé assez d’heure |  |
| Le serveur crée les fiches de salaire |  | Les fiches de salaires sont créées dans le répertoire désigné |

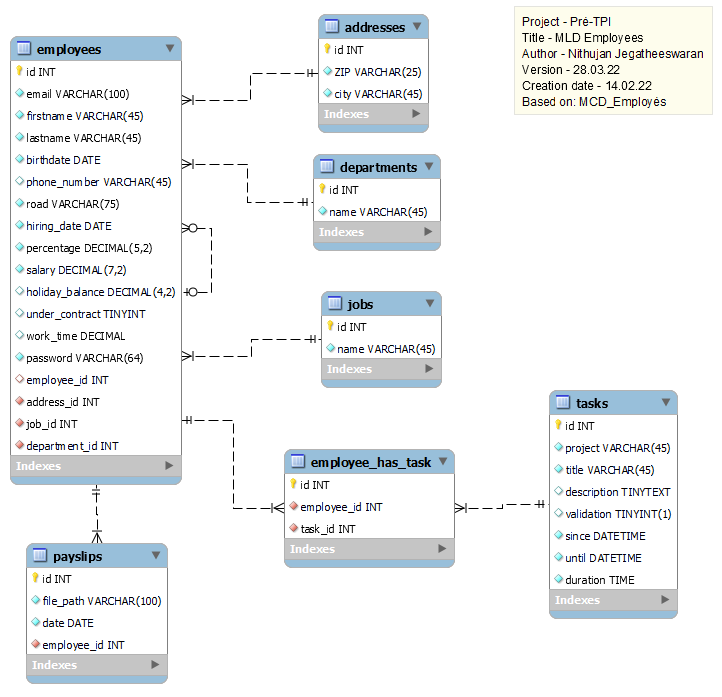
### Base de données

Le SGBDR choisi est MySQL car il est gratuit, permet de s’adapter à un grand nombre d’applications. De plus, le fonctionnement de base de ce dernier est connu car il a été utilisé durant la formation FPA (Formation Professionnelle Accélérée) d’informaticien.

* MCD :

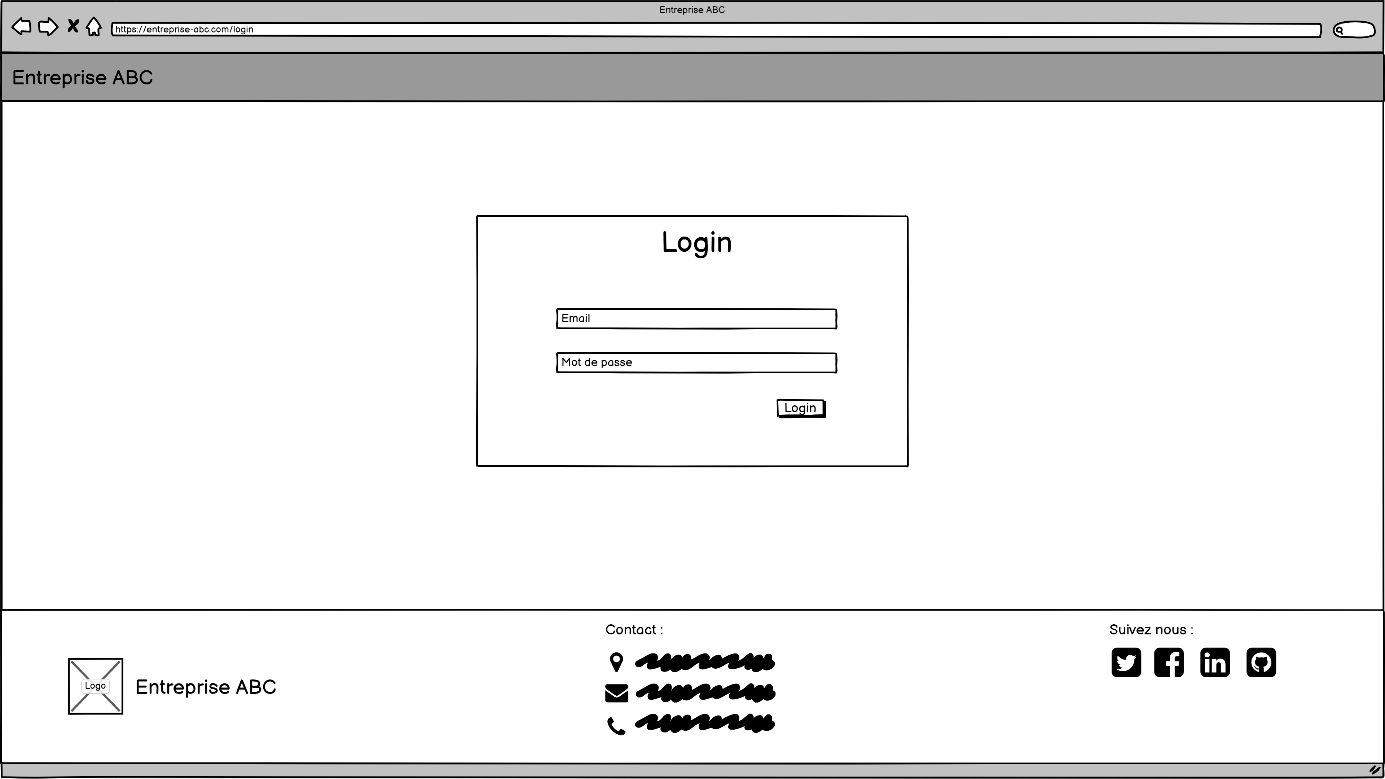


* MLD

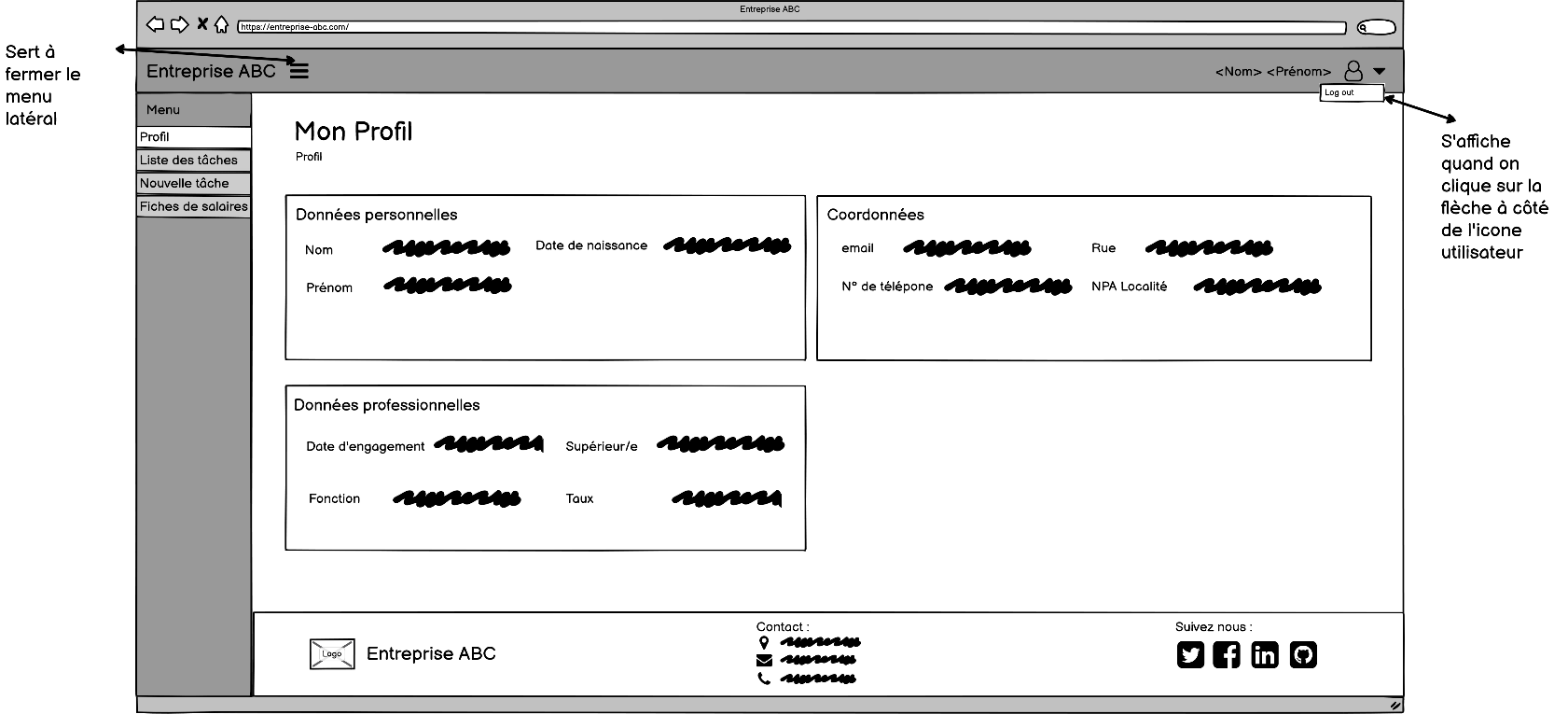


### Maquettes des pages web

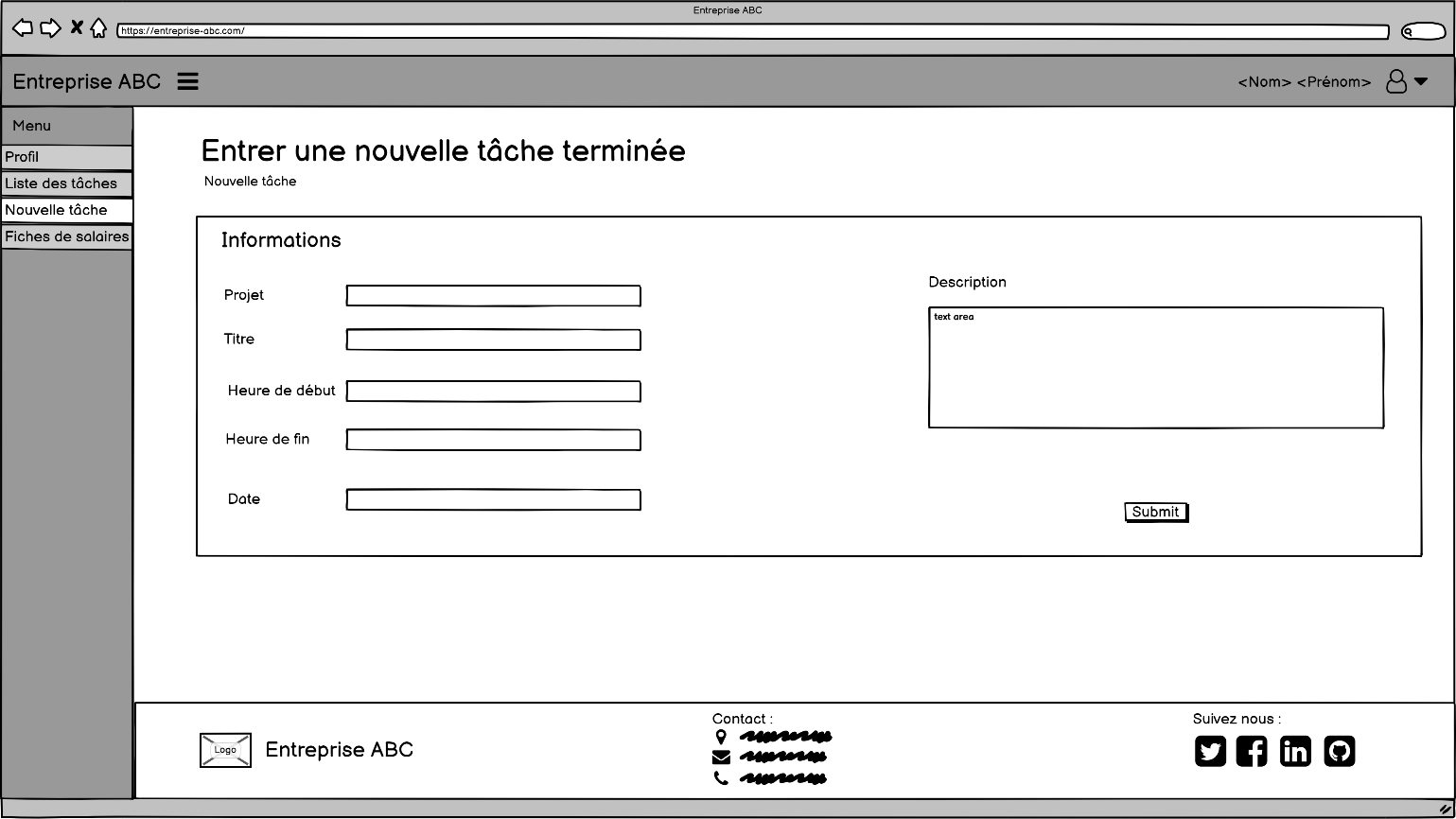
* Page de connexion :

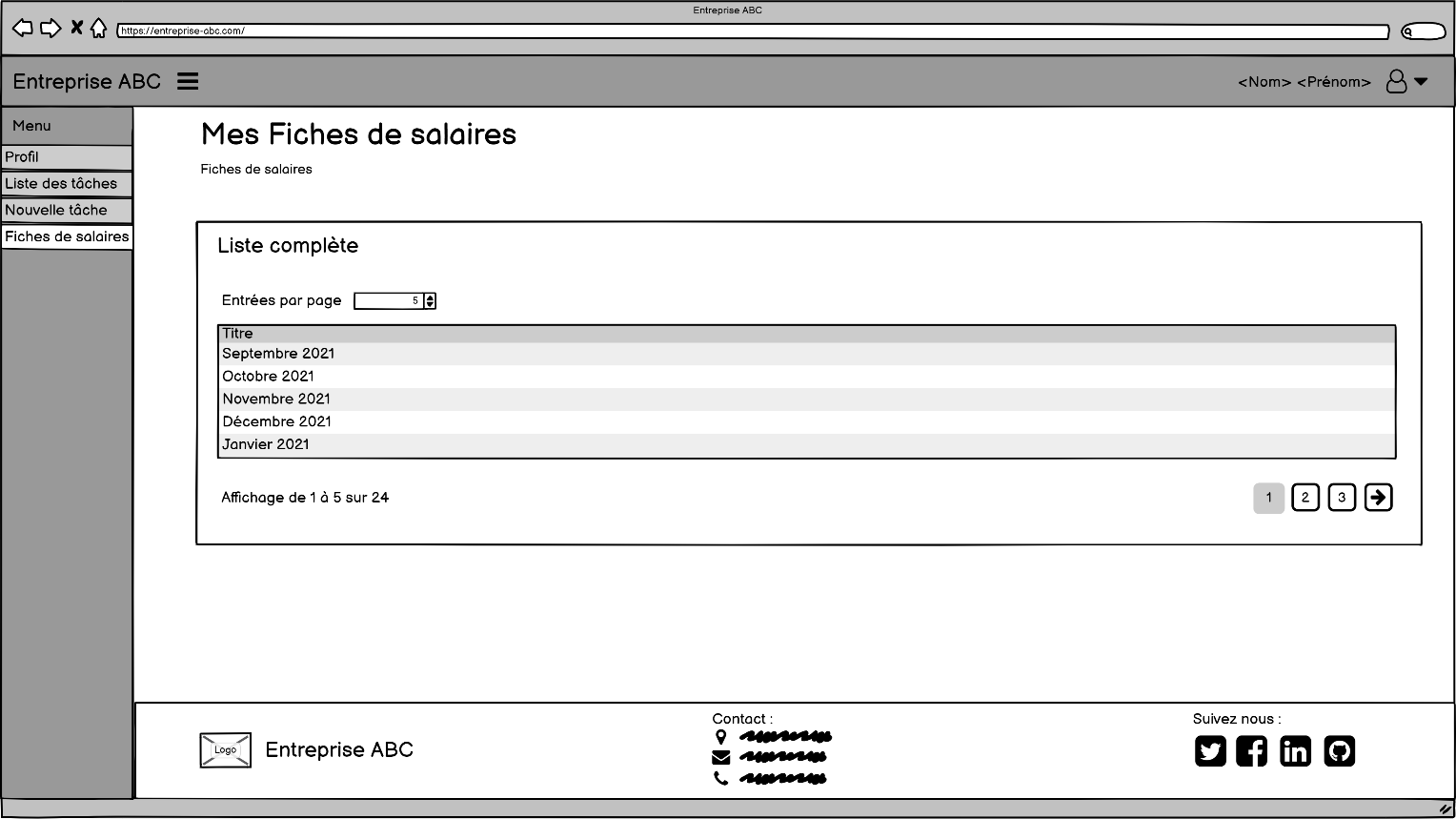


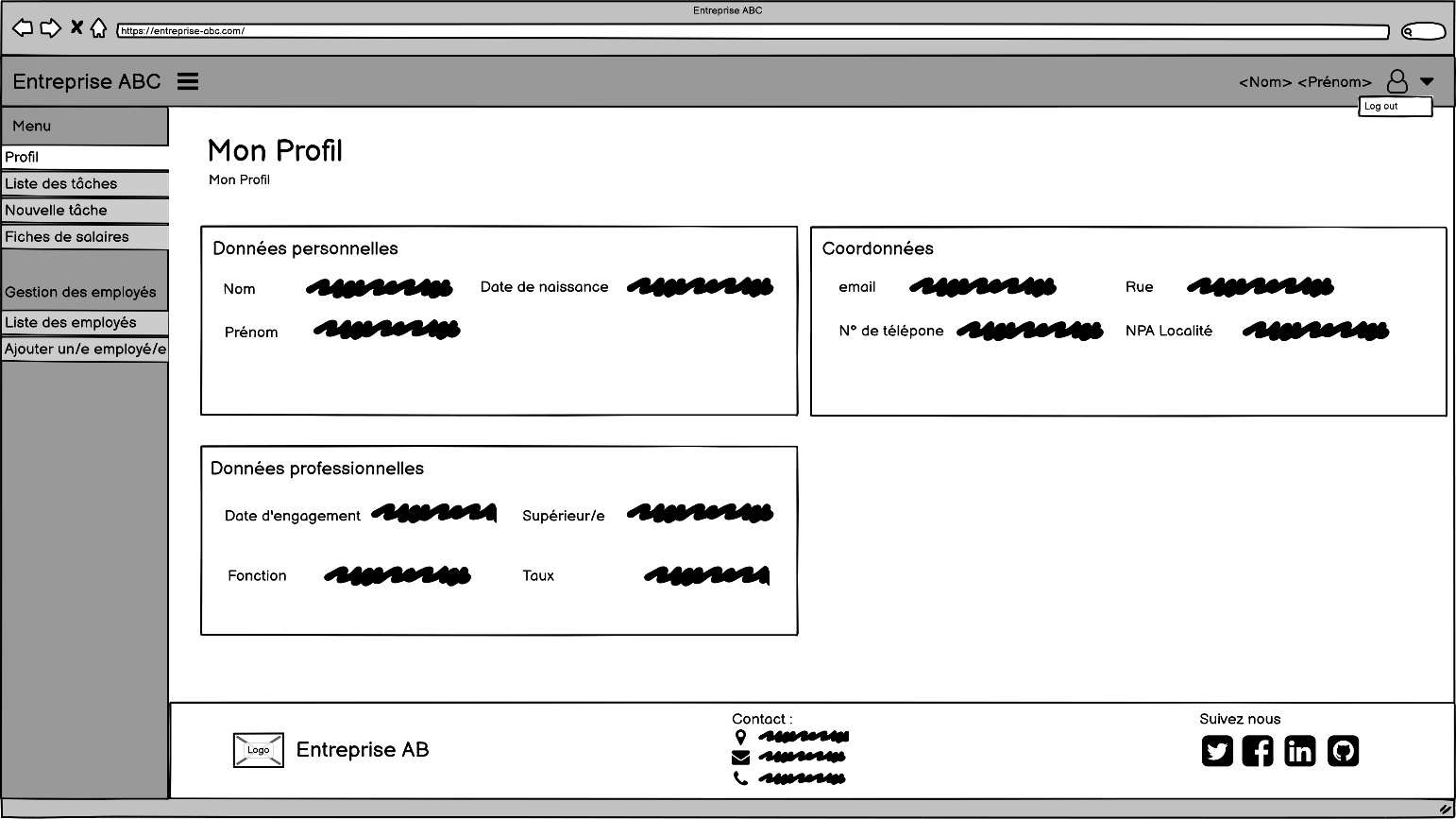
* Employé/e non RH :
  + Page de profil :



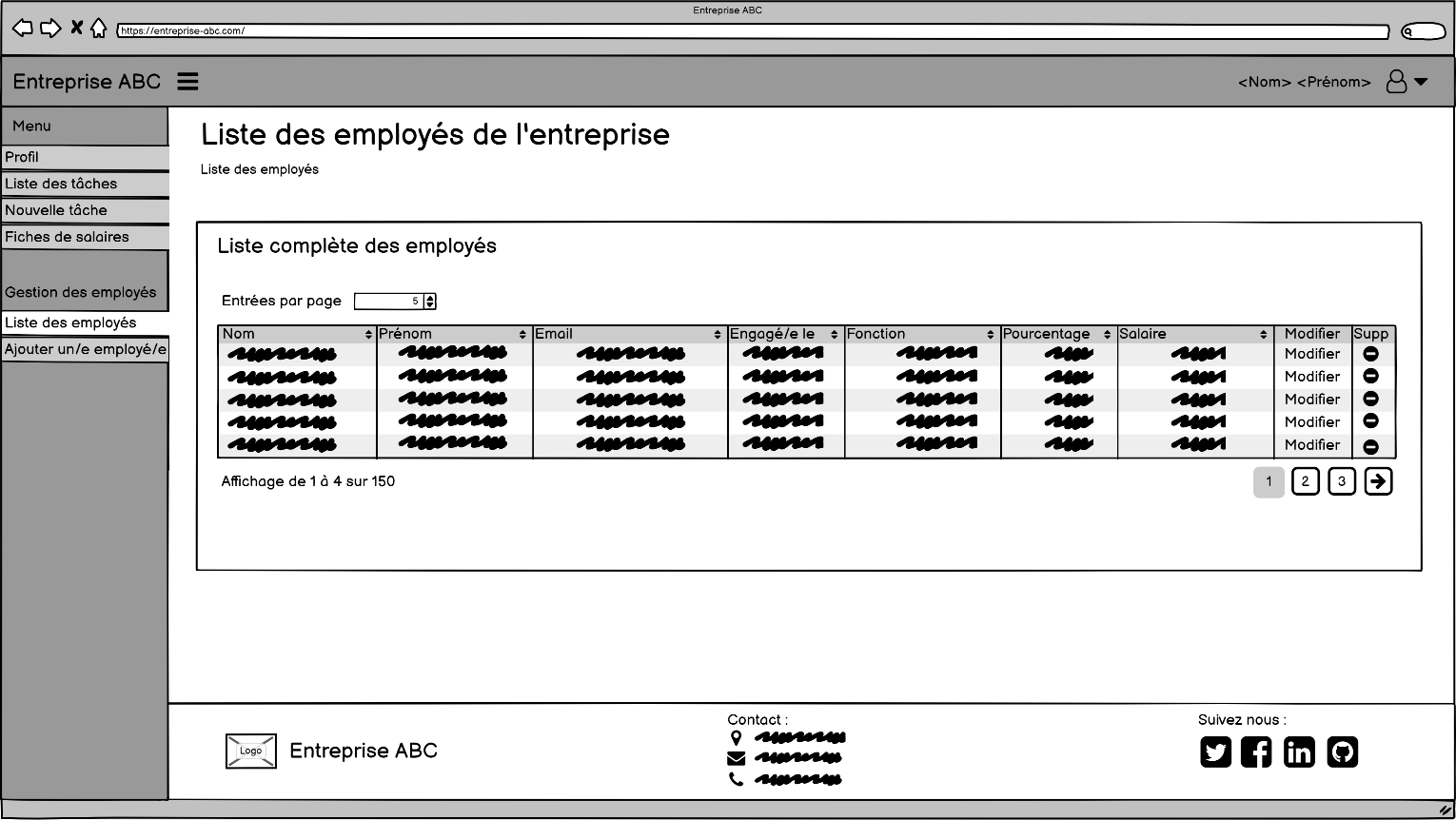
* + Page pour ajouter une tâche terminée :



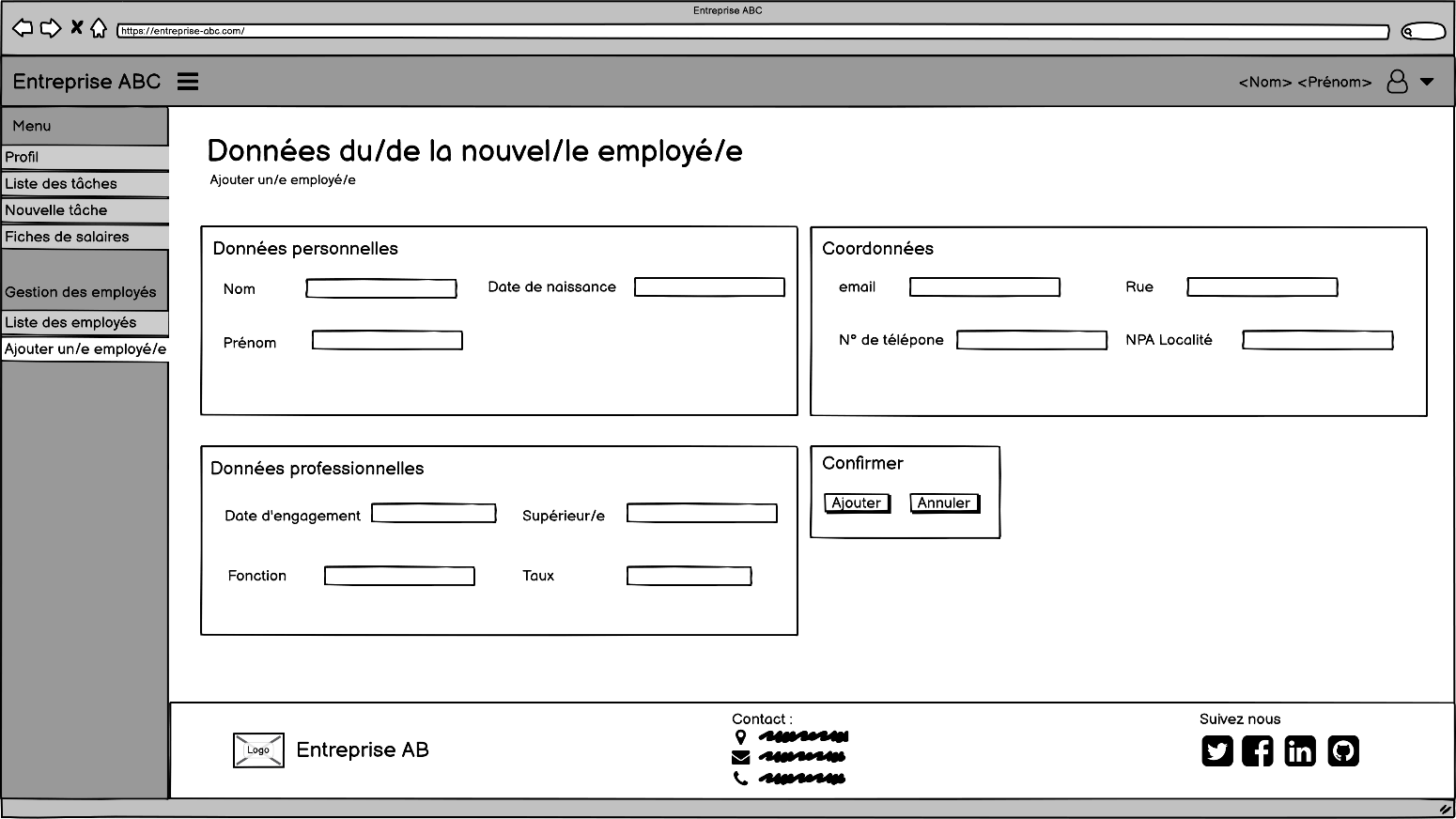
* + Page permettant de consulter les fiches de salaires
* Employé/e RH
* Page de profil montrant la différence dans le menu :



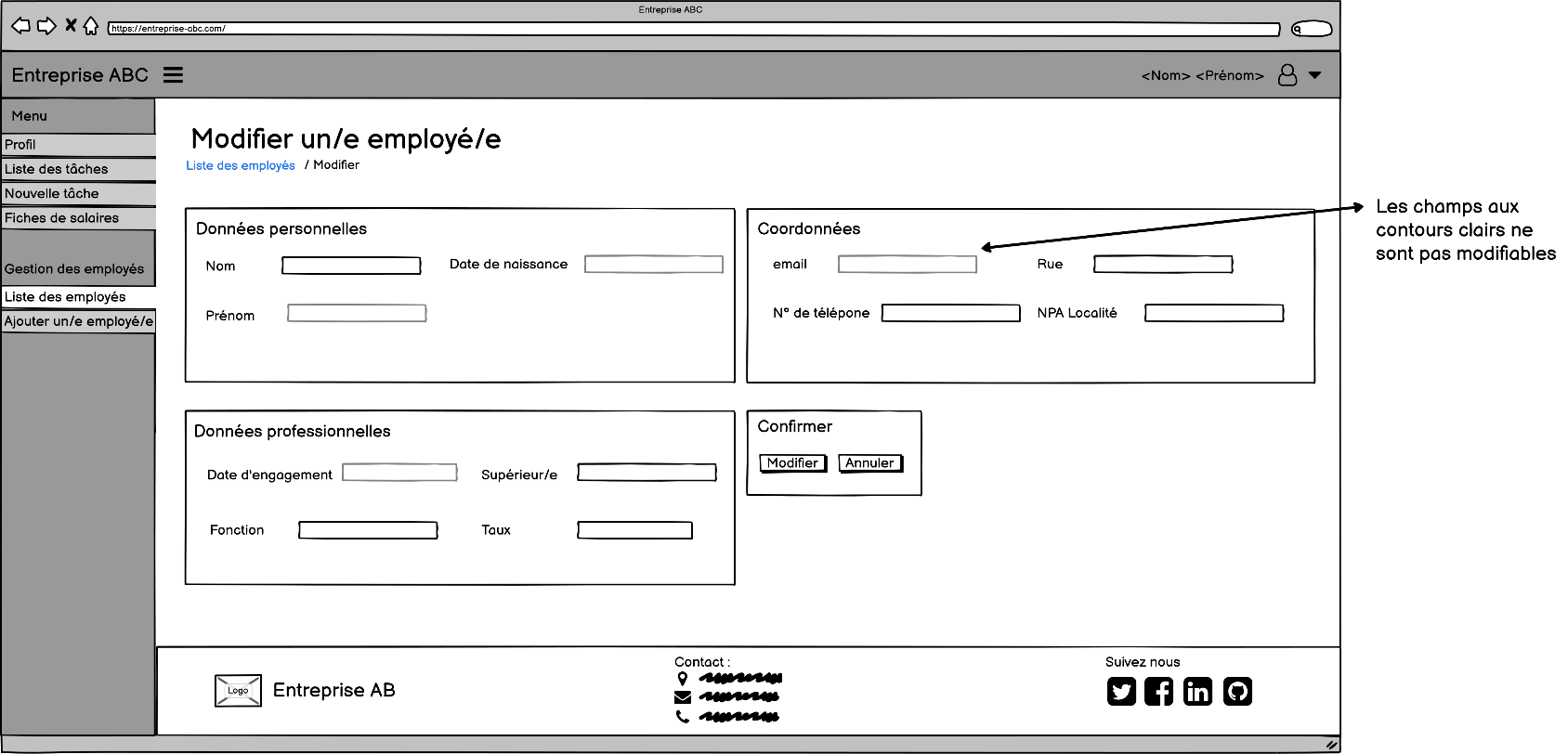
* Page avec la liste des employés :



* Page pour ajouter un/e employé/e :



* Page pour modifier un/e employé/e existant/e :



## Stratégie de test

Les tests de l’application web seront tous manuels. Ils seront effectués en parallèle à l’implémentation des diverses fonctionnalités. Ils seront faits sur plusieurs navigateurs : Firefox, Chrome et Microsoft Edge.

Le but principal des divers tests sera de permettre une utilisation basique des fonctionnalités proposées tout en prévenant d’éventuelles manipulations erronées.

Pour rendre les tests des différentes pages web crédibles, la base de données liée au programme contiendra des données réalistes et en nombre suffisant.

Afin de tester l’ergonomie et les différentes fonctionnalités de l’application des connaissances et/ou certains camarades de classe pourront m’assister.

## Risques techniques

Mes Connaissances de Python et des différents modules existants sont basiques, il faudra donc consacrer du temps afin d’apprendre ceux-ci.

Afin de permettre une bonne compréhension de la technologie utilisée, les différents fonctions et modules seront testés via du code jetable et des POC (Proof Of Concept). Ces tests seront parfois effectués en dehors des heures de travail.

## C:\Pré-TPI\documentation\planif-courante.pngPlanification

Retard du principalement à la prise en main des librairies, la documentation à double à certains moment (Flask-SQLAlchemy - SQLAlchemy). Les MLD / MCD ont été mise à jour afin d’améliorer la sous-traitance du projet s’elle devait avoir lieu.

## 

## Dossier de conception

**Logiciels utilisés** :

Ce projet a été réalisé sur un ordinateur avec Windows 10 et les logiciels suivants ont été utilisé :

* Draw.io v14.9.6
* MySQL Workbench 8.0.27
* HeidiSQL 11.3.0.6295 (64 bit)
* dbForge Data Generator for MySQL (Version d’essai) v2.4.526
* Balsamiq Cloud
* PyCharm 2021.2.3 (Edition professionnelle)
* Navigateurs Web :
* Opera
* Firefox
* Google Chrome

Draw.io a permis de modéliser les tables de la base de données via un Modèle Conceptuel de Donnée (MLD). MySQL Workbench a été utilisé lors de la création du Modèle Logique de Données (MLD) et du script de génération de la base de données. DbForge est un logiciel payant qui a servi afin de créer des données réalistes et en nombre suffisant soit cinquante entrées par tables. Enfin HeidiSQL est utilisé comme serveur MySQL pour communiquer avec l’application

Balsamiq Cloud est le logiciel qui a été utilisé pour concevoir les maquettes web présentes dans ce document. PyCharm est une IDE qui permet de développer des applications en Python. La présence de plusieurs navigateurs web différents se justifie par le besoin de tester l’application sur des environnements de test variés

**Base de données**

Les données ajoutées aux tables ‘employees’ et ‘tasks’ via l’application web seront vérifiées et approuvées via l’application.

**Table ‘employees’**

Table qui contient les données des employés d’une entreprise. Les employés ne possèdent qu’un seul et unique supérieur hiérarchique ou ils n’en ont pas, dans le second cas cela signifie qu’ils sont eux-mêmes des superviseurs. Les heures de travail doivent être remise à zéro lors de la génération de leurs fiches de salaires. Les champs de texte ont été limités dans le nombre de caractères qu’ils peuvent contenir. Les champs obligatoires de cette table qui ne font pas partie du formulaire utilisé lors de l’ajout d’un/e employé/e ont tous une valeur par défaut :

* phone\_number : ‘’
* holiday\_balance : 0
* under\_contract : 1
* work\_time : 0
* employee\_id : NULL

Ces valeurs par défaut ont une réelle utilité lorsqu’un/e nouvel/le employé/e est ajouté/e à la base de données via l’application web. En effet le formulaire prévu à cet effet ne contient pas de champs pour toutes les colonnes de la table ‘employees’. Le mot de passe de chaque employé est créé via l’application après avoir validé tous les champs du formulaire.

**Table ‘tasks’**

Cette table est conçue afin de pourvoir accueillir les données des tâches accomplies par les employés d’une entreprise. Tous ses champs sont obligatoires sauf les colonnes ‘description’ et ‘validation’. L’ajout ou la modification d’une tâche par un/e employé/e met à jour ses heures de travail.

**Table ‘addresses’, ‘departments’ et ‘jobs’**

Ces tables sont présentes afin de mieux décrire les entrées de la table ‘employees’. Les tables spécifiques à l’entreprise, c’est-à-dire les tables ‘departments’ et ‘jobs’, devront donc être des listes exhaustives afin de permettre d’ajouter facilement un employé à la base de données via un formulaire qui présente ces listes.

**Table ‘payslips’**

Cette table a pour objectif de pouvoir accéder aux fiches de salaires des employés. La colonne ‘file\_path’ permettra au serveur web d’accéder et de mettre à disposition les fiches de salaire crées au format PDF. La création de ces bulletins sera abordée plus loin dans ce document.

**Code**

Le projet a été réalisé avec une architecture MVT (Model-View-Template) …….

**Design web**

(insérer captures d’écran et comparer avec maquettes)

# Réalisation

## Dossier de réalisation

**Librairies tierces utilisées**

Flask – Framework web permettant de concevoir un site web en python, propose des fonctions minimalistes contrairement à son concurrent Django mais il est plus simple à mettre en place et laisse plus de liberté en termes de développement. Ce framework utilise d’autres librairies externes, notamment Jinja2 et Werkzeug. Le premier permet de rendre plus dynamique la manipulation des fichiers HTML du site via des fonctionnalités de templating intéressantes qui permettent de ne pas se répéter. Werkzeug est un Web Server Gateway Interface (WSGI), soit une interface qui permet à l’application Flask d’effectuer des communications sur internet.

Flask-SQLAlchemy – Extension pour Flask qui a pour but l’utilisation du packages SQLAlchemy. Ce package est un Object Relational Mapper (ORM) qui permet de générer des requêtes SQL via l’utilisation de classes et de fonction prédéfinies. L’extension Flask permet de faire communiquer les deux lors, notamment, de la mise en route de l’application. L’utilisation de fonctions permet de ne pas avoir à gérer le risque d’injection SQL car celles-ci s’occupent sécuriser les entrées des utilisateurs utilisées lors des requêtes. En plus d’une base de données MySQL, ce package fonctionne également les bases de données SQLite, Postgresql, Oracle, MS-SQL, Firebird, Sybase.

mysql-connector-python – Package python qui permet la communication avec une base de données mysql. Il permet notamment à Flask-SQLAlchemy de faire ses requêtes vers une base de données.

python-dateutil – Package python qui offre la possibilité d’effectuer des actions sur des objets de classe date, datetime et time. Il permet, par exemple de calculer une différence entre deux dates et renvoyé le résultat en plusieurs unité de temps différentes.

**Arborescence des répertoires de l’application :**

Une image contenant texte

Description générée automatiquement

Description du contenu des répertoires :

* Les répertoires commençant par des points (.) et les \_\_pychache\_\_ sont créés automatiquement par l’IDE afin de faire fonctionner le programme
* employee\_mgmt\_app :
  + run.py : fichier lancé par le programme pour se lancer
  + config.py : contient toutes les variables qui aident à configurer l’application
  + requirements.txt : liste des packages python nécessaire pour faire fonctionner le projet
* app :
  + \_\_init\_\_.py : fichier d’initialisation de l’application
  + model :
    - Les modules python propres à chaque table de la base de données
    - Un module contenant toutes les classes python des tables de la base de données MySQL
    - test-queries.py : un fichier commenté qui permet de mieux comprendre les fonctions SQLAlchemy permettant de récupérer des informations de la base de données
* static :
  + Comme son nom l’indique ce répertoire contient tous les éléments statiques comme liés à l’application Flask (CSS, Javascript, images, etc)
* templates :
  + Toutes les pages HTML sont placés dans ce dossier

## Description des tests effectués

|  |  |  |
| --- | --- | --- |
| Scénario | Nithujan Jegatheeswaran (auteur du projet) | Sathujan Jegatheeswaran (frère de l’auteur) |
| AW-01 : Connexion au site |  |  |
| AW-02 : Voir les informations |  |  |
| AW-02A : Ajouter les tâches effectuées |  |  |
| AW-02B : Editer les tâches précédemment entrées |  |  |
| AW-04A, Ajouter un/e employé/ |  |  |
| AW-04B, Historiser un/e employé/ |  |  |
| AW-04C, Modifier un/e employé/ |  |  |
| AW-05, Générer des fiches de salaire |  |  |
| *Code couleur : vert= réussite, jaune = erreur, gris = non exécuté / non implémenté* | | |

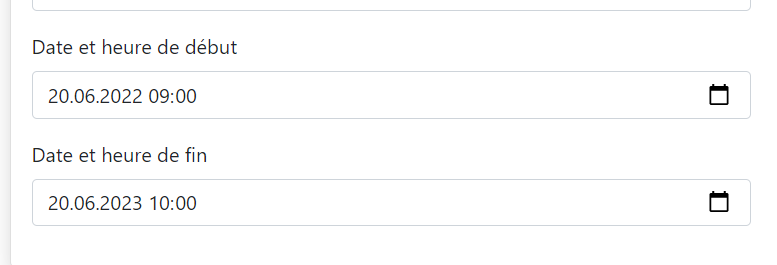
**Environnement de test :**

Les tests ont été réalisés sur Opera, Google Chrome et Firefox. Les ordinateurs où les tests ont été faits avaient Windows 10 comme système d’exploitation.

**AW02A & AW02B :**

. Les principales erreurs concernaient les champs de date et date-heure qui n’avait pas de valeur minimum et maximum, ce qui provoquait des erreurs dans le programme Python.

Valeurs entrées sur le formulaire pour ajouter une nouvelle tâche :



Erreur générée :



Exemple de correction :

Une image contenant texte

Description générée automatiquement

**AW05 :**

Ce test n’a pas été réalisé car la fonctionnalité n’a pas été implémenté.

## Travail restant

Les tests de robustesse des différents formulaires pourraient être poussé plus loin pour gérer un maximum de possibilité. Au niveau des pages web disponibles, la page concernant les fiches de salaire n’a pas pu être implémenté mais une librairie utile a déjà été sélectionnée : PyPDF2. Il manque tout de même la partie concernant la planification de tâches qui n’a pas été abordée durant ce projet.

## Liste des documents fournis

* Le rapport de projet v1.0
* Le manuel d’installation v1.0
* Le manuel d’utilisation v0.9

# Conclusions

En reprenant la liste des fonctionnalités à mettre en place pour cette application, les objectifs qui concernent la génération de bulletin de salaire d’employé au format PDF et l’envoi d’un mail d’avertissement en cas d’heure de travail insuffisant n’ont pas été atteint.

La réalisation de ces objectifs a dû être retardée car la mise en place d’une base de données cohérente et réaliste a été privilégiée. En effet toutes les données liées à l’application dépendent de cette base de données qui doit donc être complète et avoir une structure qui permette de ne pas avoir à l’altérer.

En plus de la base de données les formulaires permettant d’entrer ou de modifier des champs dans la base de données ont été longuement testés et les différentes valeurs entrées par l’utilisateur ont été validées par le programme.

Si ce projet devait être poursuivi, il serait prioritaire d’implémenter la fonction permettant de créer les fiches de salaires des employés à chaque fin de mois. Suite à cela il serait intéressant d’optimiser le code pour éviter un maximum de se répéter. Les formulaires pourront être gérés via l’extension Flask-WTF qui permet de valider les formulaires et de gérer les messages d’erreurs renvoyés à l’utilisateur.

# Annexes

## Résumé du rapport du TPI / version succincte de la documentation

Ce Pré-TPI a été réalisé du 31.01.22 au 01.04.22. Ses objectifs principaux consistaient en la création d’un application web en utilisant le langage de programmation et la base de données voulu.

Cette application devait permettre à un département RH d’une entreprise de gérer ses employés : ajout, modification et archivage. Les employés peuvent aussi se connecter afin d’accéder à diverses informations les concernant. Ils peuvent également indiquer le temps qu’ils ont passé à effectuer des tâches.

## Sources – Bibliographie

(*voir journal de travail*)

## Manuel d'Installation

## Manuel d'Utilisation

1. AW : Application Web [↑](#footnote-ref-1)